

# Bienvenue

Développeur Web  
Les bases

Powered by  Sliddev



# Je me présente



Thomas PIERRE,  
Développeur Front-End senior



## Mon parcours :

■ 2013 - 2015, Responsable de communication numérique chez **Harley Davidson**



■ 2015 - 2016, License développement de project Web chez **Efficom**



■ 2015 - 2017, Intégrateur chez **PéoLéo**



■ 2017 - 2018, Développeur Front-End chez **HobbyNote**

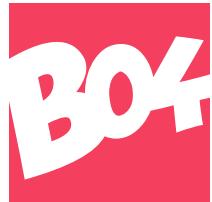


■ depuis 2018, Lead Développeur Front-End chez **Atecna**



■ depuis 2015, Gérant de **StudioB04**





## Micro agence numérique 360.

StudioB04 est une micro agence spécialisée dans la création de sites Web, et de supports de communication.

Nous accompagnons nos clients dans leur stratégie numérique et leur communication.  
Nous militons pour un Web plus sobre et plus accessible. Nos activités :

- Développement et maintenance de solutions Web (sites vitrines, plateformes e-commerce, applications, etc.)
- Audit d'accessibilité numérique (106 critères du R.G.A.A)
- Formation et sensibilisation à l'accessibilité numérique et au numérique responsable
- Interventions en centre de formation

```
nce-vue/src M
mance-svelte/src M
ce-react/src M
erformance-svelte/src
ormance-vue/src/components U
mance-react/src/TableCel U
erformance-react/src/services U

15  });
16
17  const toggleList = () => {
18    show = !show;
19  };
20
21  const addCount = (quant) => {
22    for (let i = 0; i < quant; i++) {
23      number += 1;
24    }
25  };
26
27  const disableCountry = (country) => {
28    data = data.map((item) => {
29      if (item.country == country) {
30        item.disabled = !item.disabled;
31      }
32      return item;
33    });
34  };
35  </script>
36
<main>
  <button>{number}</button>
  <button>+count(1)>add</button>

```

# Le programme

## Introduction et bases fondamentales

1. Rappel des bases de l'environnement Web.
2. Présentation des outils de développement.
3. Présentation du projet de formation.

# 1. L'environnement Web

# Qu'est ce qu'une application Web ?

Une application Web est un programme informatique conçu pour être exécuté par un navigateur internet.

Il peut s'agir d'un simple site internet, d'un site plus complexe comme un e-commerce, d'une application mobile ou d'un outil de gestion d'entreprise par exemple.

Les applications Web sont en général composées de 2 parties bien distinctes :

# Le Front-End

## ou "côté client"

C'est tout ce que l'utilisateur voit :

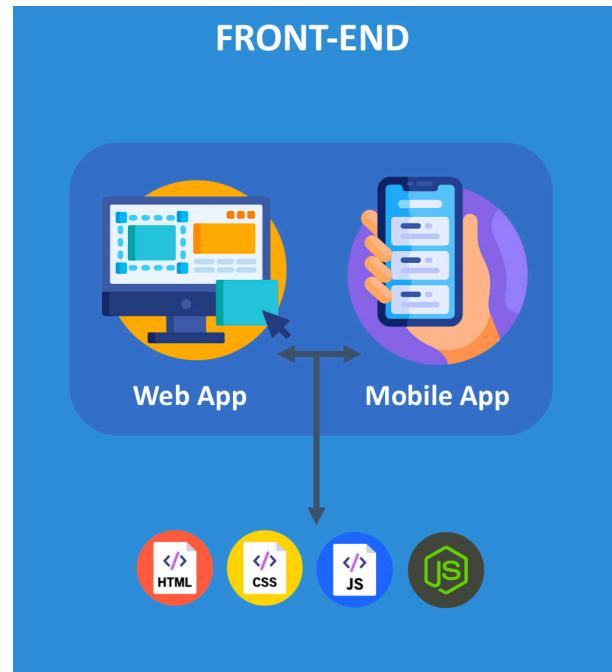
Le design, les animations, les graphiques, etc.

Depuis l'arrivée des **frameworks JS**, de plus en plus de tâches sont confiées au **Front-End** pour des applications plus rapide et plus fluides mais toujours plus complexes et gourmandes en ressources.

Le développeur Front-End a également en partie la responsabilité du **SEO**, de la **WebPerf** et de l'**accessibilité**.

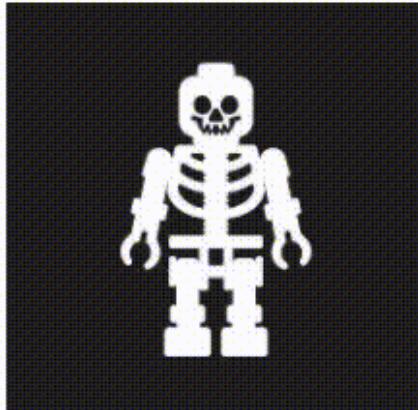
### ⚠️ Attention !

Quelles que soient les technologies utilisées pour le développement, le code devra être compilé afin d'être compris et interprétés par les navigateurs Web ! Les seuls langages que le navigateur comprend sont **HTML**, **CSS** et **JavaScript**.



