



# MODULE 03

## Contrôle domotique

# MODULE 03

## SÉANCE SYSTÈME 01

### TP D'INFORMATIQUE

Durée 2h30

## PRISE EN MAIN DE L'API REST DECONZ ET ANALYSE WIRESHARK

### BLOC DE COMPÉTENCES

U6 - VALORISATION DE LA DONNÉE ET CYBERSÉCURITÉ

### COMPÉTENCE(S)

C10 EXPLOITER UN RÉSEAU INFORMATIQUE

### OBJECTIF PÉDAGOGIQUE

Étude de la configuration d'une installation domotique : utilisation d'une API HTTP (REST). Analyse Wireshark des données.

### CONNAISSANCES ISSUES DU RÉFÉRENTIEL

- |   |          |
|---|----------|
| • Réseaux IoT   | Niveau 2 |
| • Réseaux informatiques (protocoles, équipements et outils usuels et industriels) | Niveau 3 |
| • Outils logiciels d'évaluation, de traçabilité de l'information                  | Niveau 3 |

### CONNAISSANCES OPÉRATIONNALISÉES

- |   |          |
|---|----------|
| • Utilisation d'une API REST à partir de sa documentation | Niveau 3 |
| • Analyser une communication réseau avec Wireshark        | Niveau 2 |

## TD

# Comprendre l'API REST

### Qu'est ce que le REST ?

Une API, ou interface de programme d'application, est un ensemble de règles qui définissent comment les applications ou les unités peuvent se connecter et communiquer entre eux. Une API REST est une API conforme aux principes de conception du style architectural REST ou Representational State Transfer. C'est pour cette raison que les API REST sont parfois appelées API RESTful .

Les API REST communiquent **via des requêtes HTTP** pour exécuter des fonctions de base de données standard telles que la **création, la lecture, la mise à jour et la suppression** d'enregistrements dans une ressource.

La représentation de la ressource peuvent être fournies à un client dans pratiquement n'importe quel format, y compris JavaScript Object Notation (JSON), HTML, XML, PHP ou un texte en clair

### Structure d'une requête



Figure 1: Structure d'une requête HTTP

Dans une requête HTTP, nous aurons plusieurs paramètres dont la méthode et l'URL qui nous intéressent ici. D'abord intervient la méthode puis la ressource via l'URL.

A quoi sert la méthode ?

A quoi sert la ressource (URL) ?

## Méthodes et ressource dans une requête

Les méthodes de requête HTTP permettent de définir l'action que l'on souhaite réaliser sur la ressource indiquée. Il en existe plusieurs mais nous allons nous focaliser sur les suivantes GET, POST, PUT, DELETE. Quand à elle, la ressource permet de définir la demande de l'utilisateur.

Définir dans quel cas chaque méthode est utilisée dans une API REST :

**GET** :

**POST** :

**PUT**

**DELETE**

Quels caractères sont utilisés comme séparateurs dans une trame HTTP ? Indiquer les 4 caractères de fin de trame, comment les écrire dans un code en langage C ?

## Format JSON

Les données sont envoyées au format JSON.

Qu'est ce que le format JSON ?

Donner un exemple de données au format JSON :

## TP

# L'API deconz

### Objectif

Comprendre l'API utilisée par deconz ([https://dresden-elektronik.github.io/deconz-rest-doc/about\\_rest/](https://dresden-elektronik.github.io/deconz-rest-doc/about_rest/))

### Préparation

Dans cette partie nous aurons besoin d'un client REST permettant de tester les requêtes HTTP.

| Installer une extension REST Client pour Firefox ou Postman ou Boomrang pour Chrome ou Edge

### Les bases de l'API

Selon le lien précédent, donner les informations suivantes :

| Quelle ressource faut ajouter à l'URL pour avoir des informations sur toutes les lumières ?

| Quelle ressource faut ajouter à l'URL pour avoir des informations sur la lumière numéro 1 ?

| Quelle ressource faut ajouter à l'URL pour avoir des informations sur tous les capteurs ?

| Que doivent faire les clients avant de pouvoir interroger l'API ?

| Quels sont les avantages à utiliser l'API REST ?

## Trouver l'adresse IP de la passerelle domotique

Pour scanner un réseau, vous pouvez utiliser le logiciel « Advanced IP Scanner »

Installer le logiciel précédent sur votre poste

Scanner le réseau 172.20.21.0-254/24

Trouver le module ayant comme site web gatewayCIELIR

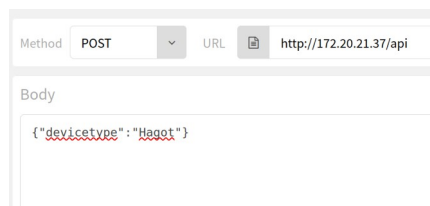
## Obtention d'une clé API

Donner les requêtes HTTP permettant de trouver l'adresse ip de la passerelle

Donner les requêtes HTTP permettant récupérer une clé pour l'API en utilisant votre nom

Pour obtenir une clé de l'API, il faut demander à l'enseignant d'activer la demande de clé.

Ouvrir le client rest de votre navigateur, le compléter avec la requête précédente afin d'obtenir une clé d'utilisation. Exemple de client rest :



The screenshot shows a REST client interface. The 'Method' is set to 'POST'. The 'URL' is 'http://172.20.21.37/api'. The 'Body' is a JSON object: {"deviceType": "Hagot"}.

Ouvrez wireshark et vérifiez l'envoi de la requête HTTP et la réception de la réponse. Quelle est la durée de la transaction (entre l'envoi de la requête HTTP et la réponse du serveur) ? Pour faciliter la mesure, vous pouvez définir le premier élément comme étant la référence temporelle : menu éditer → Fixer le temps de référence.

Noter ici votre clé

## Interroger un élément

Quelle requête permet d'obtenir la liste des adresses mac de tous les éléments connectés à la passerelle ?

Exécuter la requête et noter les deux premières valeurs

Quelle requête permet d'obtenir les informations d'un élément selon son adresse mac ?

Exécuter la requête et noter le nom du fabricant du 3<sup>ème</sup> élément

Exécuter la requête permettant d'obtenir la liste des différentes lumières présentes. Donner le nombre d'élément présent

Constatez vous un ou plusieurs éléments particuliers ? Si oui pourquoi ?

Quel paramètre permet de faire la différence entre les éléments ?

Quel est sa valeur pour les lumières ?

Exécuter la requête permettant d'obtenir la liste des différents capteurs présents. Donner le nombre d'élément présent

Constatez vous un ou plusieurs éléments particuliers ? Si oui pourquoi ?

### Étude de la réponse JSON

Notepad++ dispose d'une extension permettant de rendre plus lisible les données JSON. Ouvrir Notepad++ -> ALT+P -> Plugin Manager -> JSON Viewer -> Install. Redémarrer Notepad++. Vous pouvez maintenant utiliser le raccourci : CTRL + ALT +SHIFT + M ou bien ALT+P -> Plugin Manager -> JSON Viewer -> Format JSON. Appliquer ce formatage à la liste des capteurs.

### Modifier un élément

Parmi les données JSON d'une lumière, quelle paramètre permet d'avoir son état ?

Donner la requête permettant d'éteindre la première prise de votre salle

Exécuter la requête précédente et vérifier le changement d'état.

Donner la requête permettant d'éteindre la lumière de votre salle

Exécuter la requête précédente et vérifier le changement d'état.



# MODULE 03

## SÉANCE SYSTÈME 02

### TP D'INFORMATIQUE

Durée 2h30

## CLIENT HTTP EN C++ : ENVOI REQUÊTE REST

### BLOC DE COMPÉTENCES

U6 - VALORISATION DE LA DONNÉE ET CYBERSÉCURITÉ

### COMPÉTENCE(S)

C08 - CODER

### OBJECTIFS PÉDAGOGIQUES

Analyse UML de l'installation. Codage d'un client HTTP permettant de demander la liste du matériel domotique.

### CONNAISSANCES ISSUES DU RÉFÉRENTIEL

- |  |          |
|--|----------|
| • Langages de modélisation                                       | Niveau 2 |
| • Programmation orientée objet                                   | Niveau 3 |
| • Programmation réseau   | Niveau 2 |
| • Outils logiciels d'évaluation, de traçabilité de l'information | Niveau 3 |

### CONNAISSANCES OPÉRATIONNALISÉES

- |  |          |
|--|----------|
| • Utiliser une classe C++                          | Niveau 3 |
| • Envoyer une trame HTTP                           | Niveau 3 |
| • Analyser une communication réseau avec Wireshark | Niveau 2 |
| • Versionner un code                               | Niveau 2 |

## TP

# Envoi de la requête GET permettant d'obtenir la liste des appareils

### Copie de la requête GET

En utilisant un client REST, envoyer à nouveau la requête GET permettant d'obtenir la liste des capteurs. Analyser les échanges sous Wireshark et copier la requête complète dans un bloc-note. Chaque fin de ligne doit être complétée par `\r\n`, les 4 caractères `\r\n\r\n` doivent terminer la trame HTTP.

La requête complète :

Quel filtre Wireshark avez-vous utiliser ?

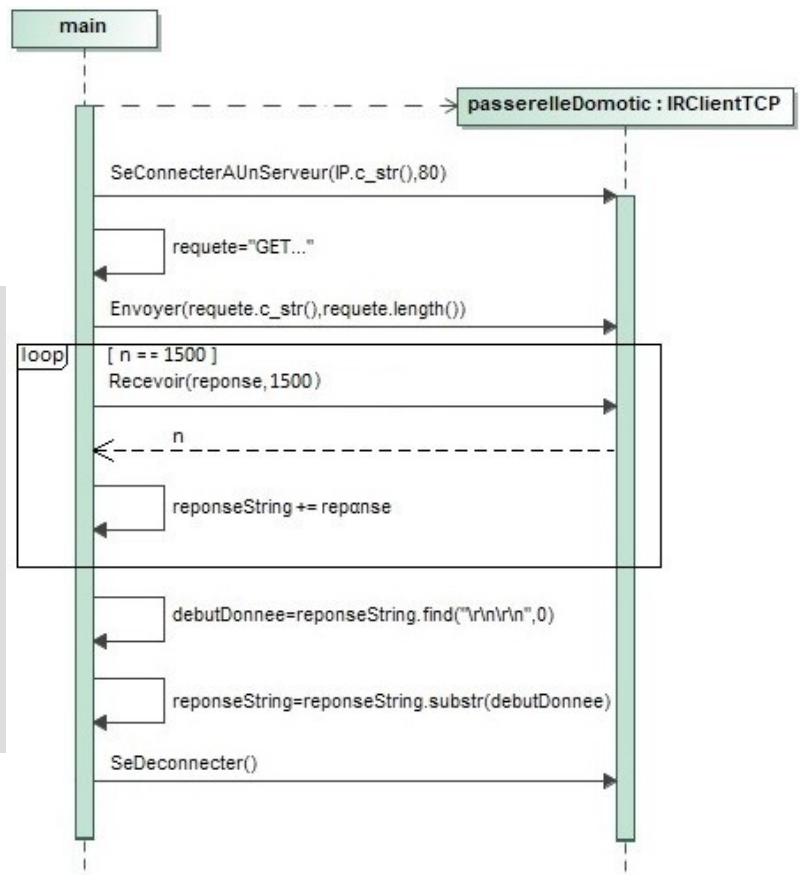
Quels sont les protocoles sur lesquels HTTP est basé ?

