Chapitre : Introduction Générale aux Technologies Web

- Fondamentaux du Web
- Composants du Web
- Modèle Client-Serveur
- Rôles des Navigateurs
- Front-End et Back-End
- Outil necessaire pour démarrer la programmation web

Par Robert DIASSÉ

Fondamentaux du Web

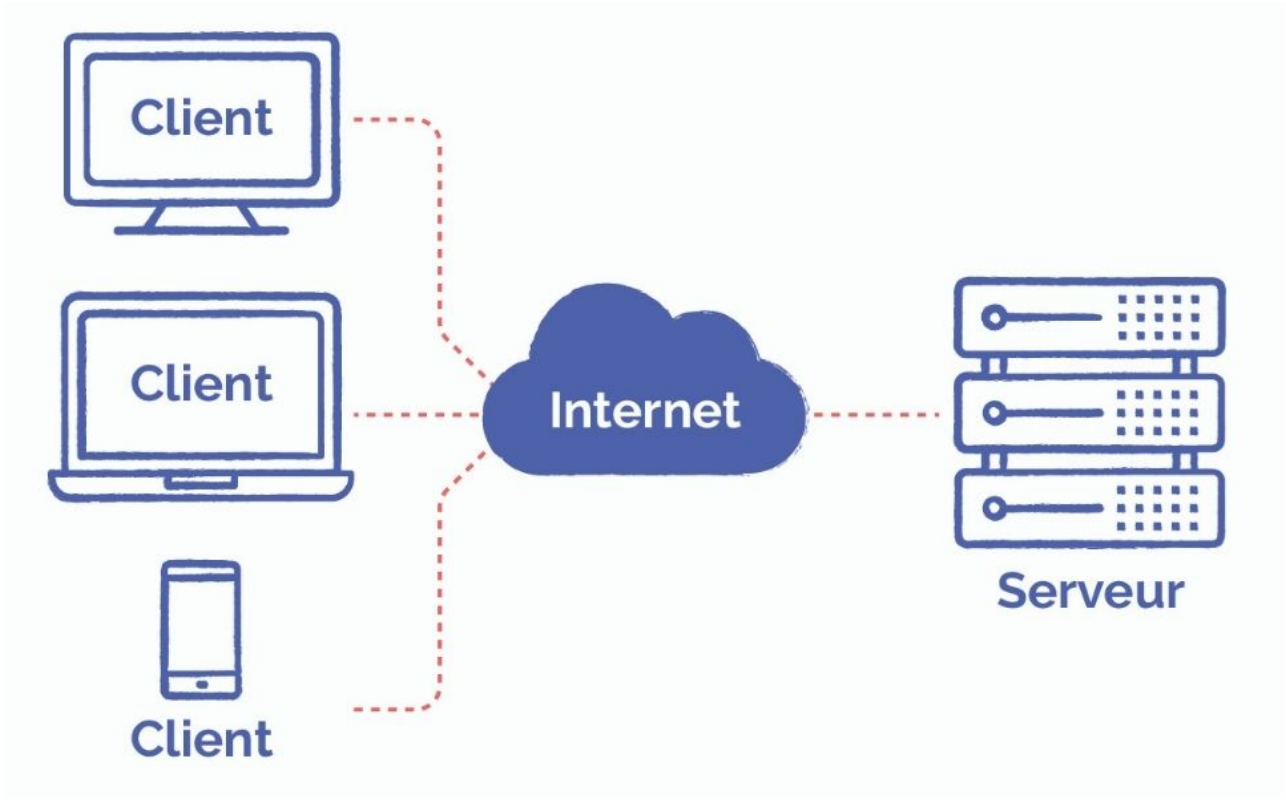
Le World Wide Web, souvent abrégé en WWW, est un système mondial d'information accessible via Internet. Il permet aux utilisateurs de consulter des ressources (pages web, images, vidéos, etc.) liées entre elles par des hyperliens.
Le Web a révolutionné la communication, l'accès à l'information, le commerce en ligne, le travail depuis n'importe quel endroit et bien d'autres domaines encore.

Composants du Web

Le Web repose sur une architecture client-serveur. Les navigateurs web agissent en tant que clients, demandant des ressources aux serveurs web, qui les servent en réponse.
Quand vous entrez une URL dans un navigateur (par exemple, "https://www.google.com"), le navigateur envoie une requête HTTP au serveur de Google. Le serveur renvoie alors la page web, que le navigateur affiche pour que vous puissiez la consulter.

