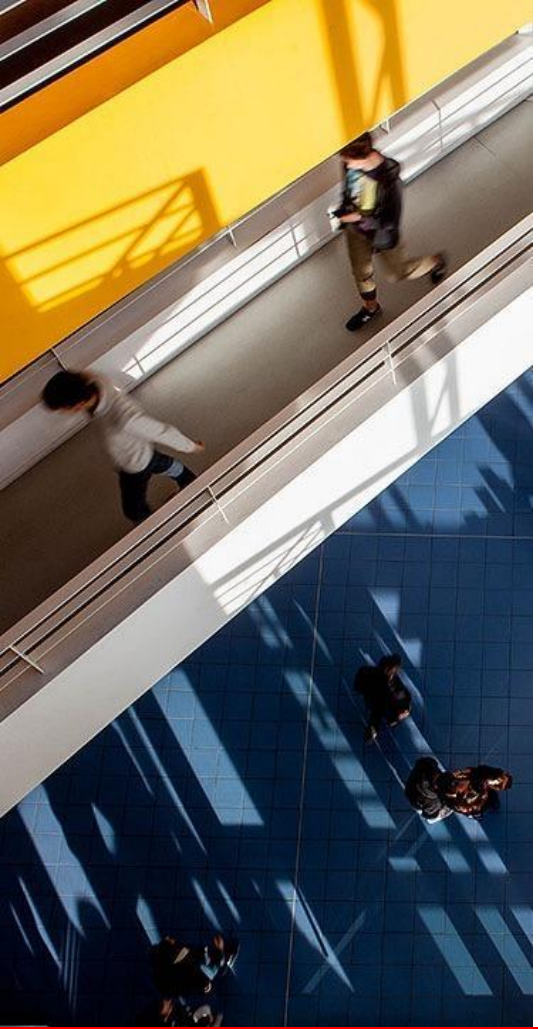


Communication Web
- AJAX -

Groupe des étudiants : CIR2

Ayoub KARINE
Numéro de bureau : A3-78

Définition

- AJAX (Asynchronous JAvascript and Xml) est une architecture permettant de construire des sites internet ou des applications Web interactives et dynamiques.

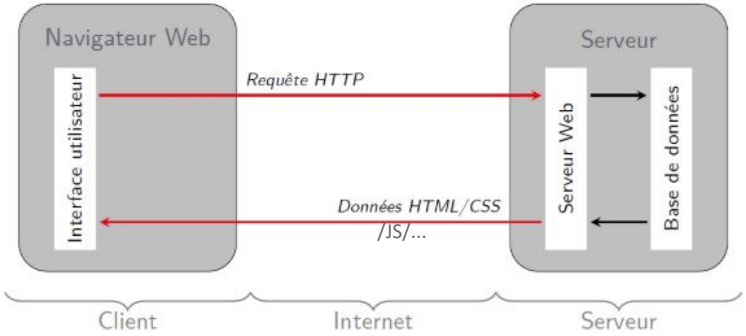
- AJAX se base principalement sur :
  - JavaScript
  - l'objet XMLHttpRequest
  - L'interface du DOM (Document Object Model)

- AJAX permet de réaliser des appels asynchrones depuis un client vers un service web

- **AJAX évite le rechargement de la page en insérant directement la réponse (XML ou texte) dans le DOM de la page**.

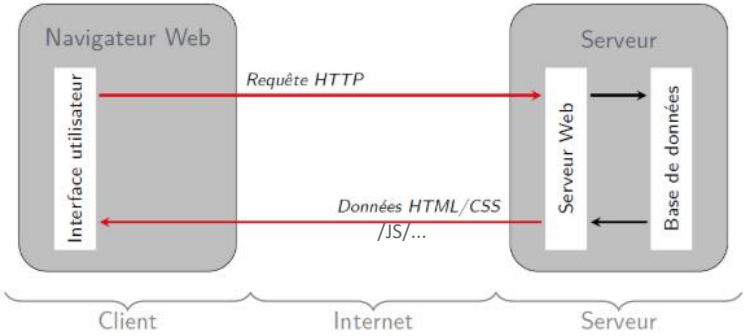
SANS AJAX

Première communication :