# Le Front-End

**HTML**  
structure



**HTML**

C'est la structure de  
la page: le squelette.

**CSS**  
presentation/appearance



**CSS**

C'est l'apparence  
de la page : la peau.

**JavaScript**  
dynamism/action



**JavaScript**

Ce sont interactions et les  
animations : les muscles.

# Le Back-End

## ou "côté serveur"

C'est tout ce que l'utilisateur ne voit pas :

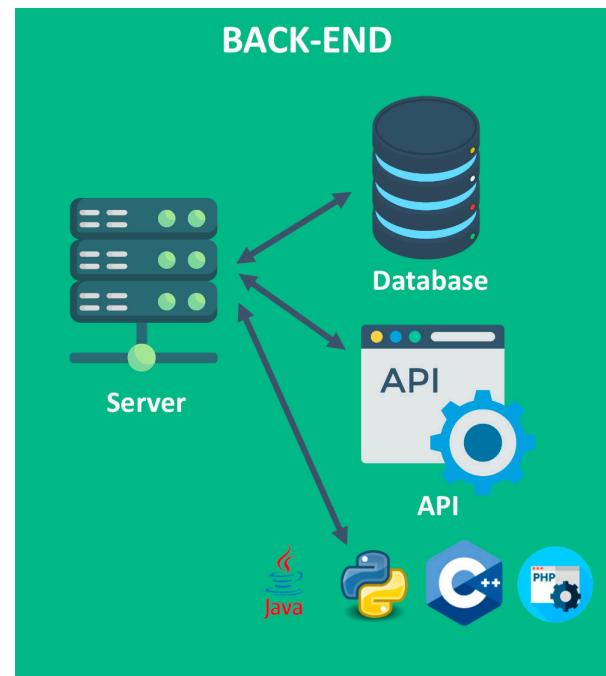
Les requêtes vers la base de données, les API, l'authentification, etc.

Les utilisateurs ne peuvent pas voir ou interagir directement avec le code puisqu'il est exécuté sur le serveur.

En revanche, leurs actions sur la page peuvent déclencher des actions qui seront gérées par le **Back-End** comme un formulaire d'inscription par exemple.

### Important !

Contrairement au Front-End, il n'y a aucune limitation de langage pour le développement côté Back-End puisque le code sera exécuté par le serveur : NodeJS, PHP, Java, .Net, C#, Python, Ruby, etc...



# En résumé



## Front-End

Le **Front-End** est responsable de l'expérience utilisateur. Il doit proposer des interfaces responsives, performantes et accessibles.

## Back-End

Le **Back-End** est responsable de la qualité de la donnée qui sera affichée au client, de la rapidité des temps de réponse, de la sécurité, etc.

### Important !

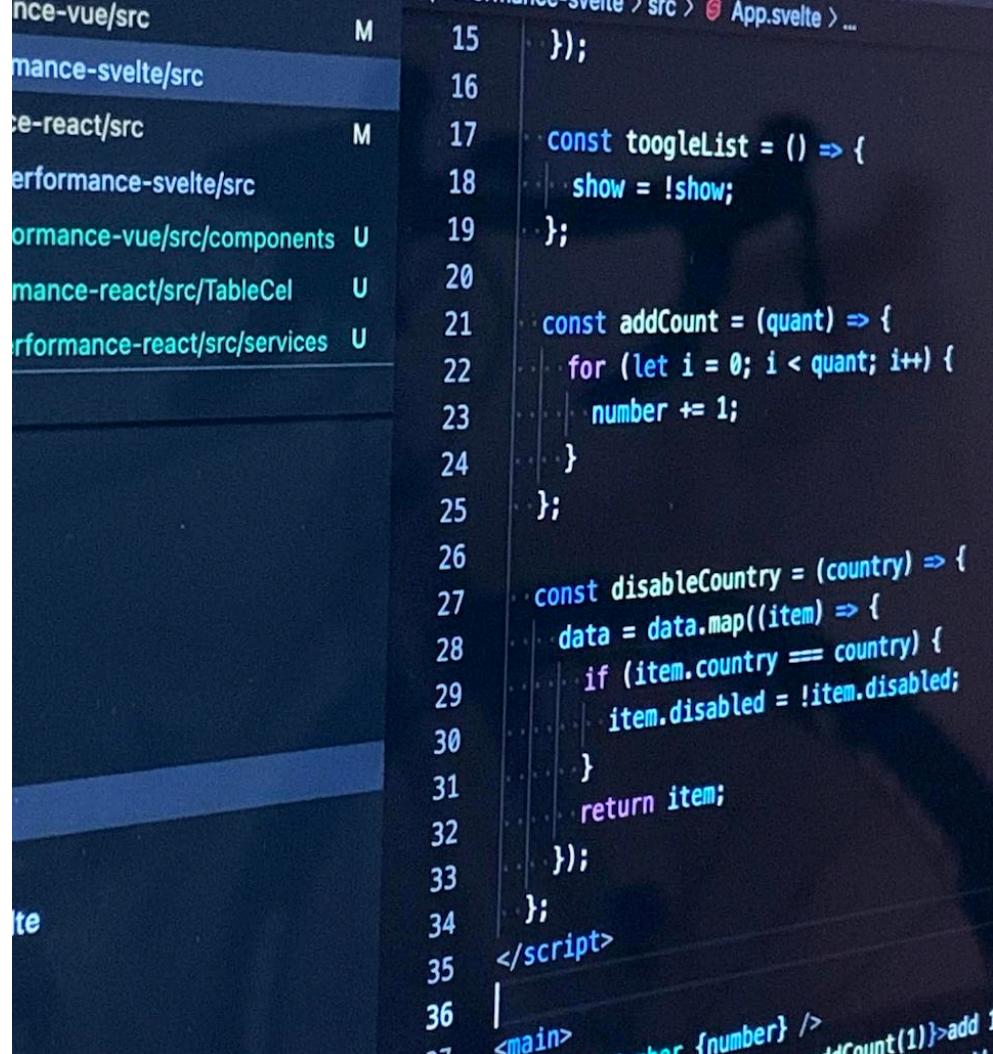
Les deux sont aussi indispensables l'un que l'autre au bon fonctionnement d'une application Web et doivent communiquer entre eux de manière efficace pour que l'application fonctionne.

