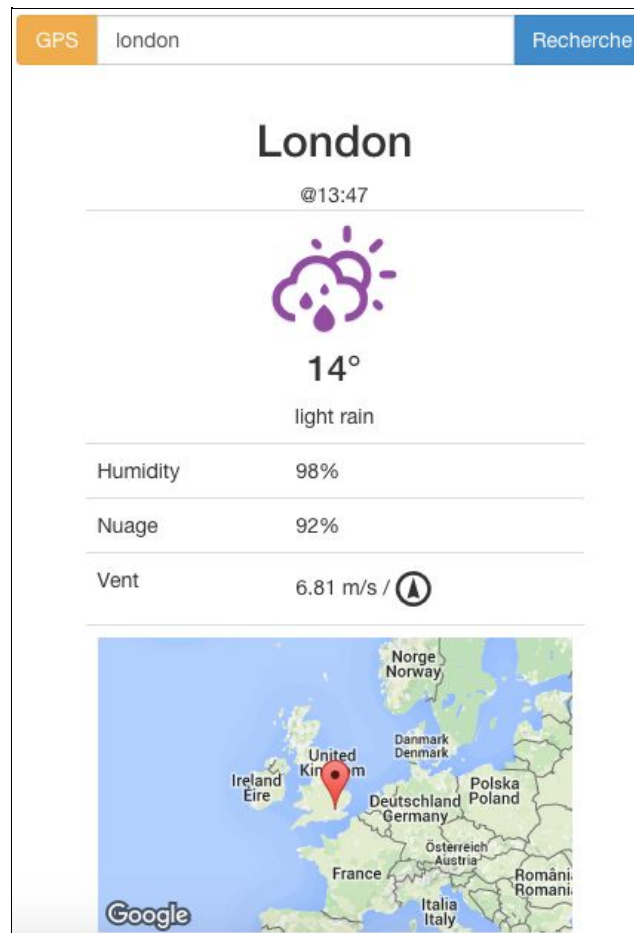




Master 3IR | 3ir2017.slack.com/messages/aw/

TP AW #5 : JS + Météo

Développement d'une application météo en JavaScript (vanilla) & [OpenWeatherMap](#).
[Vanilla JS](#) = Javascript sans aucun plugin (par exemple sans jQuery) permettant d'avoir une application web plus légère et donc plus rapide à charger.



Affichage de la météo pour Londres

Plan du TP

[OpenWeatherMap - générer votre API Key](#)

[Plate-forme de développement](#)

[HTML - Formulaire de recherche](#)

[JS - Gestion du formulaire](#)

[JS - Compléter la fonction searchCity pour faire un appel AJAX avec la ville en paramètre](#)

[JS - Compléter la fonction searchLatLng](#)

[Ajout de la geolocalisation HTML5](#)

OpenWeatherMap - générer votre API Key

> Créer un compte openweathermap sur <http://openweathermap.org/appid> et récupérer votre API Key :

Exemple de clé (Api Key/appid): 0ada432b59deb9716c357092c5f79be6

Exemple d'appel à l'API:

<http://api.openweathermap.org/data/2.5/weather?q=Paris&appid=0ada432b59deb9716c357092c5f79be6>

Analyser la structure de la réponse http://openweathermap.org/current#current_JSON

Documentation sur toute l'API : <http://openweathermap.org/current>

1. Plate-forme de développement

- Éditeur de texte ([visual studio code](#), notepad++, emacs, ...)
- Navigateur web (Mozilla Firefox, Google Chrome, Internet Explorer...)
- Librairie **Bootstrap**
 - <http://getbootstrap.com/getting-started/>
- Liste des imports à utiliser dans le <head>

```
<link rel="stylesheet" href="http://getbootstrap.com/dist/css/bootstrap.min.css">
<script src="meteo.js"></script>
```