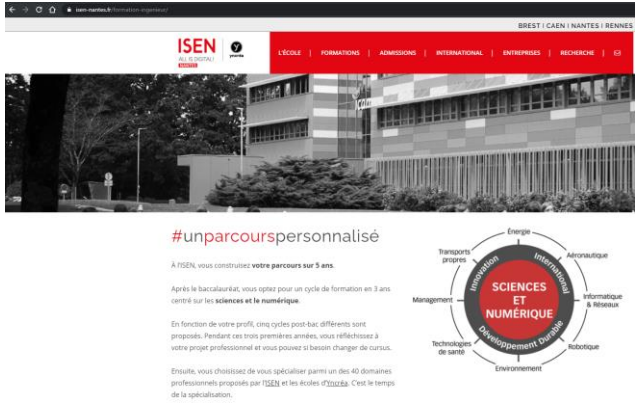


SANS AJAX

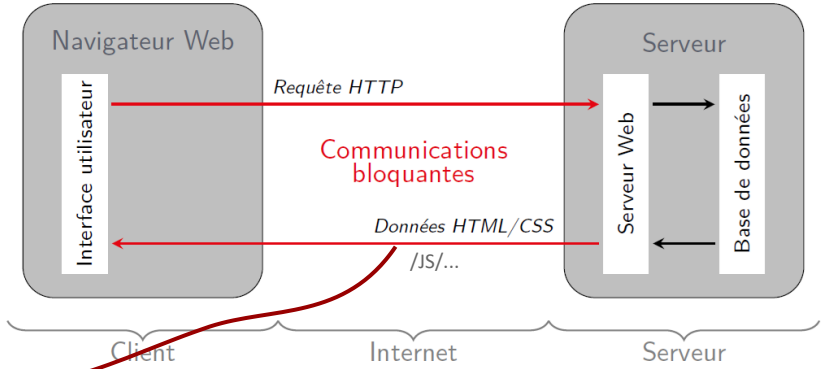
Première communication :



Deuxième communication :

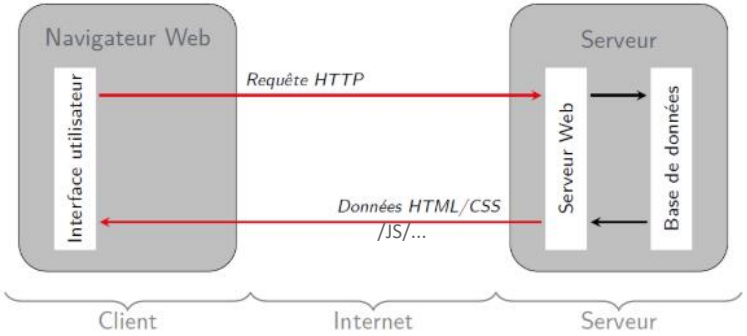


Le modèle de communication standard :



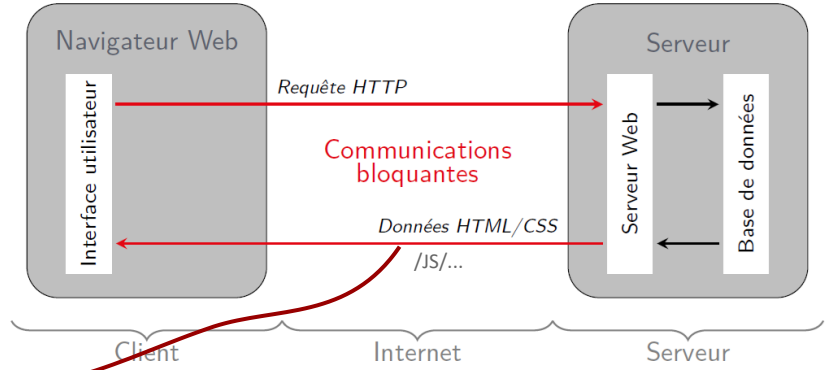
SANS AJAX

Première communication :