# En résumé

Bien que souvent, le Front-End et le Back-End cohabitent tous les deux au sein du même projet (un seul dépôt pour toute la code base), il est de plus en plus fréquent qu'il soit séparés et gérés par des équipes totalement différentes.

Ainsi, une application Web peut tout à fait avoir un **Front-End** qui se connecte à un ou plusieurs **Back-End** via des API, ou qu'un même **Back-End** serve plusieurs **Front-End** différents.

On parle alors d'architecture **M.A.C.H** pour **Microservices, API, Cloud et Headless** : Au lieu de créer une grosse application monolithique, on va créer plusieurs petites briques indépendantes.



```
App.svelte > src > App.svelte ...  
15 );  
16  
17 const toggleList = () => {  
18   show = !show;  
19 };  
20  
21 const addCount = (quant) => {  
22   for (let i = 0; i < quant; i++) {  
23     number += 1;  
24   }  
25 };  
26  
27 const disableCountry = (country) => {  
28   data = data.map((item) => {  
29     if (item.country == country) {  
30       item.disabled = !item.disabled;  
31     }  
32     return item;  
33   });  
34 };  
35 </script>  
36  
37 <main> {#number} /> <button>+</button> <button>-</button>  
38 <ul>{list}</ul>
```

# Les protocoles

Comme on l'a vu précédemment, pour qu'une application Web fonctionne, il faut que le **Front-End** et le **Back-End** puissent communiquer.

**Voici un exemple concret :**

- Un utilisateur tape une recherche sur le site de **Google**.
- Lorsqu'il appuie sur entrée, une requête est envoyée au serveur de **Google**.
- En plus de la recherche, la requête va contenir des informations sur l'utilisateur (langue, géoloc, etc.).
- Le serveur va chercher la réponse dans ses différentes bases de données.
- Puis il va compiler toutes ces données pour formuler une réponse qu'il retourne au client.
- Le client affiche la réponse qu'il a reçue.

## Bon à savoir !

Une page Web est constituée de plusieurs éléments distants comme des **images**, des **polices**, des fichiers **CSS** et **JS**, etc. Pour chacun de ces éléments une requête est envoyée à un ou plusieurs serveurs.

Votre navigateur envoie constamment un grand nombre de requêtes lors de la navigation.

# Le protocole HTTP

## (HyperText Transfer Protocol)

Le **HTTP** est l'un des protocoles les plus fondamentaux et les plus utilisés dans le Web. Il permet la communication entre un **client** (généralement un navigateur) et un **serveur Web**.

C'est ce protocole de communication qui a été utilisé dans l'exemple précédent.

HTTP est un protocole basé sur un modèle **requête-réponse** :

1. **Requête** : Le **client** envoie une requête au serveur.
2. **Réponse** : Le **serveur** traite la requête et renvoie une réponse, souvent sous forme de page **HTML**, de donnée au format **JSON**, ou de **fichier**.

### 💡 Le saviez-vous ?

C'est la création du protocole HTTP en 1991 qui a permis l'invention d'Internet tel qu'on le connaît aujourd'hui.

# Structure d'une requête HTTP

Une requête HTTP est composée de plusieurs éléments :

## Ligne de requête

- l'URL (chemin d'accès du serveur qu'on cherche à joindre).
- la méthode ( GET , POST , DELETE , etc.) pour dire au serveur ce qu'on souhaite faire.

## Les en-têtes (Headers)

Fournissent des métadonnées sur la requête (la langue, le type de contenu, des informations sur l'utilisateur, etc.)

## Le Corps (Body, optionnel)

Peut contenir des données supplémentaires, souvent utilisées avec les méthodes comme POST ou PUT

# Les méthodes HTTP

Les méthodes définissent l'action que le client souhaite effectuer sur le serveur :

- `GET` : Récupérer une ressource (lecture seule)
- `POST` : Envoyer des données au serveur
- `PUT` : Créer ou mettre à jour une ressource
- `DELETE` : Supprimer une ressource

D'autres méthodes existent mais sont très peu utilisées, comme :

- `HEAD` : Récupérer uniquement les métadonnées d'une ressource.
- `OPTIONS` : Vérifier les méthodes disponibles pour une ressource.
- `PATCH` : Mettre à jour partiellement une ressource.