/!\ Rappel : l'ordre d'importation des librairies JS est important

2. HTML - Formulaire de recherche

- Ecrire le code HTML pour le formulaire de saisie de la ville

```
<form id="searchCity">
  <input type="search" id="city" required placeholder="Votre ville, Paris, NY..." />
  <input type="submit" />
</form>
<div id="result"></div>
```

- Mettre en forme le rendu HTML du formulaire avec les classes CSS de Bootstrap

<http://getbootstrap.com/components/#input-groups-buttons-segmented>

3. JS - Gestion du formulaire

Prendre en compte le code en rouge qui est absent de la version papier du TP

```
window.onload = function(){
  document.getElementById("searchCity").addEventListener("submit", function(event){
    event.preventDefault(); // pour annuler le rechargement de la page
    var city = document.getElementById("city").value;
    searchCity(city);
  });
};
```

```

}

function searchCity(_city){
  console.log('searchCity','Hello from '+_city);
  //A compléter dans la suite du TP
}

function searchLatLng(_lat, _lng){
  console.log(searchLatLng,'Hello from '+_lat+', '+lng);
  //A compléter dans la suite du TP
}

```

meteo.js

□ : “window.onload” permet de s’assurer que tout le document HTML(DOM) est déjà téléchargé par le navigateur web avant d’exécuter du code JS.

4. JS - Compléter la fonction searchCity pour faire un appel AJAX avec la ville en paramètre

http://www.w3schools.com/xml/ajax_intro.asp

<http://youmightnotneedjquery.com/#request>

```

function searchCity(_city){
  var request = new XMLHttpRequest();
  request.open('GET',
    'http://api.openweathermap.org/data/2.5/weather?q='+_city+'&appid=0ada432b59deb9716c357092c5f79be6', true);

  request.onload = function() {
    if (request.status >= 200 && request.status < 400) {
      // Success!
      var resp = request.responseText;

      //VOTRE CODE JS pour afficher les données météo sur votre page
      // en mettant à jour la DIV “result”
    } else {
      // We reached our target server, but it returned an error
    }
  };
  request.onerror = function() {
    // There was a connection error of some sort
  };
  request.send();
}

```

Appel d’un webservice en JS

5. Icône pour l’illustration météo

Utiliser les icônes fournies par openWeatherMap

exemple : pour le code 02d
<http://openweathermap.org/img/w/02d.png>



6. JS - Compléter la fonction searchLatLng

// API openweather avec lat/lng : <http://openweathermap.org/current#geo>

7. Ajout de la géolocalisation HTML5

L'API Géolocalisation HTML5 est utilisée pour obtenir la position géographique d'un utilisateur (si il utilise un navigateur récent)

Ajouter un bouton à coté du champ de saisie de la ville

- a. en Javascript, copier ce code pour **intercepter le click sur le bouton GPS** et **écrire le code pour demander la géolocalisation** à l'utilisateur

```
window.onload = function(){  
  //[... Code JS précédent ...]  
  
  document.getElementById("gps").addEventListener("click", function(){  
    // ici votre code pour demander la géolocalisation à l'utilisateur  
    // et qui appellera la fonction searchLatLng(_lat, _lng)  
  });  
  
}
```

code pour intercepter le click sur le bouton GPS

- i. reprendre le code JS sur cette exemple :
http://www.w3schools.com/html/tryit.asp?filename=tryhtml5_geolocalon_map
- ii. documentation et fonction JS de géolocalisation disponibles ici :
http://www.w3schools.com/html/html5_geolocation.asp
- b. La géolocalisation vous donnera la **latitude** et la **longitude** de l'utilisateur
 - i. Sous les données météo, afficher une image de Google Maps centrée sur ces coordonnées GPS
<http://maps.googleapis.com/maps/api/staticmap?markers=latitude,longitude&size=640x400&zoom=5>

Exemple avec New-York (latitude : 40.714728, longitude : -73.998672)



<http://maps.googleapis.com/maps/api/staticmap?markers=40.714728,-73.998672&size=640x400&zoom=5>