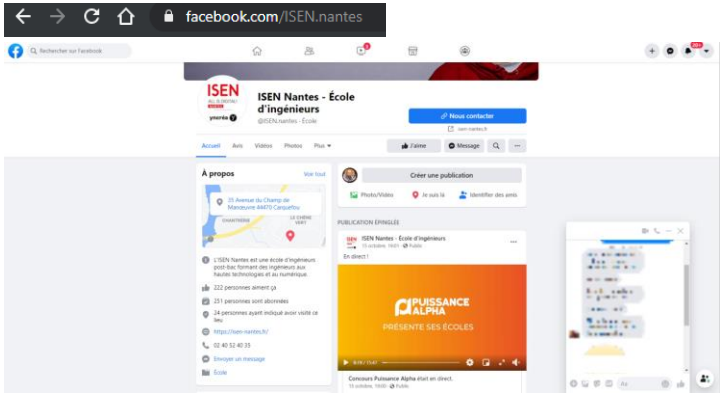
Deuxième communication :

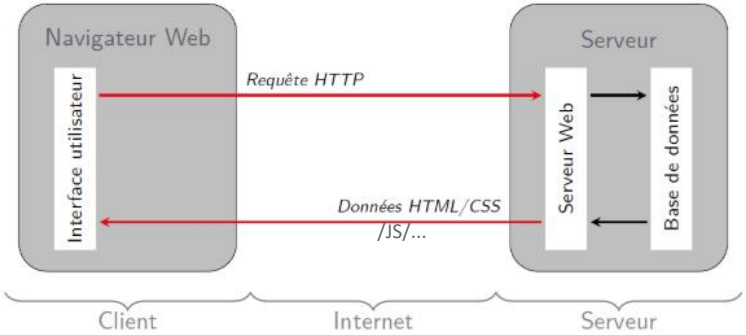
Le modèle de communication standard :



AVEC AJAX

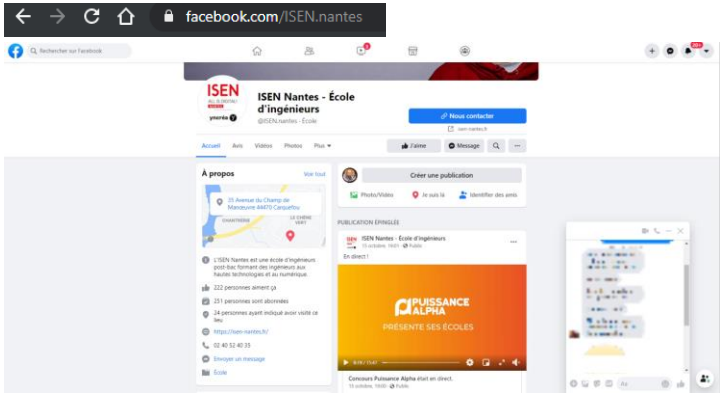
Première communication :