# Structure d'une réponse HTTP

Une réponse HTTP comprend :

## Ligne détat

Contient la version du protocole, un code de statut et un message. Exemple : `HTTP/1.1 200 OK`

## Les en-têtes (Headers)

Fournissent des informations sur la réponse, comme le type de contenu, la taille, ou des informations de cache. Exemple : `Content-Type: text/html; charset=UTF-8`

## Le Corps (Body, optionnel)

Contient les données demandées (**HTML**, **JSON**, **image**, etc.).

# Les codes de statut HTTP

Les codes de statut indiquent le résultat de la requête :

## 1xx : Information

`100 Continue` : La requête peut continuer.

## 2xx : Succès

`200 OK` : La requête à réussie.

`201 Created` : La ressource à été créée.

## 3xx : Redirections

`301 Moved Permanently` : URL déplacée de façon permanente.

`302 Found` : Redirection temporaire.

## 4xx : Erreurs côté client

`400 Bad Request` : La requête est mal formulée.

`401 Unauthorized` : L'authentification est nécessaire.

`404 Not Found` La ressource demandée est introuvable.

## 5xx : Erreurs côté serveur

`500 Internal Server Error` : Erreur interne au serveur.

`503 Service Unavailable` : Serveur temporairement indisponible.

[Voir tous les codes sur MDN Web Doc](#)

# Les versions d'HTTP

## 1. HTTP/0.9 (1991)

- Première version du protocole **HTTP**, très simple et minimaliste
- Limité à des requêtes `GET` pour récupérer des fichiers **HTML** uniquement.
- Aucune en-tête, ni statut, ni métadonnées.

## 2. HTTP/1.0 (1996)

- Support des en-têtes pour inclure des métadonnées (comme `Content-Type` )
- Introduction des codes de statut ( `200` , `404` , etc. ).
- Chaque requête ouvre une nouvelle connexion.

## 3. HTTP/1.1 (1997)

- Prend en charge les connexions persistantes (réutilisation des connexions).
- Ajout de nouvelles méthodes (comme `OPTIONS` , `PUT` , et `DELETE` ).
- Introduction de la gestion de cache avec des en-têtes comme `Cache-Control`

# Les versions d'HTTP

## 4. HTTP/2 (2015)

- Multiplexage des requêtes : plusieurs requêtes/réponses simultanées sur une seule connexion TCP.
- Compression des en-têtes HTTP pour réduire la bande passante.
- Priorisation des ressources pour optimiser les performances.

## 5. HTTP/3 (2022)

- Basé sur **UDP** au lieu de **TCP**.
- Réduction des temps de latence grâce à une connexion plus rapide et résiliente.
- Conçu pour améliorer les performances des applications en temps réel (streaming, jeux, etc.).
- Uniquement en **HTTPs**

# La sécurité avec HTTPs

HTTPs (HyperText Transfer Protocol Secure) ajoute une couche de sécurité à HTTP grâce au chiffrement des données.

Les connexions HTTPs reposent sur un certificat de chiffrement et protègent contre les attaques de type interception (*man-in-the-middle*). Seuls le **client** et le **serveur** connaissent la clé qui permet de déchiffrer les données.

HTTPs garantie l'intégrité et la confidentialité des données échangées

## ⚠️ Attention !

HTTPs est devenu la norme sur internet. 99% des sites utilisent ce protocole sécurisé et les navigateurs vous mettront en garde si vous tentez d'accéder à un site via une URL non sécurisée en HTTP.

## 💡 Astuce !

Pour s'assurer que vous êtes bien sur un site sécurisé, pensez à vérifier l'url complète, elle doit commencer par `https://`. Un clique sur l'icone vous donnera également des informations sur le protocole utilisé.

# Les autres protocoles

En dehors de HTTP, plusieurs autres protocoles jouent un rôle clé dans le fonctionnement d'Internet et des applications réseau :

- **FTP** (**File Transfer Protocol**) Utilisé pour transférer des fichiers entre un client et un serveur.
- **SFTP** (**Secure File Transfer Protocol**) Une version sécurisée de FTP
- **SMTP** (**Simple Mail Transfer Protocol**) Utilisé pour envoyer des e-mails.
- **IMAP** (**Internet Message Access Protocol**) Permet de recevoir des e-mails.
- **SSH** (**Secure Shell**) Permet une gestion et une connexion sécurisées à distance à un serveur.
- **WebSocket** Permet une communication bidirectionnelle et en temps réel entre un navigateur et un serveur, utilisé pour des applications comme les chats et les jeux en ligne.
- Etc...

## 2. Les outils de développement

# Les outils de développement

En tant que développeur, vous devrez être capable de comprendre et d'analyser le fonctionnement d'une application Web pour déceler les bugs et les corriger.

Les navigateurs embarquent des outils de développement pour aider les développeurs à diagnostiquer et optimiser leurs applications Web. Le navigateur le plus utilisé et le plus pratique pour le développement est **Google Chrome** grâce aux **DevTools**.