Copier de la même manière la liste des lampes.

## Codage du programme principal permettant d'envoyer la requête et d'afficher la réponse

Mettre en place le code correspondant au diagramme de séquence ci-contre.

Quel doit être la classe de l'objet requete, sachant que cette classe contient les méthodes `c_str()` et `length()` ? A quoi servent ces 2 méthodes ?



Donner la signification des arguments de la méthode `find`. Que renvoie cette méthode ? A quelle classe appartient-elle ?

Même question pour la méthode `substr` ?

Ajouter l'affichage de la réponse.  
Versionner le code complet.

# MODULE 03

## SÉANCE SYSTÈME 03

### TP D'INFORMATIQUE

Durée 2h30

## ENVOI D'UNE REQUÊTE GET EN C++ EXTRACTION DE DONNÉES JSON

### BLOC DE COMPÉTENCES

U6 - VALORISATION DE LA DONNÉE ET CYBERSÉCURITÉ

### COMPÉTENCE(S)

C08 - CODER

### OBJECTIFS PÉDAGOGIQUES

Codage de l'envoi d'une requête GET permettant de demander l'état d'un capteur de température. Extraction de données JSON : affichage de la température, de l'humidité et de la pression atmosphérique.

### CONNAISSANCES ISSUES DU RÉFÉRENTIEL

- |  |          |
|--|----------|
| • Langages de modélisation                                       | Niveau 2 |
| • Programmation orientée objet                                   | Niveau 3 |
| • Programmation réseau   | Niveau 2 |
| • Outils logiciels d'évaluation, de traçabilité de l'information | Niveau 3 |

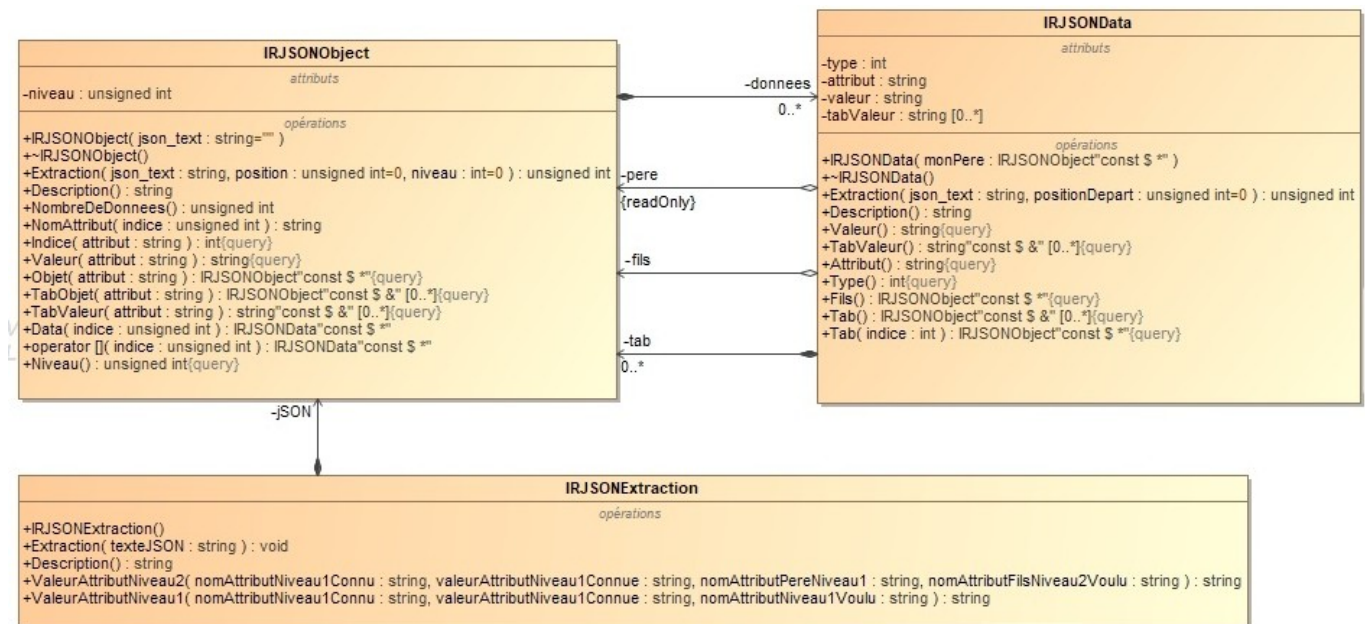
### CONNAISSANCES OPÉRATIONNALISÉES

- |  |          |
|--|----------|
| • Utiliser une classe C++                          | Niveau 3 |
| • Envoyer une trame GET en HTTP                    | Niveau 3 |
| • Analyser une communication réseau avec Wireshark | Niveau 2 |
| • Versionner un code                               | Niveau 2 |

## TP

## EXTRACTION DES MESURES D'UN CAPTEUR DE TEMPÉRATURE

## Extraction des données JSON : la classe IRJSONExtraction



Créer un objet JSON de la classe IRJSONExtraction dont le code est donné. Appeler la méthode Extraction, puis afficher la liste des capteurs à l'aide de la méthode Description. Le code :

## Extraction des données du capteur de température

Dans la donnée JSON correspondant à la liste des capteurs, à quoi correspond l'attribut de niveau 0 ?

Quelle est le nom de l'attribut de niveau 1 permettant d'identifier un capteur de température, quelle est sa valeur pour un des capteurs de température ?

Quel est le nom de l'attribut de niveau 2 correspondant à la température ?

Quel est le nom de son père de niveau 1 ?

Ajouter au code l'appel de la méthode ValeurAttributNiveau2 permettant d'afficher la température :

```
JSON.ValeurAttributNiveau2("name", "Temp A6-1", "state", "temperature");
```

Afficher maintenant l'humidité et la pression atmosphérique. Le code :

Afficher la valeur de l'attribut « type », en utilisant la méthode ValeurAttributNiveau1 :

```
JSON.ValeurAttributNiveau1("name", "Temp A6-1", "type");
```

Versionner le code complet.

# MODULE 03

## SÉANCE SYSTÈME 04

### TP D'INFORMATIQUE

Durée 2h30

## ENVOI D'UNE REQUÊTE PUT EN C++

### BLOC DE COMPÉTENCES

U6 - VALORISATION DE LA DONNÉE ET CYBERSÉCURITÉ

### COMPÉTENCE(S)

C08 - CODER

### OBJECTIFS PÉDAGOGIQUES

Codage de l'envoi d'une requête PUT permettant de modifier l'état d'une lampe multicolore. Mise en forme de la donnée au format JSON.

### CONNAISSANCES ISSUES DU RÉFÉRENTIEL

- |  |          |
|--|----------|
| • Langages de modélisation                                       | Niveau 2 |
| • Programmation orientée objet                                   | Niveau 3 |
| • Programmation réseau   | Niveau 2 |
| • Outils logiciels d'évaluation, de traçabilité de l'information | Niveau 3 |

### CONNAISSANCES OPÉRATIONNALISÉES

- |  |          |
|--|----------|
| • Utiliser une classe C++                          | Niveau 3 |
| • Envoyer une trame PUT en HTTP                    | Niveau 3 |
| • Analyser une communication réseau avec Wireshark | Niveau 2 |
| • Versionner un code                               | Niveau 2 |

## TP

# ENVOI DE LA REQUÊTE PUT PERMETTANT DE PILOTER UNE LAMPE

### Copie de la requête PUT

En utilisant un client REST, envoyer la requête PUT permettant d'allumer ou d'éteindre une lampe. Analyser les échanges sous Wireshark et copier la requête complète dans un bloc-note. Chaque fin de ligne doit être complétée par `\r\n`, les 4 caractères `\r\n\r\n` doivent terminer la trame HTTP.

La requête complète sans la donnée JSON :

La donnée JSON permettant d'allumer la lampe (elle sera placée à la suite de la requête HTTP) :

La donnée JSON permettant de modifier la couleur pour du rouge :

La donnée JSON permettant de fixer l'intensité maximum :



## Envoi de la requête PUT

En vous basant sur le code d'envoi de la requête GET, coder l'envoi de la requête PUT, la donnée JSON doit être placée à la suite de la requête (concaténation). Le code :

Ajouter une boucle permettant à l'utilisateur d'allumer la lampe en saisissant « true », ou bien de l'éteindre en saisissant « false ».

Ajouter le pilotage de la couleur et de l'intensité lumineuse.

Versionner le code complet.

# **MODULE 03**

## **SÉANCE SYSTÈME 05**

### **TP D'INFORMATIQUE**

**Durée 2h30**

## **CODAGE DES CLASSES CAPTEUR ET LAMPE EN C++**

### **BLOC DE COMPÉTENCES**

U6 - VALORISATION DE LA DONNÉE ET CYBERSÉCURITÉ

### **COMPÉTENCE(S)**

C08 - CODER

### **OBJECTIFS PÉDAGOGIQUES**

Codage d'une classe Capteur intégrant l'état d'un capteur et la génération de la requête GET. Codage d'une classe Lampe intégrant l'état de la lampe et la génération de la requête PUT.

### **CONNAISSANCES ISSUES DU RÉFÉRENTIEL**

- |  |          |
|--|----------|
| • Langages de modélisation d'application | Niveau 3 |
| • Programmation orientée objet           | Niveau 3 |
| • Programmation réseau                   | Niveau 3 |

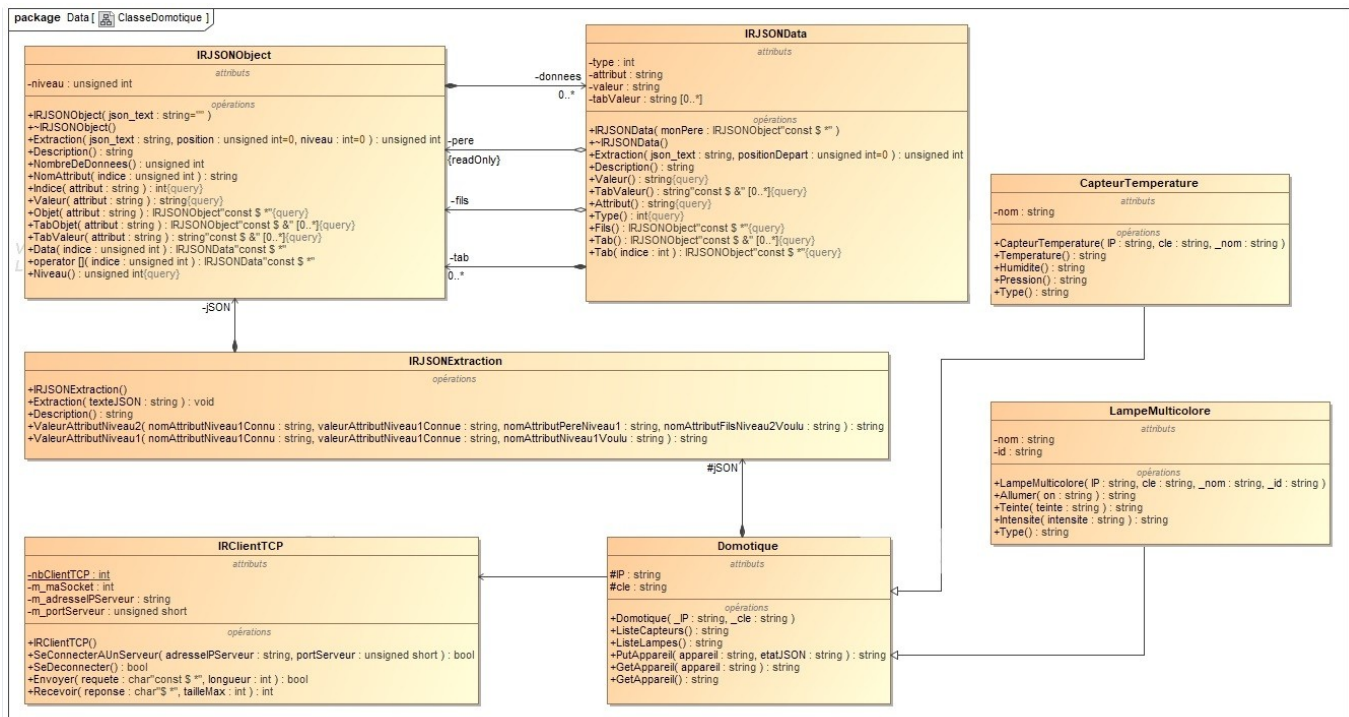
### **CONNAISSANCES OPÉRATIONNALISÉES**

- |  |          |
|--|----------|
| • Écrire une classe C++ à partir d'un diagramme de classes | Niveau 2 |
| • Versionner un code                                       | Niveau 2 |