Modèle Client-Serveur

Le modèle client-serveur est le fondement du World Wide Web. Dans cette partie, nous plongerons plus en profondeur dans cette architecture, en examinant comment les navigateurs envoient des requêtes HTTP aux serveurs, qui renvoient ensuite des réponses. Comprendre ce modèle est essentiel pour travailler avec les technologies web.



Dans une architecture client-serveur, il existe plusieurs types de requêtes et de réponses.HTTP = HyperText Transfer Protocol est un protocole de communication utilisé pour transférer des données sur le World Wide Web (WWW). Voici quelques exemples courants :

Types de requêtes HTTP :

- **GET** : lire des données (lecture seule)
- **POST** : modification des données (insertion, modification, suppression)
- **PUT** : remplacement complet des données relatives à un enregistrement
- **DELETE** : suppression d'un enregistrement

En ce qui concerne les types de réponses, ils sont généralement déterminés par le code d'état HTTP renvoyé par le serveur. Voici quelques exemples courants :

- **200 OK** : La requête a réussi, et la réponse renvoie le résultat attendu.
- **201 Created** : La requête a réussi et un nouveau ressource a été créée en conséquence.
- **204 No Content** : La requête a réussi, mais il n'y a pas de représentation à renvoyer (c'est-à-dire que la réponse est vide).
- **400 Bad Request** : La requête n'a pas pu être comprise ou était incorrecte.
- **401 Unauthorized** : L'authentification a échoué ou n'a pas encore été fournie.
- **403 Forbidden** : L'authentification a réussi, mais l'utilisateur authentifié n'a pas accès à la ressource demandée.
- **404 Not Found** : La ressource demandée n'a pas pu être trouvée.
- **500 Internal Server Error** : Une erreur s'est produite dans le serveur.

Syntaxe d'une requête http

```
Méthode de Requête URI Protocole/Version En-têtes (Headers) Corps (Body,
facultatif)
```

1- Méthode de Requête : La méthode de requête indique l'action à effectuer(GET,POST etc.)

2- URI (Uniform Resource Identifier) : L'URI identifie la ressource cible de la requête. Il s'agit généralement d'une URL (Uniform Resource Locator) qui spécifie l'emplacement précis de la ressource sur le web.

3- Protocole/Version : Indique le protocole utilisé (par exemple, "HTTP" ou "HTTPS") suivi de la version. Par exemple, "HTTP/1.1" ou "HTTP/2".

4- En-têtes (Headers) : Les en-têtes HTTP contiennent des informations supplémentaires sur la requête, telles que des informations sur le client, le type de contenu accepté, des informations d'authentification, etc. Les en-têtes sont généralement présentés sous la forme de paires "Nom: Valeur".

5- Corps (Body, facultatif) : Le corps de la requête est utilisé pour envoyer des données au serveur. Il est généralement utilisé dans les requêtes `POST` pour envoyer des données de formulaire, des JSON, des fichiers, etc. Le corps est facultatif, selon la méthode de requête.

Exemple :

```
GET /page-daccueil.html HTTP/1.1 Host: www.exemple.com Accept: text/html,
application/xhtml+xml, application/xml;q=0.9, */*;q=0.8 User-Agent: Mozilla/5.0
(Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko)
Chrome/58.0.3029.110 Safari/537.36
```

Dans cet exemple, la méthode de requête est `GET`, l'URI est `/page-daccueil.html`, le protocole est `HTTP/1.1`, et il y a plusieurs en-têtes qui définissent les préférences du client. Il n'y a pas de corps dans cette requête car il s'agit d'une requête `GET`.

Ces interactions entre le client et le serveur permettent une communication efficace et dynamique dans l'architecture client-serveur.

Rôles des Navigateurs

Les navigateurs web jouent un rôle crucial dans l'interprétation des données reçues des serveurs. Ils comprennent le code HTML pour afficher la structure de la page, appliquent les styles CSS pour la mise en forme, et exécutent le code JavaScript pour ajouter des fonctionnalités interactives.
Les navigateurs Web peuvent exécuter plusieurs types de réponses. Voici quelques exemples courants :

1. **Réponses HTML** : Les navigateurs peuvent interpréter et afficher des documents HTML. Ces documents peuvent inclure des éléments tels que du texte, des images, des vidéos, des formulaires, etc.

2. **Réponses CSS** : Les navigateurs peuvent interpréter et appliquer des feuilles de style en cascade (CSS) pour styliser les documents HTML.

3. **Réponses JavaScript** : Les navigateurs peuvent exécuter du code JavaScript pour créer des pages Web interactives.