## 💡 Pour ouvrir les DevTools

- sur Windows ou Linux : `Ctrl + Shift + I` ou `F12`
- sur Mac : `Cmd + Option + I`

# Vue d'ensemble des onglets des DevTools

1. **Elements** : Explorer et modifier le DOM et le CSS en direct.
2. **Console** : Permet d'exécuter du code JavaScript, afficher des messages et des logs
3. **Sources** : Parcourir tous les fichiers sources qui constituent la page en cours.
4. **Network** : Surveiller les requêtes réseau (HTTP) et leur performance.
5. **Performance** : Analyser, et optimiser la **WebPerf** en surveillant les temps de chargement et d'exécution
6. **Memory** : Identifier les fuites de mémoire.
7. **Application** : Gérer les données stockées localement (**localStorage**, **cookies**, etc.).
8. **Security** : Vérifier les certificats et la sécurité des ressources.
9. **Lighthouse** : Générer un audit des performances, de l'accessibilité, et du SEO.

## 💡 Pratique !

Il est possible de trier les onglets selon vos préférences ou d'en ajouter de nouveau grâce à des Extensions Chrome !

Passons en revue les principaux onglets

# Elements

Cette page se divise en deux parties :

- Le DOM (Document Object Model). C'est la structure HTML du document en direct.

- Des outils pour manipuler les styles CSS, les propriétés, les attributs d'accessibilité, etc.

## Démonstration

The screenshot shows the Chrome DevTools interface with the 'Elements' tab selected. The top part displays the DOM tree for the current page, which includes various script and style elements. A red arrow points from the text 'HTML du document en direct' to this DOM tree. The bottom part of the screenshot shows the 'Computed' tab in the styles panel, which lists all the CSS properties applied to the selected element, along with their values and source. Another red arrow points from the text 'Des outils pour manipuler les styles CSS' to this 'Computed' tab. To the right of the styles panel, there is a visual representation of the element's bounding box and its padding, border, and margin, showing a green rectangle inside an orange one.

```
> <script nonce="..."></script>
> <h1 class="bNg8Rb" style="webkit-user-select:none">Liens d'accessibilité</h1>
> <div jscontroller="EufiNb" class="wYq63b">...</div> flex
> <div class="CvDJxb YNk70c" jscontroller="tIj4fb" jsaction="rcuQ6b:npT2md;" id="searchform">...
>   </div> grid
>   <span class="kpshf line gsr bilit big mdm" style="display:none"></span>
> <div class="main" id="main">...</div>
> <script nonce="..."></script>
>   <!-- cctlcm 5 cctlcm -->
> <style>...</style>
> <script nonce="..."></script>
> <script nonce="..."></script>
> <script nonce="..."></script>
<link href="/xjs/_ss/k=xjs.s.Sss9a00YxUg.L.B1.0/am=A0I0IAQAAATAABCAKgAIAAAAAAAAA...os,gwc,hsm,j...sa_mb4ZUb,_Et90b,SNUn3,qddgKe,sTsDMc,dtl0hd,eHDfl,YV5bee,d,csi" onerror="_rtf(this)" rel="stylesheet" nonce="..."/>
> <script nonce="..."></script>
<div></div>
```

html body#gsr.srp

Styles Computed Layout Event Listeners DOM Breakpoints Properties Accessibility

Filter :> .cls +, □

element.style {  
}  
@media (max-width: 1124px) { search?sc...34&dpr=1:24  
.srp {  
 --center-abs-margin: 28px;  
}  
.srp {  
 search?sc...34&dpr=1:24  
 --center-abs-margin: 180px;  
 --center-width: 652px;  
 --rhs-margin: 76px;  
 --rhs-width: 372px;  
 --lhs-refinements-width: 0px;  
 --lhs-margin: 24px;  
 --lhs-width: calc(var(--center-abs-margin) - var(--lhs-margin) - var(--center-margin));  
 --center-margin: 24px;  
 position: relative;  
 min-height: 100vh;  
}  
body {  
 search?sc...34&dpr=1:24

position 0  
margin --  
border --  
padding --  
0 - - - 1080x3387.420 - - - 0  
0 - - - - - - - - - 0

background-attachment scroll  
background-clip border-box  
background-color none  
background-image  
background-origin padding-box  
background-position-x 0%  
background-position-y 0%  
background-repeat repeat

# Console

La **Console** permet de manipuler le code **JavaScript** du document et d'afficher les messages (logs)

Les messages sont triés en fonction de leur types : **Messages**, **Erreurs**, **Warnings**, **Info** et **Verbose**.

## Démonstration

The screenshot shows the Chrome DevTools Network tab with the following details:

- Network Requests:** 179 messages, 11 errors, 160 warnings, 0 info, 8 verbose.
- Violations:** A warning about 'DOMContentLoaded' handler taking 313ms.
- Errors:** Multiple errors related to network requests being blocked by the client, such as 'ERR\_BLOCKED\_BY\_CLIENT' for various domains like connect.facebook.net, www.mobsuccess.com, via.batch.com, ams.creativecdn.com, secure.adnxs.com, idsync.rlccdn.com, www.decathlon.fr, s.yimg.com, login-ds.dotomi.com, and cookie-matching.mediarithmics.com.
- Warnings:** A warning about Chrome moving towards a new experience allowing users to browse without third-party cookies.
- Logs:** Numerous logs from various domains including gtm.js, https://www.google.com, www.google.es, www.google.fr, www.google.it, www.google.nl, www.google.be, www.google.pl, doubleclick.net, bat.bing.com, gstatic.com, sync.commander1.com, atmdt.com, tag.goldenbees.fr, crm4d.com, adsrvr.org, adnx.com, sdk.privacy-center.org, checkoutshopper-live.adyen.com, checkoutshopper-test.adyen.com, klarnaevt.com, klarna.com, klarnacd.com, klarnacd.net, voucher.decalthon.net, apigift.decalthon.com, site.booxi.com, and content.js.