## TP

## Codage de la classe Domotique

Le code doit maintenant être réorganiser sous forme de classes. La classe Domotique permet de gérer les requêtes REST quelque soit l'appareil. Cette classe se spécialise ensuite en 2 classes filles : CapteurTemperature et LampeMulicore. Voici le diagramme de classe complet :



Qu'est ce qu'une composition ? Dans le diagramme ci-dessus, donner un exemple de composition.

Qu'est ce qu'un héritage ? Dans le diagramme ci-dessus, donner un exemple d'héritage.

## Test de la classe Domotique

Créer un nouveau projet en mode console permettant de tester la classe Domotique donnée.

### Programme principal :

```
int main()
{
    Domotique domotique("172.20.21.4", "4FA1C71BD7");
    std::cout<<domotique.ListeCapteurs()<<std::endl<<std::endl;
    std::cout<<domotique.ListeLampes()<<std::endl<<std::endl;
    system("PAUSE");
    return 0;
}
```

### Le code de la déclaration de la classe :

```
#include "IRJSONExtraction.h"
#include "IRClientTCP.h"
class Domotique
{
protected:
    IRJSONExtraction jSON;
    string IP,cle;
public:
    Domotique(string _IP,string _cle);
    string ListeCapteurs();
    string ListeLampes();
    string PutAppareil(string appareil,string etatJSON);
    string GetAppareil(string appareil);
};
```

### Le code de la définition des méthodes :

```
#include <sstream>
#include "Domotique.h"
Domotique::Domotique(string _IP,string _cle)
{
    IP=_IP;
    cle=_cle;
}
std::string Domotique::ListeCapteurs()
{
    jSON.Extraction(GetAppareil("sensors"));
    return jSON.Description();
}
std::string Domotique::ListeLampes()
{
    jSON.Extraction(GetAppareil("lights"));
    return jSON.Description();
}
std::string Domotique::PutAppareil(std::string appareil,std::string etatJSON)
{
    IRClientTCP passerelleDomotic;
    std::string requete="PUT /api/"+cle+"/"+appareil+"/state HTTP/1.1"+"r\nHost: "+IP+"r\n
nConnection: keep-alive\r\nUser-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36
(KHTML, like Gecko) Chrome/117.0.0.0 Safari/537.36\r\nContent-Type: application/json\r\nAccept: */*\r\n\r\n";
    bool ok=passerelleDomotic.SeConnecterAUnServeur(IP,80);
    if(ok)
    {
        requete=requete+etatJSON;
        passerelleDomotic.Envoyer(requete.c_str(),requete.length());
        char reponse[1501];
        int n;
        n=passerelleDomotic.Recevoir(reponse,1500);
        reponse[n]=0;
        passerelleDomotic.SeDeconnecter();
        return reponse;
    }
    return "Serveur introuvable";
}
```

```

std::string Domotique::GetAppareil(std::string appareil)
{
    IRCClientTCP passerelleDomotic;
    bool ok=passerelleDomotic.SeConnecterAUnServeur(IP.c_str(),80);
    std::string requete="GET /api/"+cle+"/"+appareil+" HTTP/1.1"+"r\nHost: "+IP+"r\nConnection:
keep-alive\r\nUser-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like
Gecko) Chrome/117.0.0.0 Safari/537.36\r\nContent-Type: application/json\r\nAccept: */*\r\n\r\n";
    std::string reponseComplete="";
    if(ok)
    {
        passerelleDomotic.Envoyer(requete.c_str(),requete.length());
        char reponse[1501];
        int n,NbOctetsDonnees=0,contentLengthInt,debutDonnee;
        do
        {
            std::string rechercheDebut="Content-Length: ",rechercheFin="\r\n";
            n=passerelleDomotic.Recevoir(reponse,1500);
            reponse[n]=0;
            std::string reponseString=reponse;
            if(!NbOctetsDonnees)
            {
                int d=reponseString.find(rechercheDebut,0);
                int f=reponseString.find(rechercheFin,d);
                d=d+rechercheDebut.length();
                std::string contentLength=reponseString.substr(d,f-d);
                std::stringstream ss;
                ss<<contentLength;
                ss>>contentLengthInt;
                debutDonnee=reponseString.find("\r\n\r\n",f)+4;
                NbOctetsDonnees=n-debutDonnee;

                strcpy(reponse,reponseString.substr(debutDonnee,NbOctetsDonnees).c_str());
            }
            else NbOctetsDonnees+=n;
            reponseComplete=reponseComplete+reponse+"\n";
        }while(NbOctetsDonnees<contentLengthInt);
    } else return "Serveur introuvable";
    passerelleDomotic.SeDeconnecter();
    return reponseComplete;
}

```

## TP

### Codage des classes

#### CapteurTemperature et LampeMulticolore

Comment se code un héritage dans la déclaration d'une classe ?

Expliquer le passage d'arguments à la classe mère, dans la définition du constructeur de la classe CapteurTemperature :

```
CapteurTemperature::CapteurTemperature(string IP,string cle,string _nom):Domotique(IP,cle)
{
    nom=_nom;
}
```

Coder et tester les 2 classes. Exemple de programme principal :

```
int main()
{
    CapteurTemperature captTemperature("172.20.21.4","4FA1C71BD7","Temp A7-1");
    std::cout<<captTemperature.Type()<<std::endl;
    std::cout<<captTemperature.Temperature()<<std::endl;
    std::cout<<captTemperature.Humidite()<<std::endl;
    std::cout<<captTemperature.Pression()<<std::endl<<std::endl;
    LampeMulticolore lampe("172.20.21.4","4FA1C71BD7","Extended color light 3","3");
    std::cout<<lampe.Type()<<std::endl;
    while(1)
    {
        std::cout<<"teinte (0-65536), intensite (0-255), allumer (true false) : ";
        std::string teinte,intensite,allumer;
        std::cin>>teinte>>intensite>>allumer;
        lampe.Allumer(allumer);
        lampe.Teinte(teinte);
        lampe.Intensite(intensite);
    }
    return 0;
}
```

Pour la classe LampeMulticolore, on donne :

```
LampeMulticolore::LampeMulticolore(std::string IP, std::string cle, std::string _nom, std::string
_id):Domotique(IP, cle)
{
    nom=_nom;
    id=_id;
}
std::string LampeMulticolore::Allumer(std::string on) // "true" "false"
{
    return PutAppareil("lights/"+id, "{ \"on\": \"+on+\" }");
}
```

Recopier le code de la déclaration de votre classe LampeMulticolore :

Versionner le code complet.

# **MODULE 03**

## **SÉANCE SYSTÈME 06**

### **TP D'INFORMATIQUE**

**Durée 2h30**

## **INTERFACE GRAPHIQUE DOMOTIQUE EN C++**

### **BLOC DE COMPÉTENCES**

U6 - VALORISATION DE LA DONNÉE ET CYBERSÉCURITÉ

### **COMPÉTENCE(S)**

C08 - CODER

### **OBJECTIFS PÉDAGOGIQUES**

Codage d'une interface graphique permettant d'afficher l'état des capteurs et de piloter les lampes.

### **CONNAISSANCES ISSUES DU RÉFÉRENTIEL**

- Langages de développement, de description, de création d'API et les IDE associés Niveau 3
- Programmation orientée objet Niveau 3

### **CONNAISSANCES OPÉRATIONNALISÉES**

- Gestion des objets graphiques et des événements. Niveau 2
- Versionner un code Niveau 2

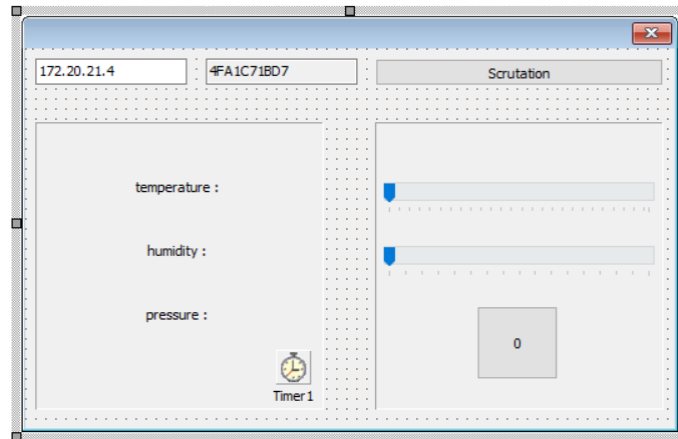


## TP

# Intégration dans une interface graphique des classes

## CapteurTemperature et LampeMulticolore

Créer un projet graphique sous C++Builder. Ajouter les éléments suivants :



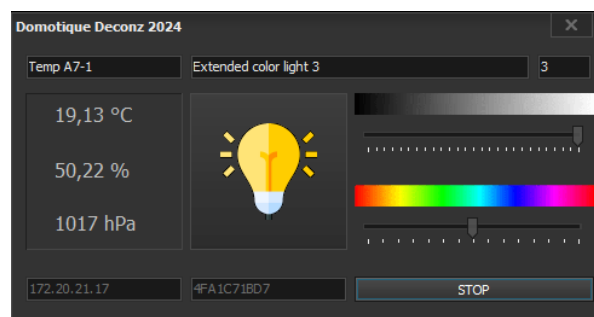
L'appui sur Scrutation lance le timer chargé d'appeler les méthodes de la classe CapteurTemperature. Le bouton « 0 » permet d'allumer la lampe, les 2 trackBars règlent l'intensité et la couleur. Les cadres sont des Bevels.

Coder entièrement l'interface et valider son fonctionnement.

Ajouter une image permettant de visualiser les couleurs de la teinte, puis une autre pour l'intensité :



Exemple d'interface finale :



Versionner le code complet.