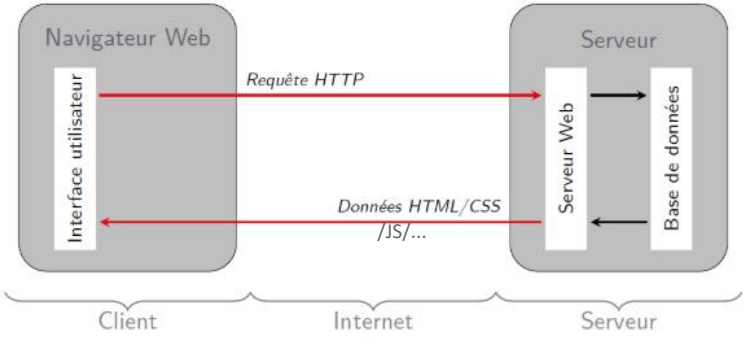




AVEC AJAX

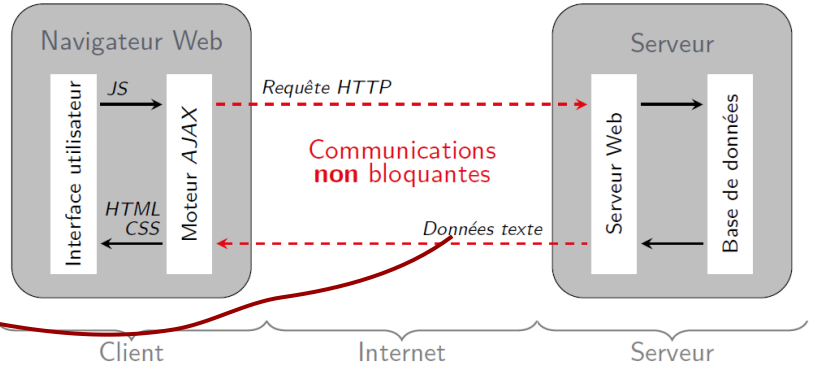
Première communication :





Deuxième communication :

Le modèle de communication avec ***AJAX*** :

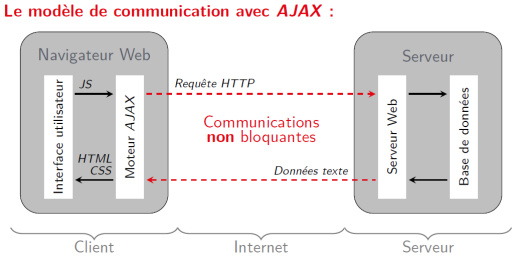


AJAX en JavaScript : XMLHttpRequest

- L'objet XMLHttpRequest permet de faire une requête AJAX en JavaScript. Il existe dans tous les navigateurs et permet de faire des requêtes synchrones (déprécié) et asynchrones.

- L'objet XMLHttpRequest possède des méthodes et des propriétés :
  - **Méthodes** : envoi des requêtes du client vers les serveurs
  - **Propriétés** : traitement de la réponse envoyée par le serveur suite à une requête.

AJAX en JavaScript : XMLHttpRequest



- **Méthodes de l'objet XMLHttpRequest**:

  - open(type, url, [async]) : initialisation de la requête avec :
    - **type** : GET, POST, PUT ou DELETE (type de requête)
    - **url** : indique l'url du serveur où envoyer la requête
    - **async** : true pour asynchrone (par défaut), false pour synchrone

  - send([data]) : envoi de la requête HTTP :
    - **data** : données à envoyer avec la requête (optionnel)

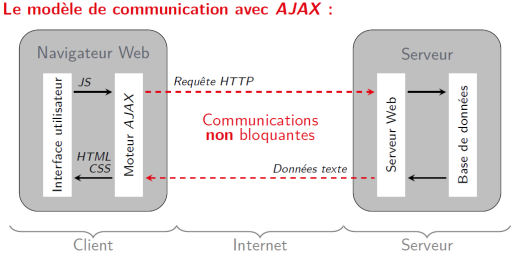
  - setRequestHeader(...) : spécification des en-têtes à envoyer au serveur en même temps que la requête (authentification, type de données...) ;

  - abort() : arrêt la requête en cours

AJAX en JavaScript : XMLHttpRequest

- **Propriétés de l'objet XMLHttpRequest**:

Le modèle de communication avec *AJAX* :