# Network

L'onglet Network vous permet de voir toutes les requêtes HTTP envoyées par l'application.

Il est possible de trier les requêtes par type, et de voir le détail pour chaque requête (la durée, le statut, la taille de la resource, etc.).

## Démonstration

The screenshot shows the Network tab in the Chrome DevTools developer console. The tab is selected, and the main area displays a table of network requests. The columns in the table are Name, Status, Type, Initiator, Size, and Time. The table lists numerous requests, many of which are blocked or failed. The initiator column often shows URLs like 'clarity.js:2' or 'insight.old.min.js:1'. The size column shows file sizes in bytes, and the time column shows the duration of each request in milliseconds. The requests are categorized by type: xhr, fetch, and other. A filter bar at the top allows for filtering by various criteria like status code or initiator.

| Name                                  | Status        | Type  | Initiator            | Size  | Time   |
|---------------------------------------|---------------|-------|----------------------|-------|--------|
| collect                               | 204           | xhr   | clarity.js:2         | 287 B | 88 ms  |
| collect                               | 204           | xhr   | clarity.js:2         | 287 B | 90 ms  |
| wa/                                   | (blocked:...) | xhr   | insight.old.min.js:1 | 0 B   | 2 ms   |
| pzfaei?tid=G-8MP56RY1RX&v=2&gt...     | 200           | fetch | 6mr5ng6fpqk4un1.js?  | 85 B  | 38 ms  |
| attribution_trigger?pid=6400748&ti... | (blocked:...) | xhr   | insight.old.min.js:1 | 0 B   | 35 ms  |
| pzfaei?tid=G-8MP56RY1RX&v=2&gt...     | 200           | fetch | 6mr5ng6fpqk4un1.js?  | 84 B  | 46 ms  |
| pzfaei?tid=G-8MP56RY1RX&v=2&gt...     | 200           | fetch | 6mr5ng6fpqk4un1.js?  | 87 B  | 206 ms |
| skaze/                                | 200           | xhr   | VM111 lib.js:3       | 518 B | 87 ms  |
| skaze/                                | 200           | xhr   | lib.js:3             | 520 B | 80 ms  |
| f8352ae6-6681-4232-bcb7-dbe7fcc...    | 200           | fetch | scevent.min.js:2     | 425 B | 211 ms |
| collect                               | 204           | xhr   | clarity.js:2         | 287 B | 92 ms  |
| pzfaei?tid=G-8MP56RY1RX&v=2&gt...     | 200           | fetch | 6mr5ng6fpqk4un1.js?  | 84 B  | 45 ms  |
| collect                               | 204           | xhr   | clarity.js:2         | 287 B | 89 ms  |
| collect                               | 204           | xhr   | clarity.js:2         | 287 B | 88 ms  |
| collect                               | 204           | xhr   | clarity.js:2         | 287 B | 276 ms |
| collect                               | 204           | xhr   | clarity.js:2         | 287 B | 294 ms |
| collect                               | 204           | xhr   | clarity.js:2         | 287 B | 285 ms |
| collect                               | 204           | xhr   | clarity.js:2         | 287 B | 267 ms |
| collect                               | 204           | xhr   | clarity.js:2         | 287 B | 299 ms |
| collect                               | 204           | xhr   | clarity.js:2         | 287 B | 369 ms |
| collect                               | 204           | xhr   | clarity.js:2         | 287 B | 92 ms  |
| collect                               | 204           | xhr   | clarity.js:2         | 287 B | 90 ms  |
| collect                               | 204           | xhr   | clarity.js:2         | 287 B | 116 ms |
| collect                               | 204           | xhr   | clarity.js:2         | 287 B | 91 ms  |

# Application

L'onglet Application vous permet de voir tout ce qui est enregistré par l'application sur l'ordinateur : les Cookies, le Local Storage, le Session Storage, etc.

Il affiche également les services tels que les Notification Push, le cache, etc.

## Démonstration

The screenshot shows the Chrome DevTools interface with the 'Application' tab selected. On the left, a tree view lists various application components: Manifest, Service workers, Storage, Cookies, Local storage, Session storage, IndexedDB, Private state tokens, Interest groups, Shared storage, Cache storage, Storage buckets, Background services, and Frames. The 'Cookies' section is expanded, showing a list of stored cookies with columns for Name, Value, and various status indicators. A red arrow points from the text 'Il affiche également les services tels que les Notification Push, le cache, etc.' to the 'Background services' section. Another red arrow points from the text 'L'onglet Application vous permet de voir tout ce qui est enregistré par l'application sur l'ordinateur : les Cookies, le Local Storage, le Session Storage, etc.' to the 'Cookies' section. At the bottom right, a message says 'Select a cookie to preview its value'.