# **MODULE 03**

## **SÉANCE SYSTÈME 07**

### **TP D'INFORMATIQUE**

**Durée 2h30**

## **ANDROID : UTILISATION DE LA CLASSE CAPTEUR**

### **BLOC DE COMPÉTENCES**

U5 - EXPLOITATION ET MAINTENANCE DE RÉSEAUX INFORMATIQUES

### **COMPÉTENCE(S)**

C06 - VALIDER UN SYSTÈME INFORMATIQUE

### **OBJECTIFS PÉDAGOGIQUES**

Installation Android Studio et test de déploiement sur tablette. Codage de l'application permettant d'afficher l'état des capteurs sur tablette.

### **CONNAISSANCES ISSUES DU RÉFÉRENTIEL**

- |  |          |
|--|----------|
| • Maîtrise des environnements de développement, d'intégration, de déploiement logiciel et des versions logicielles associées | Niveau 2 |
| • Langages de développement, de description, de création d'API et les IDE associés   | Niveau 3 |
| • Tests unitaires et d'intégration   | Niveau 3 |

### **CONNAISSANCES OPÉRATIONNALISÉES**

- |  |          |
|--|----------|
| • Installer un environnement de développement et de test pour appareil mobile sous Android | Niveau 2 |
| • Coder une application pour Android utilisant les sockets                                 |          |
| • Versionner un code   | Niveau 2 |

# Android - Utilisation de la classe Capteur

## Objectif de la séance

L'objectif de cette partie est de créer une application Android qui fonctionnera donc sur tous les smartphones et tablettes possédant l'OS Android. Cette application permettra l'affichage des données des capteurs (SS07) : la température, le taux d'humidité et la pression. Cette application réutilisera les classes que vous avez développées en C++ lors des précédentes séances.

## Création d'un projet sur Android Studio

Android Studio est un environnement de développement créé par Google pour créer des application pour l'OS Android. Cet environnement de développement utilise le Java et maintenant de Kotlin comme langage de développement. Il est cependant possible d'ajouter des fonctions natives en C++ avec le NDK (Native Development Kit) d'Android. C'est ce que nous ferons. Nous pourrons ainsi réutiliser du code que nous avons déjà écrit et testé.

Créer un nouveau projet Android en utilisant le NDK. Suivre le tuto vidéo fourni : [AndroidStudio-CréationDUnProjetNDK.mp4](#).

Dans le fichier AndroidManifest.xml, il faut ajouter au dessus de "<application", la ligne :

```
<uses-permission android:name="android.permission.INTERNET" />
```

## Test de compilation et d'installation de l'application sur un émulateur

Par défaut, lors de l'installation d'Android, un AVD a été installé. Un AVD est un Android Virtual Device, c'est-à-dire un appareil virtuel fonctionnant sous Android. L'appareil virtuel est un téléphone Pixel 3A.

Tester si votre nouvelle application se lance bien sur l'appareil virtuel

## Test de compilation et d'installation de l'application sur une tablette

Tester également si votre nouvelle application se lance bien sur la tablette

## Ajout des éléments dans l'IHM (Voir le tuto vidéo)

Voir le tuto : [AndroidStudio-PositionnementDesElementsDansLaFenetre.mp4](#)

- Ajout d'un Button et d'un TextEdit
  - Lien vers un tuto de mise en page : [Use ConstraintLayout to design your Android views](#)
  - Button :
    - id : buttonMAJ
    - text : Mise à jour des capteurs
  - TextEdit :
    - id : textEditTemperature
    - text : La température

En suivant le tuto vidéo, ajouter ces éléments dans l'IHM. Vérifier leur placement.

## Gestion du clic sur le Button et affichage de la température

Voir le tuto : *AndroidStudio-ImporterEtUtiliserDesClasses.mp4*

Ajout des classes dans le projet

Ajout des classes dans le répertoire cpp (.h et .cpp)

Ajout des fichiers sources (.cpp) dans le fichier CMakeLists.txt

- Ajout d'une nouvelle fonction native (tuto video)
  - Ajout de la fonction dans la classe MainActivity
    - `public native String RecupererTemperatureSalleA7();`
  - Ajout automatique de la fonction dans le fichier native-lib.cpp
  - Écriture du code de la fonction en C++ :
    - Création d'un objet `capTemp` de la classe `CapteurTemperature`
    - Récupération de la température dans un objet `strTemp` de la classe `string` en appelant la méthode `Temperature()` avec l'objet `capTemp`.
    - Renvoyer l'objet `strTemp`.
  - Appeler la fonction `RecupererTemperatureSalleA7()` lors du clic sur `buttonMAJ`, récupérer sa valeur de retour et la mettre dans le `textEditTemperature`.

En suivant le tuto vidéo, effectuer l'ensemble de ces étapes.

Faire de même pour les données de pression :

- Ajouter un nouveau `textEdit` dont l'id est `textEditPression`
- Ajouter une nouvelle fonction native : `public native String RecupererPressionSalleA7();`
- Ajouter automatiquement cette fonction dans le fichier `native-lib`
- Écrire le code en C++ de cette fonction
- Appeler la fonction `RecupererPressionSalleA7()` lors du clic sur `buttonMAJ`, récupérer sa valeur de retour et l'afficher dans le `textEditPression`

Faire de même pour les données d'humidité:

- Ajouter un nouveau `textEdit` dont l'id est `textEditHumidite`
- Ajouter une nouvelle fonction native : `public native String RecupererHumiditeSalleA7();`
- Ajouter automatiquement cette fonction dans le fichier `native-lib`
- Écrire le code en C++ de cette fonction
- Appeler la fonction `RecupererHumiditeSalleA7()` lors du clic sur `buttonMAJ`, récupérer sa valeur de retour et l'afficher dans le `textEditHumidite`

## En bonus, des pistes d'amélioration :

- Améliorer l'affichage des données :
  - Ajouter la virgule à la température,
  - Arrondir à l'unité la pression,
  - Ajouter les unités de chacune des mesures.
- Ajouter les capteurs de la salle A6
- Rafraîchir les données toutes les minutes
- Améliorer le design de l'application : un thermomètre, un baromètre, une jauge d'humidité...

# **MODULE 03**

## **SÉANCE SYSTÈME 08**

### **TP D'INFORMATIQUE**

**Durée 2h30**

## **ANDROID : UTILISATION DE LA CLASSE LAMPE**

### **BLOC DE COMPÉTENCES**

U6 - VALORISATION DE LA DONNÉE ET CYBERSÉCURITÉ

### **COMPÉTENCE(S)**

C08 - CODER

### **OBJECTIFS PÉDAGOGIQUES**

Codage d'une application Android permettant de piloter les lampes d'une installation domotique.

### **CONNAISSANCES ISSUES DU RÉFÉRENTIEL**

- Langages de développement, de description, de création d'API et les IDE associés Niveau 3
- Programmation orientée objet Niveau 3

### **CONNAISSANCES OPÉRATIONNALISÉES**

- Concevoir une interface graphique sous Android : gestion des événements java et communication en C++ natif Niveau 2
- Versionner un code Niveau 2

# Android : Utilisation de la classe LampeMulticolore

## Objectif de la séance

L'objectif de cette séance est d'ajouter à l'application Android la gestion des lumières multicolores : la gestion de l'allumage de la lampe, de la gestion des couleurs et de l'intensité.

## Gestion de l'allumage et de l'extinction de la lampe

Dans l'interface graphique (activity\_main.xml) :

Dans l'interface graphique (activity\_main.xml), ajouter 2 boutons dans votre interface : le 1<sup>er</sup> bouton : (id : buttonOn, text : ON) et le 2ème bouton (id:buttonOff, text : OFF)

Dans la classe d'interface MainActivity (MainActivity.java)

Dans la classe d'interface (MainActivity.java), gérer les clics sur ces deux boutons. Lors du clic sur le buttonOn, appeler la fonction AllumerLampeMulticoloreA6(). Lors du clic sur le buttonOff, appeler la fonction EteindreLampeMulticoloreA6().

Tout en bas de la classe d'interface (MainActivity.java), déclarer les 2 fonctions natives : AllumerLampeMulticoloreA6() et EteindreLampeMulticoloreA6().

Dans le fichier native-lib.cpp :

Écrire le code de ces 2 fonctions en C++ dans le fichier native-lib.cpp. Utiliser la classe LampeMulticolore comme dans votre projet en C++.

## Gestion de la brillance

Dans l'interface graphique (activity\_main.xml) :

Ajouter un champ de type texte (Plain Text) permettant de saisir la valeur de la brillance (entre 0 et 65535) : id : editTextBrillance, text : 30000.

Ajouter un bouton (id : buttonBrillance, text : Brillance) permettant d'appliquer cette valeur.

Dans la classe d'interface MainActivity (MainActivity.java)

Dans la classe d'interface MainActivity (MainActivity.java), ajouter la déclaration de la fonction native : ModifierBrillanceLampeMulticoloreA6(String valeur).

Dans le fichier native-lib.cpp :

Écrire le code de cette fonction en C++ dans le fichier native-lib.cpp. Utiliser la classe LampeMulticolore. Vous aurez également besoin de la remarque suivante :

Au début de cette fonction, il sera nécessaire de convertir la chaîne de caractères valeur de type jstring passée en argument en un objet de la classe string. C'est ce que fait l'instruction suivante :

```
string valeurBrillance = env->GetStringUTFChars(valeur, NULL);
```

Pour finir, dans la classe d'interface MainActivity (MainActivity.java) :

Gérer le clic sur le bouton `buttonBrillance` puis : Récupérer la valeur la brillance saisie dans le `editTextBrillance` et appeler la fonction native `ModifierBrillanceLampeMulticoloreA6()` en lui passant en argument la valeur saisie.

## Gestion de l'intensité

Dans l'interface graphique (`activity_main.xml`) :

Ajouter un champ de type texte (Plain Text) permettant de saisir la valeur de l'intensité (entre 0 et 255) : id : `editTextIntensite`, text : 255.

Ajouter un bouton (id : `buttonIntensite`, text : Intensité) permettant d'appliquer cette valeur.

Dans la classe d'interface `MainActivity` (`MainActivity.java`)

Dans la classe d'interface `MainActivity` (`MainActivity.java`), ajouter la déclaration de la fonction native : `ModifierIntensiteLampeMulticoloreA6(String valeur)`.

Dans le fichier `native-lib.cpp` :

Écrire le code cette fonction en C++ dans le fichier `native-lib.cpp`. Utiliser la classe `LampeMulticolore`. Vous aurez également besoin de la remarque suivante :

Au début de cette fonction, il sera nécessaire de convertir la chaîne de caractères `valeur` de type `jstring` passée en argument en un objet de la classe `string`. C'est ce que fait l'instruction suivante :

```
string valeurIntensite = env->GetStringUTFChars(valeur, NULL);
```

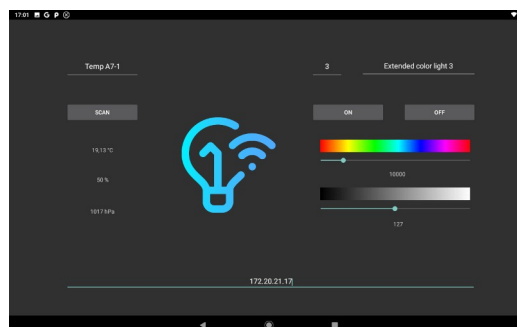
Pour finir, dans la classe d'interface `MainActivity` (`MainActivity.java`) :

Gérer le clic sur le bouton `buttonIntensite` puis : Récupérer la valeur l'intensité saisie dans le `editTextIntensite` et appeler la fonction native `ModifierIntensiteLampeMulticoloreA6()` en lui passant en argument la valeur saisie.