4. **Réponses JSON** : Les navigateurs peuvent traiter les réponses JSON (JavaScript Object Notation) qui sont souvent utilisées dans les applications Web modernes pour transférer des données.

5. **Réponses d'image** : Les navigateurs peuvent afficher plusieurs types d'images, y compris JPEG, PNG, GIF, SVG, etc.

6. **Réponses de redirection** : Les navigateurs peuvent suivre les réponses de redirection HTTP (codes de statut 3xx) pour charger une nouvelle URL.

7. **Réponses d'erreur** : Les navigateurs peuvent afficher des messages d'erreur en fonction des codes de statut d'erreur HTTP reçus (codes de statut 4xx pour les erreurs côté client et 5xx pour les erreurs côté serveur).

Il est important de noter que la façon dont un navigateur traite une réponse dépend en grande partie des en-têtes HTTP inclus dans la réponse.

Front-End et Back-End

Une distinction importante dans le développement web est la différence entre le front-end et le back-end.

- Le front-end se concentre sur la création de l'interface utilisateur en utilisant HTML et CSS.
- Le back-end gère la logique métier et la gestion des données. Comprendre cette distinction est essentiel pour une carrière réussie dans le développement web.

Outil nécessaire pour démarrer la programmation web

Avant de plonger plus profondément dans les technologies web, nous discuterons des outils nécessaires pour commencer à programmer des sites web. Cela inclut les éditeurs de code **IDE**, les navigateurs de développement, les systèmes de gestion de versions, et d'autres ressources utiles pour les développeurs web. Pour démarrer le développement web, il est important d'avoir les bons outils à votre disposition. Voici une liste d'outils essentiels pour commencer :

1. **Éditeur de Code** :

    - Visual Studio Code, SublimeText, Atome, Webstorm : éditeur de code open source très populaire avec une large gamme d'extensions pour faciliter le développement web.

2. **Navigateurs Web** :

    - Google Chrome : Utilisé par la majorité des développeurs web pour ses outils de développement avancés.
    - Mozilla Firefox : Dispose également d'outils de développement performants.
    - Microsoft Edge : Utile pour tester des sites web sous Windows.

3. **Système de Gestion de Versions** :

    - Git : Utilisé pour suivre les changements de code, collaborer avec d'autres développeurs, et gérer les versions de votre projet.

4. **Hébergement de Sites Web** :
    - GitHub : Plateforme d'hébergement de code source, utile pour le partage de projets et la collaboration.
    - GitLab : Une alternative à GitHub avec des fonctionnalités similaires.
    - Bitbucket : Offre également des fonctionnalités de suivi de problèmes et d'intégration continue.
5. **Environnement de Développement Local** :
    - XAMPP, MAMP ou WAMP (selon votre système d'exploitation) : Ces outils vous permettent de configurer un serveur web local pour le développement.
    - Node.js : Si vous travaillez avec des frameworks JavaScript (comme React ou Vue.js), Node.js est essentiel pour exécuter des scripts JavaScript côté serveur.
6. **Outils de Conception et de Prototypage** :
    - Adobe XD, Sketch ou Figma : Pour créer des maquettes et des prototypes d'interfaces utilisateur.
    - Balsamiq : Utile pour créer des wireframes et des maquettes rapidement.
7. **Gestionnaires de Paquets** :
    - npm (Node Package Manager) : Pour gérer les bibliothèques JavaScript.
    - Composer : Pour les projets PHP.
    - pip : Pour les projets Python.
8. **Bases de Données** :
    - MySQL, PostgreSQL, SQLite : Des systèmes de gestion de bases de données couramment utilisés pour stocker des données web.
    - MongoDB : Une base de données NoSQL populaire pour des cas d'utilisation spécifiques.
9. **Outils de Test et de Débogage** :
    - Postman : Pour tester des API.
    - DevTools de navigateur (déjà mentionnés) : Pour inspecter et déboguer le code HTML, CSS et JavaScript.
10. **Outils de Productivité** :
    - Slack : Pour la communication d'équipe.
    - Trello, Asana, ou Jira : Pour la gestion de projets et de tâches.
11. **Systèmes d'exploitation** :
    - Choisissez un système d'exploitation qui correspond à vos préférences (Windows, macOS, Linux) et assurez-vous d'avoir un environnement de développement approprié pour ce système.

Ces outils forment une base solide pour le développement web. En fonction de vos besoins spécifiques et de vos projets, vous pourrez ajouter d'autres outils et technologies à votre boîte à outils.