| Name        | Value               | D.   | P.   | E.   | S.   | H. | S.   | S.   | P.   | C. | P..  |
|-------------|---------------------|------|------|------|------|----|------|------|------|----|------|
| ABTasty     | uid=a3hp0mw4j...    | ...  | /    | 2... | 92   |    |      |      | L... |    | M... |
| ABTasty...  | mrasn=&lp=http...   | ...  | /    | 2... | 2... |    |      |      | L... |    | M... |
| AMCV_1...   | -637568504%7...     | ...  | /    | 2... | 2... |    |      |      |      |    | M... |
| Analytic... | AQJZaSQ6e0rrX...    | ...  | /    | 2... | 1... |    | ✓    | N... |      |    | M... |
| FPID        | FPID2.2.thZqQu...   | ...  | /    | 2... | 71   | ✓  | ✓    |      |      |    | M... |
| FPLC        | WZlGiU5Q5vH9...     | ...  | /    | 2... | 1... |    | ✓    |      |      |    | M... |
| UserMa...   | AQImBQQ0mWO...      | ...  | /    | 2... | 3... | ✓  | N... |      |      |    | M... |
| X-AB        | e4c8dfc3d7f84...    | s... | /... | 2... | 36   | ✓  | N... |      |      |    | M... |
| _ScCbts     | %5B%2212%3B...      | ...  | /    | 2... | 4... |    | L... |      |      |    | M... |
| _fbp        | fb.1.1737280908...  | ...  | /    | 2... | 33   | ✓  | L... |      |      |    | M... |
| _ga         | GA1.1.12578563...   | ...  | /    | 2... | 30   |    |      |      |      |    | M... |
| _ga_8M...   | GS1.1.17372809...   | ...  | /    | 2... | 59   |    |      |      |      |    | M... |
| _gcl_au     | 1.1.645491631.1...  | ...  | /    | 2... | 63   |    |      |      |      |    | M... |
| _gcl_aw     | GCL.173728090...    | ...  | /    | 2... | 1... |    |      |      |      |    | M... |
| _gcl_gs     | 2.1.k\$1i1737280... | ...  | /    | 2... | 3... |    |      |      |      |    | M... |
| _gtmeec     | e30%3D              | ...  | /    | 2... | 13   | ✓  | ✓    | S... |      |    | M... |
| _guid       | 6b4fd07-7f22...     | ...  | /    | 2... | 41   | ✓  | N... |      |      |    | M... |
| _scid       | acb839d4-b40f...    | ...  | /    | 2... | 41   | ✓  | L... |      |      |    | M... |
| _scid_r     | acwcFP3mCT5...      | ...  | /    | 2... | 39   |    |      |      |      |    | M... |
| _tt_ena...  | 1                   | ...  | /    | 2... | 18   |    |      |      |      |    | M... |
| _ttp        | 2pFD2NSPtHIXe...    | ...  | /    | 2... | 31   | ✓  | N... |      |      |    | M... |
| _ttp        | wnhEXqO9ECyR...     | ...  | /    | 2... | 36   |    |      |      |      |    | M... |
| aam_uuid    | 72832877648...      | ...  | /    | 2... | 4... |    |      |      |      |    | M... |
| bcookie     | "v=2&3b1ffe1a...    | ...  | /    | 2... | 4... | ✓  | N... |      |      |    | M... |
| cookie...   | 2                   | w... | /    | 2... | 14   |    | S... |      |      |    | M... |
| cookie...   | re622user...open... | w... | /    | 2... | 1    |    | S... |      |      |    | M... |

Select a cookie to preview its value

# Performance

L'onglet Performance permet d'auditer la performance d'une page Web en identifiant par exemple les resources trop lourdes, les goulets d'étranglement, etc.

La performance est présenté via les **WebCoreVitals** de Google qui sont des indicateurs sur des critères variés.

Vous pouvez également enregistrer une session pour mesurer la performance sur une navigation plus complète.

## Démonstration

The screenshot shows the Chrome DevTools Performance tab with the following details:

- Local metrics** section:
  - Largest Contentful Paint (LCP)**: 0.31 s (good). LCP element: img.d-block.w-100.img-object
  - Cumulative Layout Shift (CLS)**: 0.00 (good). Worst cluster: 3 shifts
  - Interaction to Next Paint (INP)**: - (not measured yet). Instructions: Interact with the page to measure INP.
- Next steps** section:
  - Field data**: See how your local metrics compare to real user data in the [Chrome UX Report](#).
  - Set up** button
- Environment settings** section:
  - Use the device toolbar to [simulate different devices](#).
  - CPU: No throttling
  - Network: No throttling
  - Disable network cache
- Record** and **Record and reload** buttons with keyboard shortcuts ⌘ E and ⌘ ⌂ E respectively.

The status bar at the top right shows: 6 68 47. The bottom navigation bar has tabs for **Interactions** and **Layout shifts**, with a trash icon.