## Bonus

- Changer les champs de saisie texte par des slide bar.
- N'utiliser qu'un seul bouton pour allumer et éteindre la lumière (sa valeur change en fonction de l'état de la lumière)

Exemple d'interface finale :



# MODULE 03

## SÉANCE WEB 01

### TP D'INFORMATIQUE

Durée 2h30

## LA PAGE D'ACCUEIL

### UNITÉ CERTIFICATIVE

U6 - VALORISATION DE LA DONNÉE ET CYBERSÉCURITÉ

### COMPÉTENCE(S)

C08 - CODER

### OBJECTIF PÉDAGOGIQUE

Création de 1 pages HTML contenant la présentation de la domotique ainsi que la mise en page par du CSS

### CONNAISSANCES ISSUES DU RÉFÉRENTIEL

- Langages de développement, de description, de création d'API et les IDE associés Niveau 4
- Chaînes de développements (ordinateur, embarqué, cross compilation) Niveau 3

### CONNAISSANCES OPÉRATIONNALISÉES

- Codage d'une page HTML avec une feuille de style CSS Niveau 2



# TP

## Mise en place du versionning avec git

### Prérequis

Installation de Visual Studio Code

Installation de Git pour Windows

Installation de SSHFS-Win Manager pour Windows

Si ces outils ne sont pas installés sur votre poste de travail, les installer.

### Préparation des outils de travail pour le versionning

Connecter votre poste windows à votre serveur web en utilisant SSHFS-win Manager

Créer un répertoire nommé M03SW sur votre serveur web

Ouvrir ce répertoire avec Visual Studio Code (Clic droit, Ouvrir avec VS Code)

Créer un fichier README contenant le texte suivant :

```
Module 3 Web  
Gestion de modules domotique.
```

Aller sur l'outil git puis cliquer sur le bouton initialiser un dépôt pour créer un dépôt local (le répertoire (.git) permettra de gérer le versionning du projet. Par défaut, une branche « master » a été créée.

Ajouter le fichier README aux fichiers indexés.

Puis effectuer votre premier commit sur la branche master en validant tous les changements (ceci crée un snapshot de l'état actuel de votre projet). Vous pouvez préciser le texte suivant : « premier commit avec le fichier README ».

Créer une branche SW01

C'est dans la branche SW01 que nous travaillerons tout au long de cette séance.

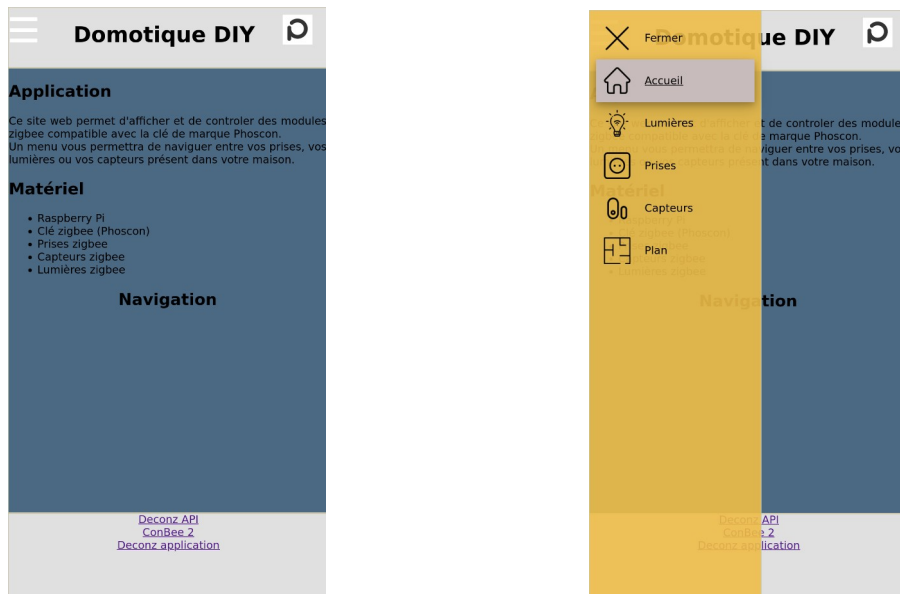
# Défi 1

## Création de la page de présentation

### Objectif

L'objectif de cette partie est d'écrire la page de présentation en HTML.

Le résultat final ressemblera à ceci :



### Première partie : le code HTML de la page d'accueil

Créer un fichier nommé index.html dans votre répertoire de travail.

Créer une structure de base d'un fichier HTML comme suit :

```

<!DOCTYPE html>
<html>
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Deconz</title>
</head>
<body>
  <nav id="nav">

  </nav>
  <header id="header">

  </header>

  <section id="section">

  </section>

  <footer id="footer">

```

```
</footer>
</body>
</html>
```

Compléter les différentes balises du code précédent afin de faire apparaître le contenu du résultat final. Chaque élément de la barre de navigation sera dans une balise « div » ayant une classe « nav », le text, quand à lui, sera dans une balise span.

Compléter les images du header avec des identifiants ayant comme valeur « imageHamburger » et « imageDomoticz ».

[GIT] Indexer le fichier index.html sur la branche SW01.

[GIT] Faire un commit avec le message : « Le code HTML de la page d'accueil »

## Deuxième partie : le code CSS de la page d'accueil

Créer le fichier style.css dans votre répertoire de travail

Compléter le fichier style.css en respectant, entre autres, les paramètres suivants

**body** : alignement du text central, largeur de « 100vw »

**header** : largeur de « 100% », une hauteur de « 10vh »

**images du header** : flottant gauche ou droite, marges extérieures de 10px

**nav** : hauteur de « 100vh », affichage désactivé, largeur de « 50% », position absolue, un « z-index » de 10

**section** : hauteur de « 75vh », une largeur de « 100% »

**footer** : hauteur de « 15vh », une largeur de « 100% », une position fixe, une valeur basse de 0

Vérifier la mise en page en faisant apparaître la barre de navigation en modifiant son affichage en « block »

[GIT] Indexer le fichier style.css sur la branche SW01.

[GIT] Faire un commit avec le message : « Le code HTML+CSS de la page d'accueil »

## Git de fin de séance : merger la branche SW01 sur la branche master

[GIT] Se placer sur la branche master et faire un « Merge Branch » de la branche SW01

La branche master contient alors maintenant le code de la branche SW01

[GIT] Publier la branche master sur GitHub

# **MODULE 03**

## **SÉANCE WEB 02**

### **TP D'INFORMATIQUE**

**Durée 2h30**

## **Initiation à l'AJAX**

### **UNITÉ CERTIFICATIVE**

U6 - VALORISATION DE LA DONNÉE ET CYBERSÉCURITÉ

### **COMPÉTENCE(S)**

C08 - CODER

### **OBJECTIF PÉDAGOGIQUE**

Comprendre et tester l'échange d'une requête HTTP par le javascript

### **CONNAISSANCES ISSUES DU RÉFÉRENTIEL**

- |  |          |
|--|----------|
| • Langages de développement, de description, de création d'API et les IDE associés | Niveau 4 |
| • Chaînes de développements (ordinateur, embarqué, cross compilation)              | Niveau 3 |
| • Outils de documentation  | Niveau 3 |

### **CONNAISSANCES OPÉRATIONNALISÉES**

## TP

# Gestion du hamburger

### Objectif :

L'objectif de cette partie est de comprendre la gestion des événements en JS. Un événement est déclenché généralement suite à une action de l'utilisateur : un clic sur un bouton, une saisie d'un texte, etc. Lorsqu'un événement est déclenché, il est possible d'appeler une fonction.

### Preparation GIT

[GIT] Créer la branche SW02

### Gestion des événements

En JS, la gestion d'un événement se fait en 2 étapes :

1. On récupère l'objet sur lequel l'utilisateur interagit
2. On indique quel événement de cet objet nous intéresse et la fonction qui sera exécutée.

### Un exemple

Supposons que nous souhaitons afficher un message popup lorsque la souris de l'utilisateur survole le footer de notre page. La première étape sera de récupérer l'objet (notre footer) avec la méthode `getElementById()` comme dans l'exemple suivant (`p_footer` est l'id du footer) :

```
var p_footer = document.getElementById("p_footer");
```

Ensuite, nous allons indiquer quel événement sur cet objet nous intéresse et quelle fonction sera appelée :

```
p_footer.addEventListener('mouseover', mon_popup);
```

Il reste maintenant à créer la fonction `mon_popup()` comme suit :

```
function mon_popup() {  
    alert("Gestion de l'évènement 'mouse over' sur mon footer");  
}
```

### Afficher le menu

Donner le code permettant de gérer l'évènement 'click' sur l'image du hamburger (son id est 'imageHamburger') et exécuter la fonction `AfficherMenu()`.

Créer un fichier fonction.js

Créer la fonction `AfficherMenu()`. Compléter le code de cette fonction qui changera le style de l'affichage de la barre de navigation en « block »

Ajouter le code gérant l'événement du click sur le hamburger après la fonction



En JS, la méthode `getElementById()` permet de récupérer un objet du corps de la page connaissant son identifiant. La méthode `addEventListener()` permet de gérer les événements d'un objet.



En JS, le mot-clé `function` permet de créer une nouvelle fonction avec un nom et si nécessaires des arguments. Ensuite, la définition de la fonction se fait entre les accolades ouvrante et fermante.

## Fermer le menu

Donner le code permettant de gérer l'évènement 'click' sur le texte « Fermer » du menu (son id est 'fermer') et exécuter la fonction `FermerMenu()`.

Créer la fonction `FermerMenu()`. Compléter le code de cette fonction qui changera le style de l'affichage de la barre de navigation en « none »

Ajouter le code gérant l'événement du click sur le texte « Fermer » après la fonction

[GIT] Indexer les fichiers sur la branche SW02.

[GIT] Faire un commit avec le message : « Gestion du hamburger »

## Préparation de la suite

Pour les données météorologiques, nous allons utiliser openweathermap

Se créer un compte sur openweathermap

Vérifier de bien avoir une clé « API Key » dans votre profil vous permettant de récupérer la météo. Cela peut prendre du temps alors passer à la suite en attendant.

## TD

# Comprendre l'AJAX

Dans cette partie nous verrons comment modifier une partie d'un site en utilisant le javascript et l'AJAX à partir des données chargées depuis le serveur au format JSON.

### Définitions

Que signifie l'acronyme AJAX ?

Quelle est la différence entre un processus synchrone et asynchrone ?