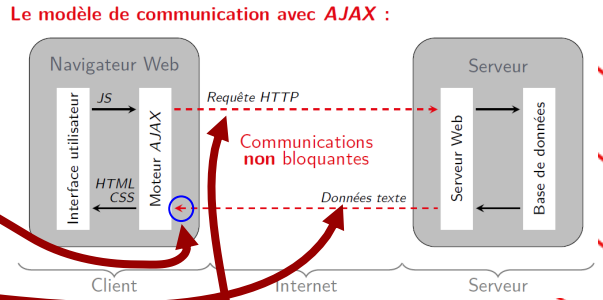


- responseText : réponse du serveur en format texte ;
- responseXML : réponse du serveur en XML;
- status : statut de la requête HTTP (voir ce lien) ;
- onload : gestionnaire d'événements appelé lorsque la réponse du serveur est reçue ;
- readyState : état de la requête :
  1 : loading
  2 : loaded
  3 : interactive
  4 : complete

AJAX en JavaScript : XMLHttpRequest

- **Exemple :**

```
1  // Create XML HTTP request.
2  let xhr = new XMLHttpRequest();
3  xhr.open('GET', 'http://server.com/request.php');
4
5  // Add onload function.
6  xhr.onload = () => {
7    switch (xhr.status) {
8      case 200:
9      case 201: console.log(xhr.responseText);
10       break;
11     default: console.log('HTTP error:' + xhr.status);
12   }
13 };
14
15 // Send XML HTTP request.
16 xhr.send();
```



Gestion des réponses
Cas des données brutes

- La méthode la plus utilisée pour intégrer la réponse du serveur dans une page est son intégration, directe ou après traitement, dans une du document HTML.

```
document.getElementById('myDivId').innerHTML =
  xhr.responseText;
```

- Les données peuvent être actualisées par une requête asynchrone de deux façons différentes :
  - Suite à une action de l'utilisateur : clic ou mouseover par exemple
  - Automatiquement, par une actualisation à intervalles réguliers

```
setInterval(func, delay, arg0, arg1, /* … ,*/ argN)
```

Pour supprimer cette intervalle on utilise la fonction :

```
clearInterval(intervalRef)
```

Gestion des réponses
Cas des données JSON

- JSON (JavaScript Object Notation – Notation Objet issue de JavaScript) est un format léger d'échange de données entre deux entités ne partageant pas le même langage. Il s'agit d'un format de données textuelles basé sur la notation des objets en JavaScript*.

- Les types de données en JSON sont :
  - Chaînes de caractères
  - Nombre
  - Booléen
  - Null

- Types d'élements structurés :
  - Tableau

```
[valeur1, valeur2, ...]
```

  - Dictionnaire

```
{"cle1": valeur1, "cle2": valeur2, ...}
```

* Pourtant, c'est une notation indépendante de tout langage de programmation

Gestion des réponses

Cas des données JSON

ISEN ALL IS DIGITAL! OUEST | yncréa

Exemple de données JSON

```json
{
  "Produits": [{
    "Nom": "Verre",
    "Prix": 9.99,
    "Quantité": 23,
    "En stock": true,
    "Couleurs": ["Vert", "Rouge", "Bleu"],
    "Marque": "Ikea"
  }, {
    "Nom": "Assiette",
    "Prix": 12.99,
    "Quantité": 0,
    "En stock": false,
    "Couleurs": ["Blanc"],
    "Marque": null
  }],
  "Magasin": "www.vaisselle.fr"
}
```

Gestion des réponses
Cas des données JSON

- **Encodage et décodage d'un fichier JSON en PHP**
  - Encodage (json_encode) :

```php
$data = array(1, true, "isen", array("fruit" => "kiwi",
  "nb" => 2));
$json = json_encode($data);
print_r($json); // [1, true, "isen", {"fruit": "kiwi",
                // "nb": 2}]
```

Tableau

↓

JSON

  - Décodage (json_decode) :

```php
$json = '[1, true, "isen", {"fruit": "kiwi", "nb": 2}]';
$data = json_decode($json);
print_r($data); // Array([0] => 1 [1] => 1 [2] => isen
                // [3] => Array([fruit => kiwi [nb] => 2]))
```