The screenshot shows a developer tools interface with a sidebar on the left displaying a file tree and a main area on the right showing a code editor. The code editor contains the following JavaScript code:

```
15  });
16
17  const toggleList = () => {
18    show = !show;
19  };
20
21  const addCount = (quant) => {
22    for (let i = 0; i < quant; i++) {
23      number += 1;
24    }
25  };
26
27  const disableCountry = (country) => {
28    data = data.map((item) => {
29      if (item.country == country) {
30        item.disabled = !item.disabled;
31      }
32      return item;
33    });
34  };
35  </script>
36  <main> <div> <button>{number}</button> <button>+count(1)</button> <button>-count(1)</button> </div> </main>
```

# Conclusion

Les outils de développement sont indispensables pour analyser et comprendre ce qui se passe dans le navigateur lorsque l'application est exécutée.

Ils vous donnent également de précieuses indications sur ce que le serveur vous envoie via les requêtes HTTP.

Ils sont très complets et tous les onglets ne sont pas utiles à tous les développeurs, mais il est important de les connaître.

Suite de la formation →

# 3. Présentation du projet

Un portail sur le cinéma en Suisse

# Le projet

- Vous allez devoir vous répartir en groupes de 2 ou 3 personnes, pour concevoir un site Web sur le cinéma.
- Le design est libre mais le site doit être responsive. L'apparence n'est pas ce qui compte ici, mais ça joue sur la perception de vos utilisateurs, alors soyez créatifs !
- Votre projet devra être architecturé clairement, selon les design patterns que nous verrons plus tard dans le module.
- Vous devrez créer un compte sur [TMDB](#) pour vous connecter à leur API afin de récupérer les données nécessaires.
- Vous êtes libres d'utiliser toutes les librairies que vous souhaitez mais vous devrez justifier vos choix.
- Vous devrez déposer votre projet sur un repo public sur Github et le documenter via le fichier README.md.

MyMovie

Bienvenue sur le Portail MyMovie

Le site de référence du cinéma ! Retrouvez toutes les informations sur vos films préférés.

tueur

257 films trouvés pour "tueur" :

The interface displays a grid of movie posters for several slasher films, each with its title and year. The posters include:

- BELIEVE ME** (2018)
- PROFESSION TUEUR 2**
- KILLER KLOWNS FROM OUTER SPACE** (1981)
- VENDREDI 13 CAPTURE 2**
- Accident Man: Hitman's Holiday** (2023)
- Friday the 13th Part 2**
- STANLEY KUBRICK LE BAISER DU TUEUR**
- TUEURS-NÉS**
- PACTE AVEC UN TUEUR**
- Jack le Tueur de Géants**
- Killer's Kiss** (1955)
- Natural Born Killers** (1994)
- Best Seller** (1987)
- Jack the Giant Killer** (1962)
- ROBERT DUVALL TERENCE HANSON BONS BAISERS DU TUEUR**
- Bruce Willis Matthew Perry PIRANHA 2 Les Tournards Volants**

Une malfêteuse portant en elle des cloches assaillies de sang tombe sur la ville et commence alors à tuer tous les habitants d'une petite ville à coups de gâteau mortels.

Alors qu'il se croit mort noyé au fond de Crystal Lake, Jason décide de faire son retour pour se venger et tuer à sa manière. Après avoir assassiné Alice, seule survivante de la première opus, ...

Il est interdit de faire la copie privée des fichiers à fin de protection des droits d'auteur.

# Attendu du projet

Votre site devra contenir au minimum :

- Un moteur de recherche
- Une page de résultats de recherche avec des filtres et une pagination
- Une page pour les films (photos, vidéos, avis, résumé, etc)
- Une page pour les acteurs et membre de l'équipe technique (photo, biographie, filmographie, etc.)
- Une page de gestion des erreurs (404, 503, etc)

Vous devrez me remettre :

- Le lien du repo Github sur lequel le projet est déposé.  
(Votre projet doit contenir un fichier README.md qui explique comment installer le projet en local.)
- Le lien de votre site hébergé sur Netlify (nous verrons comment faire plus tard)
- Un document dans lequel vous présentez votre projet, et justifiez les choix techniques que vous avez fait.

# En bonus

Vous pouvez proposer autant d'autres fonctionnalités que vous souhaitez, comme par exemple :

- Un moteur de recherche pour les **séances en cours** dans une salle de cinéma
- Une page pour **écrire un avis** sur un film
- Une page pour voir les **bandes-annonces** des films à venir
- Un site multilangue
- etc.

## ⓘ Des points bonus !

Bien sûr, ces fonctionnalités ne sont pas obligatoires, mais elles peuvent vous permettre de gagner des points supplémentaires

## 💡 Besoin d'aide ?

Je serais disponible pour vous aider et répondre à toutes vos questions lors de nos sessions de cours, ou par email [contact@studiorb04.com](mailto:contact@studiorb04.com)

# Pour la fois prochaine

Jeudi 3 Avril 2025

# Pour la fois prochaine

1. Vous répartir en binômes pour le projet.
2. Concevoir le plan de votre application et réaliser des wireframes ou des maquettes.
3. Créer un compte sur [GitHub](#) et initier un dépôt pour le projet.
4. Créer un compte sur [TMDB](#) et parcourir un peu la documentation pour manipuler l'API.
5. Tester les envois de requêtes HTTP vers l'API pour comprendre son fonctionnement.
6. Manipuler les outils de développement de [Chrome](#).

## Important

Révoir les notions abordées aujourd'hui, une interrogation écrite est prévue au prochain cours !

Merci !