Citer 3 différents formats de données pouvant être échangé entre un serveur et le client

### Analyse de code

Voici un exemple de fonction AJAX en javascript permettant d'interroger le serveur :

```
function ajax() {  
  const xhttp = new XMLHttpRequest();  
  xhttp.onreadystatechange = function() {  
    if (this.readyState == 4 && this.status == 200) {  
      var reponse=this.responseText;  
      console.log(reponse);  
    }  
  };  
  xhttp.open(method, url);  
  xhttp.send();  
}
```

Quatre principales étapes sont nécessaires pour l'échange d'information :

1. Initialiser un objet XMLHttpRequest
2. Spécifier la méthode et l'URL ciblé
3. Envoyer la demande
4. Action à faire lorsque la réponse est reçue

Situer la ou les lignes de codes pour chaque étape précédente.

A quoi sert l'objet « readyState » ? Quelles sont les différentes valeurs possibles ? Donner leurs significations ?

A quoi sert l'objet « status » ? Donner la signification des valeurs 200, 403, 404.

Quelles sont les différentes valeurs possibles de la variable « method » ?



## TD

# Comprendre le JSON

### Structure JSON

Que signifie l'acronyme JSON ?

Par quel caractère doit commencer un structure de données JSON ?

Par quel caractère doit commencer un tableau de données JSON ?

Quels sont les différents type de valeurs que peut prendre un objet JSON ?

Donner un exemple de structure JSON

### Exploitation du JSON

Quelle fonction javascript permet de convertir une chaîne de caractère JSON en un objet JSON ?

Exemple de réponse JSON du serveur openweathermap à une requête du client :

```
{
```

```
{
  "coord": {
    "lon": 2.3488,
    "lat": 48.8534
  },
  "weather": [
    {
      "id": 801,
      "main": "Clouds",
      "description": "peu nuageux",
      "icon": "02d"
    }
  ],
  "base": "stations",
  "main": {
    "temp": 22.23,
    "feels_like": 22.34,
    "temp_min": 21.34,
    "temp_max": 23.32,
    "pressure": 1010,
    "humidity": 70
  },
  "visibility": 10000,
  "wind": {
    "speed": 4.63,
    "deg": 170
  },
  "clouds": {
    "all": 20
  },
  "dt": 1655974695,
  "sys": {
    "type": 2,
    "id": 2041230,
    "country": "FR",
    "sunrise": 1655956045,
    "sunset": 1656014283
  },
  "timezone": 7200,
  "id": 2988507,
  "name": "Paris",
  "cod": 200
}
```

La réponse précédente sera stockée dans une variable JS « reponseMeteo ». Pour pouvoir utiliser les données d'un format JSON, il faut les convertir en un objet javascript. Cette conversion sera stockée dans une variable « reponseMeteoObject ».

Donner le code permettant de convertir la chaîne de caractère JSON précédente en un objet JS « reponseMeteoObject »

En utilisant la variable reponseMeteoObject , Donner le code permettant de :

Connaître le nom de la ville

Connaître le pays

Connaître la température

Connaître la distance de visibilité

Quel est l'unité de la vitesse du vent ?

Quel est l'unité de la température ?

## TP

# Récupérer des données météorologiques

### Openweathermap API

Nous utiliserons dans ce module la version 2.5 de openweathermap

En vous aidant de la documentation (<https://openweathermap.org/current>), quelle requête HTTP faut-il envoyer pour récupérer la météo actuelle au format JSON, en français, aux unités métriques, de la ville de Paris.

Pour tester la requête HTTP, nous allons utiliser une extension d'un navigateur.

Ouvrez votre navigateur et installez une extension « client rest »

Tester la requête précédente dans un navigateur en y intégrant votre clé API.

Quel paramètre nous permet d'extraire la valeur de la température, la vitesse du vent, la visibilité et l'icône ?

Quelles sont les unités des valeurs précédentes ?

Quelle requête faut-il envoyer pour récupérer l'icône correspondante ?

## Index.html et style.css

Ajouter en bas de la barre de navigation une division avec les attributs id=meteo et class=meteo  
Dans la division précédente, il faut ajouter des « span » pour la ville, la température et l'icone  
Créer des attributs « id » et « class » pour les divisions précédentes (ville\_meteo, temperature, icone\_meteo). Nous utiliserons l'id avec le javascript et la classe pour la mise en page  
Modifier le design (style.css) pour avoir un affichage convenable.

## Javascript

Créer le fichier ajax.js  
Créer la fonction « RecupererMeteo » en y intégrant le code de base AJAX pour une requête HTTP  
Modifier dans la fonction précédente la méthode et l'URL afin de récupérer la météo actuelle de Paris  
Appeler la fonction « RecupererMeteo » lorsque l'on fait afficher la barre de navigation  
Vérifier le contenu de la réponse dans la console de votre navigateur

[GIT] Indexer les fichiers sur la branche SW02.

[GIT] Faire un commit avec le message : « Initialisation ajax»

## Git de fin de séance : merger la branche SW02 sur la branche master

[GIT] Se placer sur la branche master et faire un « Merge Branch » de la branche SW02

La branche master contient alors maintenant le code de la branche SW02

[GIT] Publier la branche master sur GitHub

# MODULE 03

## SÉANCE WEB 03

### TP D'INFORMATIQUE

Durée 2h30

## Exploitation des données météorologiques

### UNITÉ CERTIFICATIVE

U6 - VALORISATION DE LA DONNÉE ET CYBERSÉCURITÉ

### COMPÉTENCE(S)

C06 – VALIDER UN SYSTÈME INFORMATIQUES  
C09 – INSTALLER UN RÉSEAU INFORMATIQUE

### OBJECTIF PÉDAGOGIQUE

Analyser et exploiter des données d'une API openweathermap afin d'afficher la météo sur le site web.

### CONNAISSANCES ISSUES DU RÉFÉRENTIEL

- Langages de développement, de description, de création d'API et les IDE associés Niveau 4
- Chaînes de développements (ordinateur, embarqué, cross compilation) Niveau 3
- Outils de documentation Niveau 3

### CONNAISSANCES OPÉRATIONNALISÉES

- Analyse Wireshark d'une communication HTTP Niveau 2

# TD

## Extraction de données JSON

### Préparation GIT

[GIT] Créer la branche SW03

### Compréhension

Exemple de réponse JSON du serveur openweathermap à une requête du client :

```
{
  "coord": {
    "lon": 2.3488,
    "lat": 48.8534
  },
  "weather": [
    {
      "id": 801,
      "main": "Clouds",
      "description": "peu nuageux",
      "icon": "02d"
    }
  ],
  "base": "stations",
  "main": {
    "temp": 22.23,
    "feels_like": 22.34,
    "temp_min": 21.34,
    "temp_max": 23.32,
    "pressure": 1010,
    "humidity": 70
  },
  "visibility": 10000,
  "wind": {
    "speed": 4.63,
    "deg": 170
  },
  "clouds": {
    "all": 20
  },
  "dt": 1655974695,
  "sys": {
    "type": 2,
    "id": 2041230,
    "country": "FR",
    "sunrise": 1655956045,
    "sunset": 1656014283
  },
  "timezone": 7200,
  "id": 2988507,
  "name": "Paris",
  "cod": 200
}
```

Pour récupérer la longitude et la latitude, il faut alors utiliser le code suivant :

```
var latitude = reponseMeteoObject.coord.lat;
```

Écrire le code permettant de récupérer la longitude, la température, la vitesse du vent, la visibilité et le taux d'humidité ainsi que le nom de la ville.

Certaines valeurs peuvent être stockées dans un tableau d'objet. Ainsi pour la description de la météo, il faut utiliser le code suivant :

```
var description = reponseMeteoObject.weather[0].description;
```

Écrire le code permettant de récupérer l'icône de la météo

## Afficher les données du site

En vous aidant du TD précédent compléter la fonction ajaxMeteo() avec le code JS permettant d'extraire les données d'un format JSON et de les enregistrer dans les variables « temperature », « icone », « longitude » et « latitude » et « ville ».

Compléter le code en affichant les valeurs de la ville, la température,

Ajouter l'icône en modifiant l'image de fond de votre division « meteo »

Modifier le design de votre site si besoin

[GIT] Indexer les fichiers sur la branche SW03.

[GIT] Faire un commit avec le message : « Affichage des données météorologiques »

## Git de fin de séance : merger la branche SW02 sur la branche master

[GIT] Se placer sur la branche master et faire un « Merge Branch » de la branche SW03

La branche master contient alors maintenant le code de la branche SW03

[GIT] Publier la branche master sur GitHub



# MODULE 03

## SÉANCE WEB 04

### TP D'INFORMATIQUE

Durée 2h30

## Analyser l'API deconz

### BLOC DE COMPÉTENCES

U6 - VALORISATION DE LA DONNÉE ET CYBERSÉCURITÉ

### COMPÉTENCE(S)

C08 - CODER

### OBJECTIF PÉDAGOGIQUE

Etudier la documentation de l'API permettant d'interroger la passerelle zigbee utilisant le logiciel deconz.

### CONNAISSANCES ISSUES DU RÉFÉRENTIEL

- Langages de développement, de description, de création d'API et les IDE associés Niveau 4
- Chaînes de développements (ordinateur, embarqué, cross compilation) Niveau 3

### CONNAISSANCES OPÉRATIONNALISÉES

- Codage d'un formulaire HTML adapté à un code PHP donné Niveau 2
- Analyse d'un code PHP de récupération des input HTML Niveau 2

## TD

# Comprendre l'API REST

### Qu'est ce que le REST ?

Une API, ou interface de programme d'application, est un ensemble de règles qui définissent comment les applications ou les unités peuvent se connecter et communiquer entre eux. Une API REST est une API conforme aux principes de conception du style architectural REST ou Representational State Transfer. C'est pour cette raison que les API REST sont parfois appelées API RESTful .

Les API REST communiquent **via des requêtes HTTP** pour exécuter des fonctions de base de données standard telles que la **création, la lecture, la mise à jour et la suppression** d'enregistrements dans une ressource.

La représentation de la ressource peuvent être fournies à un client dans pratiquement n'importe quel format, y compris JavaScript Object Notation (JSON), HTML, XML, PHP ou un texte en clair

### Structure d'une requête



Figure 2: Structure d'une requête HTTP

Dans une requête HTTP, nous aurons plusieurs paramètres dont la méthode et l'URL qui nous intéressent ici. D'abord intervient la méthode puis la ressource via l'URL.

A quoi sert la méthode ?

A quoi sert la ressource (URL) ?

## Méthodes et ressource dans une requête

Les méthodes de requête HTTP permettent de définir l'action que l'on souhaite réaliser sur la ressource indiquée. Il en existe plusieurs mais nous allons nous focaliser sur les suivantes GET, POST, PUT, DELETE.

Définir dans quel cas chaque méthode est utilisée dans une API REST :

**GET** :

**POST** :

**PUT**

**DELETE**

Quand à elle, la ressource permet de définir la demande de l'utilisateur.

## TD

# L'API deconz

### Objectif

Comprendre l'API utilisée par deconz ([https://dresden-elektronik.github.io/deconz-rest-doc/about\\_rest/](https://dresden-elektronik.github.io/deconz-rest-doc/about_rest/))

### Préparation

Dans cette partie nous aurons besoin d'un client REST permettant de tester les requêtes HTTP.

| Installer une extension REST Client pour Firefox ou Postman ou Boomrang pour Chrome ou Edge

### Les bases de l'API

Selon le lien précédent, donner les informations suivantes :

| Quelle ressource faut ajouter à l'URL pour avoir des informations sur tous les lumières ?