JSON

↓

Tableau

Gestion des réponses
Cas des données JSON

- **Encodage et décodage d'un fichier JSON en JS**
  - Encodage (JSON.stringify) :

```
var data = [1, true, 'isen', {'fruit': 'kiwi', 'nb': 2}];
var json = JSON.stringify(data);
console.log(json); // [1, true, 'isen', {'fruit: 'kiwi',
                   // 'nb': 2}]
```

  - Décodage (JSON.parse) :

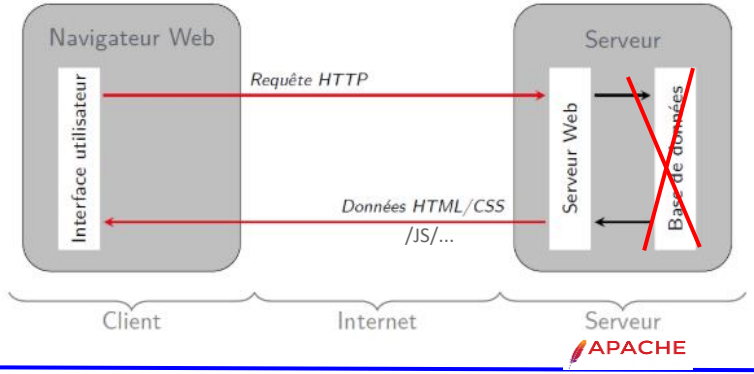
```
var json = '[1, true, 'isen', {'fruit': 'kiwi', 'nb': 2}]';
var data = JSON.parse(json);
console.log(data); // [0:1, 1:true, 2:'isen',
                   // 3:{fruit:'kiwi', nombre:2}]
```

TP1 - ex1-

Première communication :

AJAX

Date et heure



Deuxième communication :

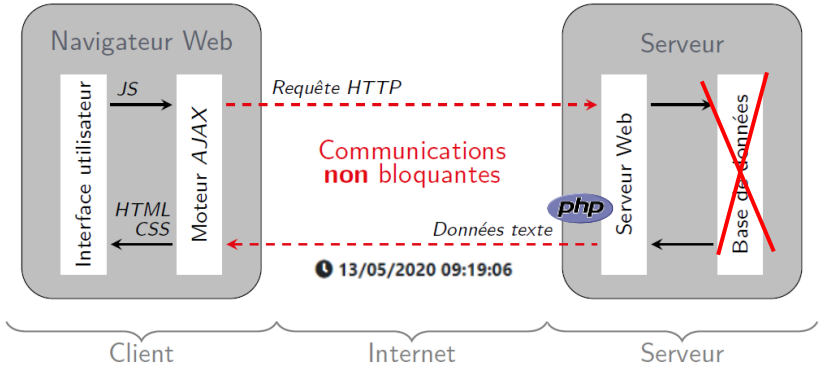
AJAX

Date et heure

13/05/2020 09:19:06

Actualisation chaque seconde

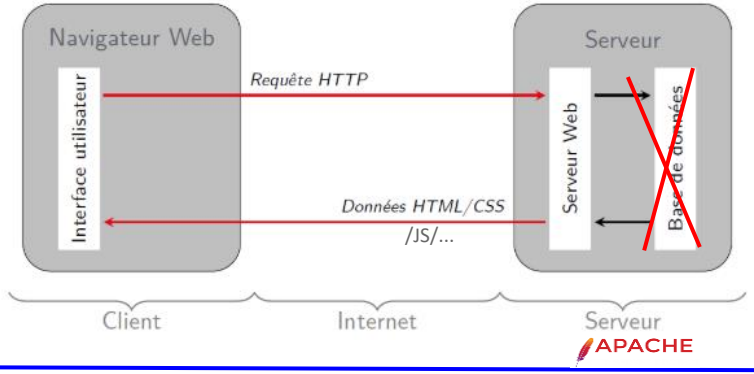
Le modèle de communication avec *AJAX* :



TP1 - ex2-

Première communication :

AJAX-JSON



Deuxième communication :

AJAX-JSON

Actualisation chaque seconde

Le modèle de communication avec *AJAX* :