GET      <http://172.20.21.230/api/<API Key>/lights>

| Quelle ressource faut ajouter à l'URL pour avoir des informations sur la lumière numéro 1 ?

GET      <http://172.20.21.230/api/<API Key>/lights/1>

| Quelle ressource faut ajouter à l'URL pour avoir des informations sur tous les capteurs ?

GET      <http://172.20.21.230/api/<API Key>/sensors>

| Que doivent faire les clients avant de pouvoir interroger le l'API ?

Avoir une clé

| Quels sont les avantages à utiliser l'API REST ?

Il est possible d'interroger une API REST depuis n'importe quelle machine et avec n'importe quel langage.

## Trouver l'adresse IP de la passerelle domotique

Pour scanner un réseau, vous pouvez utiliser le logiciel « Advanced IP Scanner »

- Installer le logiciel précédent sur votre poste
- Scanner le réseau 172.20.21.0-254/24
- Trouver le module ayant comme site web gatewayCIELIR

## Obtention d'une clé API

Donner les requêtes HTTP permettant de trouver l'adresse ip de la passerelle

GET <https://phoscon.de/discover>

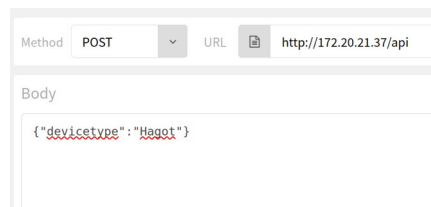
Donner les requêtes HTTP permettant récupérer une clé pour l'API en utilisant votre nom

POST <http://172.20.21.230/api>

```
{"devicetype" : "georges"}
```

Pour obtenir une clé de l'API, il faut demander à l'enseignant d'activer la demande de clé.

Ouvrir le client rest de votre navigateur, le compléter avec la requête précédente afin d'obtenir une clé d'utilisation. Exemple de client rest :



Ouvrez wireshark et vérifiez l'envoi de la requête HTTP et la réception de la réponse. Quelle est la durée de la transaction (entre le l'envoi de la requête HTTP et la réponse du serveur) ? Pour faciliter la mesure, vous pouvez définir le premier élément comme étant la référence temporelle : menu editor → Fixer le temps de référence

Noter ici votre clé

7FC997F590

## Interroger un élément

Quelle requête permet d'obtenir la liste des adresses mac de tous les éléments connectés à la passerelle ?

GET     /api/<apikey>/devices

Exécuter la requête et noter les deux premières valeurs

Quelle requête permet d'obtenir les informations d'un élément selon son adresse mac ?

GET     /api/<apikey>/devices/<device\_mac\_address>

Exécuter la requête et noter le nom du fabricant du 3<sup>ème</sup> élément

manufacturename : LUMI

Exécuter la requête permettant d'obtenir la liste des différentes lumières présentes. Donner le nombre d'élément présent

Constatez vous un ou plusieurs éléments particuliers ? Si oui pourquoi ?

Il y a des lumières et des interrupteurs commandés

Quel paramètre permet de faire la différence entre les éléments ?

"type"

Quel est sa valeur pour les lumières ?

"Extended color light"

Exécuter la requête permettant d'obtenir la liste des différents capteurs présents. Donner le nombre d'élément présent

Constatez vous un ou plusieurs éléments particuliers ? Si oui pourquoi ?

### Modifier un élément

Parmi les données JSON d'une lumière, quelle paramètre permet d'avoir son état ?

state

Donner la requête permettant d'éteindre la première prise de votre salle

Exécuter la requête précédente et vérifier le changement d'état.

Donner la requête permettant d'éteindre la lumière de votre salle

Exécuter la requête précédente et vérifier le changement d'état.

# **MODULE 03**

## **SÉANCE WEB 05**

### **TP D'INFORMATIQUE**

**Durée 2h30**

## **Gestion de lumières connectées**

### **UNITÉ CERTIFICATIVE**

U6 – VALORISATION DE LA DONNÉE ET CYBERSÉCURITÉ

### **COMPÉTENCE(S)**

C08 – CODER

### **OBJECTIF PÉDAGOGIQUE**

Coder la page web permettant de récupérer et de commander de lumières zigbee.

### **CONNAISSANCES ISSUES DU RÉFÉRENTIEL**

- Langages de développement, de description, de création d'API et les IDE associés Niveau 4
- Chaînes de développements (ordinateur, embarqué, cross compilation) Niveau 3

### **CONNAISSANCES OPÉRATIONNALISÉES**

- Mise en page d'une page web avec les grilles CSS. Niveau 2



## TD

# Analyser la réponse de l'API deconz

### Objectif

Décortiquer et comprendre le contenu de la réponse de l'API deconz des lumières

### Exemple de réponse

Voici un exemple de réponse à la requête

```
GET http://[ip]/api/[apikey]/lights
```

```
{
  "1": {
    "etag": "026bcfe544ad76c7534e5ca8ed39047c",
    "hascolor": true,
    "manufacturer": "dresden elektronik",
    "modelid": "FLS-PP3",
    "name": "Light 1",
    "pointsymbol": {},
    "state": {
      "alert": "none",
      "bri": 111,
      "colormode": "ct",
      "ct": 307,
      "effect": "none",
      "hue": 7998,
      "on": false,
      "reachable": true,
      "sat": 172,
      "xy": [ 0.421253, 0.39921 ]
    },
    "swversion": "020C.201000A0",
    "type": "Extended color light",
    "uniqueid": "00:21:2E:FF:FF:00:73:9F-0A"
  },
  "2": {
    "etag": "026bcfe544ad76c7534e5ca8ed39047c",
    "hascolor": false,
    "manufacturer": "dresden elektronik",
    "modelid": "FLS-PP3 White",
    "name": "Light 2",
    "pointsymbol": {},
    "state": {
      "alert": "none",
      "bri": 1,
      "effect": "none",
      "on": false,
      "reachable": true
    },
    "swversion": "020C.201000A0",
    "type": "Dimmable light",
    "uniqueid": "00:21:2E:FF:FF:00:73:9F-0B"
  }
}
```

```
}  
}
```

Chaque élément de notre maison connecté est répertorié, dans un format JSON, par un numéro : ici « 1 » et « 2 » car il y a deux éléments.

Supposons que la chaîne de caractères précédente soit stockée dans la variable « reponseLumiere ». Cette chaîne de caractère est converti en un objet javascript nommé « objetLumiere ».

Voici le code que nous utiliserons

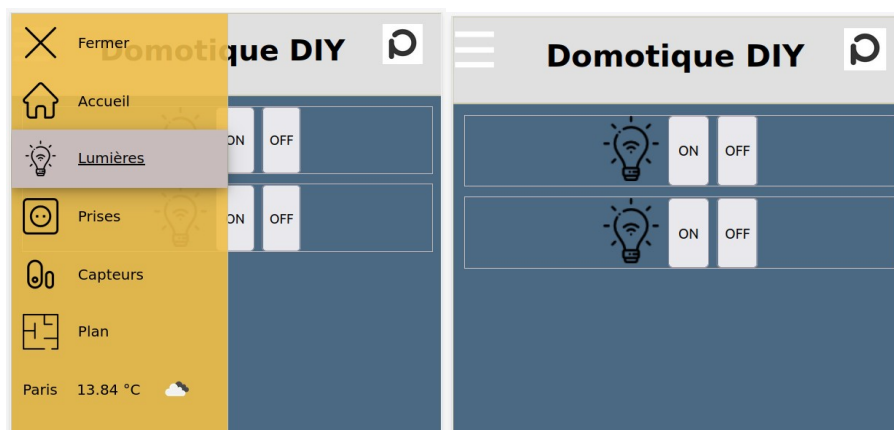
```
var lumiere="" ;  
for(num in objetLumiere){  
    let uniqueid=objetLumiere[num].uniqueid;  
    let etat=objetLumiere[num].state.on;  
    let type=objetLumiere[num].type;  
    lumiere=lumiere+'<div id="'+uniqueid+'" >';  
    lumiere+="</div>";  
}
```

Donner les valeurs de chaque variable en utilisant les données précédentes

## TP

# Afficher la liste des lumières

### Objectifs



### Préparation GIT

[GIT] Créer la branche SW05

### Menu de navigation

Ajouter un identifiant à chaque division de votre menu de navigation : lumiereNAV, capteurNAV, priseNAV, planNAV

Modifier le fichier fonction.js afin de rendre les liens du menu cliquable vers les fonctions : AfficherLumiere, AfficherPrise, AfficherCapteur

### Structure de base des lumières

Créer une fonction JS AfficherLumiere()

Ajouter le code AJAX de base en modifiant la méthode et la requête afin d'interroger les lumières  
Vérifier le bon fonctionnement en affichant la réponse dans la console du navigateur.

[GIT] Indexer tous les fichiers modifiés sur la branche SW05.

[GIT] Faire un commit avec le message : « Mise en place de la fonction AfficherLumiere »

### Afficher les lumières

La suite va nous permettre de compléter la fonction AfficherLumiere() avec les données de la domotique.  
La suite sera placée dans zone après la condition « if (this.readyState == 4 && this.status == 200) »

Créer une variable « section » contenant les informations de l'élément ayant comme identifiant « section »

Compléter avec l'initialisation de la variable lumière à une chaîne de caractère vide.

Ajouter le code permettant de faire une boucle sur les différents éléments (voir TD précédent)

Compléter la variable « lumière » avec la structure suivante :

```
lumiere= lumiere+'<button id="on">ON</button><button id="off">OFF</button>';
```

La variable précédente contient alors une structure HTML. Pour insérer un contenu HTML à partir d'un code JS nous utiliserons l'objet `innerHTML`.

Exemple de code :

```
document.getElementById('division').innerHTML='<p> Un paragraphe <p>';  
ou  
var param='<p> Un paragraphe <p>';  
document.getElementById('division').innerHTML=param;
```

Expliquer le code précédent ?

Donner le code permettant d'afficher le contenu de la variable « lumière » dans l'identifiant section de la page web

Compléter votre fichier javascript avec le code précédent

Modifier le code CSS si besoin

Pour n'afficher que les lumières, nous allons ajouter une condition sur la valeur de type.

Ajouter dans la boucle `for` la condition permettant de ne visualiser que les lumières

[GIT] Indexer tous les fichiers modifiés sur la branche SW05.

[GIT] Faire un commit avec le message : « Affichage des lumières »

## Interagir avec les lumières

Compléter les boutons « on » et « off » avec les attributs « onclick » ciblant les fonctions AllumerLumiere(num) et EteindreLumiere(num). Attention « num » est la variable de la boucle for à intégrer dans les balises des boutons.

Créer les fonctions précédentes

Les compléter avec le code de base AJAX en adaptant la méthode et l'URL afin d'allumer ou d'éteindre la lumière

[GIT] Indexer tous les fichiers modifiés sur la branche SW05.

[GIT] Faire un commit avec le message : «Interaction avec les lumières »

## Git de fin de séance : merger la branche SW05 sur la branche master

[GIT] Se placer sur la branche master et faire un « Merge Branch » de la branche SW05

La branche master contient maintenant le code de la branche SW05.

[GIT] Publier la branche master sur GitHub

# **MODULE 03**

## **SÉANCE WEB 06**

### **TP D'INFORMATIQUE**

**Durée 2h30**

## **Gestion de prises connectées**

### **UNITÉ CERTIFICATIVE**

U6 – VALORISATION DE LA DONNÉE ET CYBERSÉCURITÉ

### **COMPÉTENCE(S)**

C08 – CODER

### **OBJECTIF PÉDAGOGIQUE**

Contrôler des prises domotiques en interagissant avec une API REST

### **CONNAISSANCES ISSUES DU RÉFÉRENTIEL**

- Langages de développement, de description, de création d'API et les IDE associés
- Chaînes de développements (ordinateur, embarqué, cross compilation)

Niveau 4

Niveau 3

### **CONNAISSANCES OPÉRATIONNALISÉES**

.

## TP

# Afficher la liste des prises

### Objectifs

Créer un lien de menu permettant d'afficher la liste des prises

### Préparation GIT

[GIT] Créer la branche SW06

### Structure de base des prises

Créer une fonction JS AfficherPrise()

Ajouter le code AJAX de base en modifiant la méthode et la requête afin d'interroger les prises

Vérifier le bon fonctionnement en affichant la réponse dans la console du navigateur.

[GIT] Indexer tous les fichiers modifiés sur la branche SW06.

[GIT] Faire un commit avec le message : « Mise en place de la fonction AfficherPrise »

### Afficher les prises

La suite va nous permettre de compléter la fonction AfficherPrise() avec les données de la domotique. La suite sera placée dans zone après la condition « if (this.readyState == 4 && this.status == 200) »

Créer une variable « section » contenant les informations de l'élément ayant comme identifiant « section »

Compléter avec l'initialisation de la variable prise à une chaîne de caractère vide.

Ajouter le code permettant de faire une boucle sur les différents éléments (voir TD précédent)

Compléter la variable «prise» avec la structure suivante :

```
prise= prise+'<button id="on">ON</button><button id="off">OFF</button>';
```

Donner le code permettant d'afficher le contenu de la variable «prise» dans l'identifiant section de la page web

Compléter votre fichier javascript avec le code précédent

Modifier le code CSS si besoin

Pour n'afficher que les prise, nous allons ajouter une condition sur la valeur de type.

Ajouter dans la boucle for la condition permettant de ne visualiser que les prises

[GIT] Indexer tous les fichiers modifiés sur la branche SW06.

[GIT] Faire un commit avec le message : « Affichage des prises »

## Interagir avec les prises

Compléter les boutons « on » et « off » avec les attributs « onclick » ciblant les fonctions AllumerPrise(num) et EteindrePrise(num). Attention « num » est la variable de la boucle for à intégrer dans les balises des boutons.

Créer les fonctions précédentes

Les compléter avec le code de base AJAX en adaptant la méthode et l'URL afin d'allumer ou d'éteindre la prise

[GIT] Indexer tous les fichiers modifiés sur la branche SW06.

[GIT] Faire un commit avec le message : « Interaction avec les prises »

## Git de fin de séance : merger la branche SW06 sur la branche master

[GIT] Se placer sur la branche master et faire un « Merge Branch » de la branche SW06

La branche master contient maintenant le code de la branche SW06.

[GIT] Publier la branche master sur GitHub



# **MODULE 03**

## **SÉANCE WEB 07**

### **TP D'INFORMATIQUE**

**Durée 2h30**

## **Installation et configuration d'une Raspberry PI**

### **UNITÉ CERTIFICATIVE**

U5 - EXPLOITATION ET MAINTENANCE DE RÉSEAUX INFORMATIQUES

### **COMPÉTENCE(S)**

C09 - INSTALLER UN RÉSEAU INFORMATIQUE

### **OBJECTIF PÉDAGOGIQUE**

Installation de la Raspberry : Raspberry PI OS / Paramétrage réseau / Création des utilisateurs et gestions des droits sudo

### **CONNAISSANCES ISSUES DU RÉFÉRENTIEL**

- Systèmes d'exploitations (Windows, UNIX, Virtualisation) Niveau 3

### **CONNAISSANCES OPÉRATIONNALISÉES**

- Installer un OS Linux Niveau 2

## Préparation

### Objectif

Créer une passerelle Zigbee dans le cadre de la domotique.

### Pré-requis

La passerelle zigbee sera créée à l'aide d'une raspberry et d'une clé Conbee II de la marque Phoscon

A cela nous allons ajouter des objets connectés compatible avec le protocole zigbee

En voici la liste

- Capteur de température AQARA
- Capteur de présence AQARA
- Capteur d'ouverture de porte AQARA
- Smart+ plug Ledvance
- Smart+ Ledvance Gardenpole
- Raspberry pi

### Préparation

Rechercher et télécharger les documentations ou guides d'installation des éléments précédents

Rechercher la procédure d'installation de la clé Conbee II sur une raspberry

Faire un document résumant les étapes d'installation de la raspberry en précisant si besoin les logiciels utilisés ainsi que la procédure de réinitialisation des Gardenpole qui sera remise au client.

## Installation OS Deconz

Installer un l'OS Rasbian pour faire de la domotique en utilisant l'application Deconz de phosocon

### Installation de l'OS sur une carte SD

Suivez le tutoriel disponible sur la page dédiée du site officiel de la Raspberry pour installer le système d'exploitation : <https://www.phoscon.de/en/conbee2/sdcard>

Une fois la carte SD correctement installée, insérer la dans votre Raspberry. Ajouter un écran, une souris et un clavier, puis démarrer la Raspberry en l'alimentant.

### Configuration du réseau

Une carte réseau peut être configurée de 2 façons :

1. par une configuration automatique, c'est la configuration dynamique,
2. par une configuration manuelle, c'est la configuration statique.

Par défaut, la Raspberry est en mode dynamique.

Quelle est l'adresse IP de votre Raspberry ?

Modifier la carte réseau afin de fixer l'adresse IP à 172.20.21.1xx en remplaçant xx par la date du jour.

Modifier la passerelle, le masque et le DNS

### Test de premier démarrage

Par défaut les services SSH et VNC sont activé par défaut.

Qu'est ce que le SSH ?

Quels sont les identifiants et les mots de passe par défaut pour le SSH ?

A quoi sert le VNC?

Quels sont les identifiants et les mots de passe par défaut du VNC?

## Utilisation

Faire une mise à jour du système

Se connecter à distance la Raspberry en SSH et par VNC.

# MODULE 03

## SÉANCE WEB 08

### TP D'INFORMATIQUE

Durée 2h30

## Installation domotique

### UNITÉ CERTIFICATIVE

U6 – VALORISATION DE LA DONNÉE ET CYBERSÉCURITÉ

### COMPÉTENCE(S)

C08 – CODER

### OBJECTIF PÉDAGOGIQUE

Installation d'un système domotique

### CONNAISSANCES ISSUES DU RÉFÉRENTIEL

- Langages de développement, de description, de création d'API et les IDE associés
- Chaînes de développements (ordinateur, embarqué, cross compilation)

Niveau 4

Niveau 3

### CONNAISSANCES OPÉRATIONNALISÉES

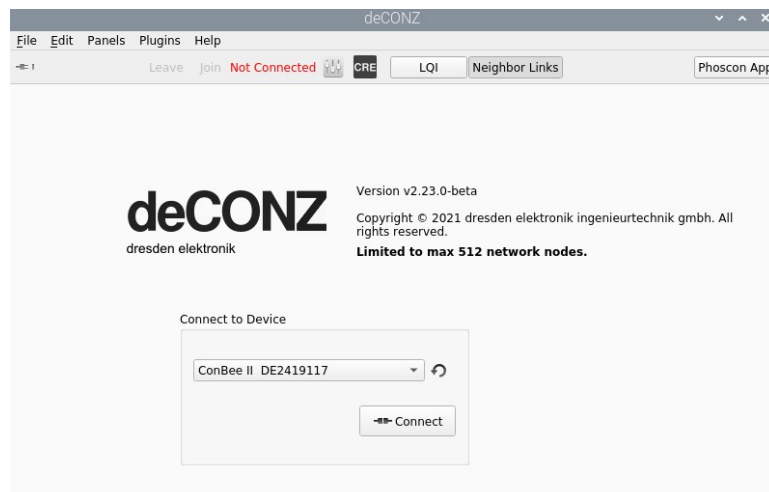
-

# Installation

Démarrer votre raspberry pi installer à la séance précédente avec l'application deconz

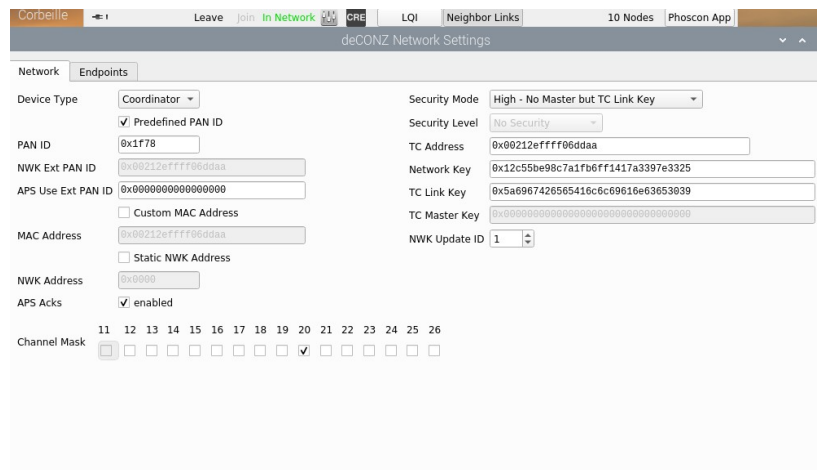
Brancher et connecter le dongle Conbee II de phoscon

Ouvrez l'application deconz



Connecter la clé ConBee II

Ouvrir le menu « Network settings » et modifier le canal 11 en 20



## Ajouter du matériel

### Pré-requis

Voici le matériel mise à votre dispositin :

- Capteur de température AQARA
- Capteur de présence AQARA
- Capteur d'ouverture de porte AQARA
- Smart+ plug Ledvance
- Smart+ Ledvance Gardenpole

### Mise en pratique

Ouvrez un navigateur et connecter vous à l'adresse IP de votre raspberry pi

Modifier le nom de la passerelle par : gatewayIR

Modifier le mot de passe par : gatewayIR

Aller dans les paramètres et remettre la langue en français

Aller dans l'onglet « plugs » et cliquer sur le bouton « Add new plug »

Procéder à la manipulation d'appariement de la prise

Aller dans l'onglet « sensors » et cliquer sur le bouton « Add new sensor»

Procéder à la manipulation d'appariement du capteur

Aller dans l'onglet « lights » et cliquer sur le bouton « Add new lights »

Procéder à la manipulation d'appariement de la lumière

Créer un premier groupe « Salon » et y intégrer les différents éléments : prise, capteur d'ouverture de porte et de température

Créer un premier groupe «Jardin» et y intégrer les différents éléments : lumière et capteur d'ouverture de mouvement

Créer un événement qui permet d'allumer les lumières lorsque le capteur de mouvement est activé

## Application android

Installer l'application Hue essentiel sur une tablette en téléchargeant l'APK sur internet

Configurer l'application pour la connecter à la passerelle Phoscon

Vérifier que vous êtes capable de contrôler les prises et les lumières