HTML5 et Multimédia

Intégration d’images, d’audio,
de vidéo et API multimédia

LP DWCA

Sommaire

1. Principes fondamentaux : HTML5 et multimédia
2. Les principales balises multimédia : images et vidéos
   - La balise <audio>
   - La balise <video>
   - La balise <img>
   - La balise <picture>
3. API Multimédia getUserMedia()
4. Les balises multimédias et l' API Multimédia getUserMedia() en détails

1. Principes Fondamentaux : HTML5 et multimédia

Définition d’un média ?

Procédé permettant la distribution, la diffusion ou la communication d'œuvres, de documents, ou de messages sonores ou audiovisuels (presse, cinéma, affiche, radiodiffusion, télédiffusion, vidéographie, télédistribution, télématique, télécommunication). On trouve aussi médium ou médium au singulier, et media, nom masculin invariable

Larousse

Le terme média (nom masculin) désigne tout moyen de distribution, de diffusion ou de communication (interpersonnelle, de masse ou de groupe) d’œuvres de documents, de messages écrits, visuels, sonores ou audiovisuels.
En général, il fait référence aux moyens de diffusion massive de l'information, comme :

- La presse (journaux, magazines)
- La radio
- La télévision
- Le cinéma
- Internet (sites web, réseaux sociaux, etc.)

Le mot vient du latin media, pluriel de médium (milieu, intermédiaire). On utilise souvent le singulier "un média" et le pluriel "les médias" en français.

Gemini

Qu’est ce qu’un média en HTML ?

Dans le contexte du HTML et du développement web, lorsque l'on parle de média, on fait référence à tout contenu non textuel qui sert de support d'information au sein d'une page web.

Ceci inclut notamment :

- Images et Photos : utilisation des balises <picture>, <img>, ou formats vectoriels comme le <svg>.
- Vidéos : Utilisation de la balise <video>.
- Audio : Utilisation de la balise <audio>.
- Contenu interactif / multimédia : Animations, graphiques dynamiques, etc., souvent intégrés via <iframe> ou d'autres techniques.

En bref, en HTML, les médias sont les vecteurs d'information et de contenu autres que le texte pur.

LP DWCA

HTML5 : Le standard multimédia

Le HTML (HyperText Markup Language) est le langage de balisage standard utilisé pour créer des pages web. Le passage à la version HTML5 a marqué une révolution majeure pour le multimédia.

- Rôle : HTML5 permet d'intégrer des contenus multimédias nativement, sans nécessiter de plugins externes (comme Flash autrefois).
- Apport Majeur : L'introduction des balises sémantiques <audio> et <video>, offrant un contrôle direct et standardisé de ces médias par le navigateur.

Enjeux du multimédia sur le web

L'intégration de ressources multimédias ne se limite pas à afficher un fichier ; elle englobe des défis techniques cruciaux :

- Performance : Le fichier (image, vidéo) doit être optimisé (bonne taille, bon format) pour garantir un chargement rapide de la page.
- Compatibilité : Le média doit pouvoir être lu sur tous les navigateurs et tous les systèmes d'exploitation (Desktop, Mobile).
- Accessibilité : Le contenu doit être compréhensible par tous, y compris les personnes ayant des déficiences visuelles ou auditives (via des textes alternatifs, des sous-titres, etc.).

2. Les principales balises multimédia : images et vidéos

La balise <audio>

L'élément HTML <audio> est utilisé afin d'intégrer un contenu sonore dans un document. Il peut contenir une ou plusieurs sources audio représentées avec l'attribut src ou l'élément <source>: le navigateur choisira celle qui convient le mieux. Il peut également être la destination de médias diffusés en continu, en utilisant un MediaStream.

Exemple de syntaxe simple

```
<audio controls autoplay>
  <source src="https://www.w3schools.com/html/horse.ogg" type="audio/ogg">
  <source src="https://www.w3schools.com/html/horse.mp3" type="audio/mpeg">
  Your browser does not support the audio element.
</audio>
```

L'exemple qui précède illustre le fonctionnement simple d'un élément <audio>. On inclut un chemin vers la ressource grâce à l'attribut src et on peut ajouter d'autres attributs afin de fournir d'autres informations : lecture automatique, lecture en boucle, utilisation des contrôles par défaut du navigateur, etc. Le contenu présent à l'intérieur des balises <audio></audio> est affiché comme contenu alternatif lorsque le navigateur ne prend pas en charge l'élément.

Les attributs

Cet élément inclut les attributs universels :

- autoplay : S'il est spécifié, l'audio commencera automatiquement la lecture dès qu'il pourra le faire, sans attendre la fin du téléchargement de l'ensemble du fichier audio.
  Note :
  - Les sites qui diffusent automatiquement de l'audio (ou des vidéos avec une piste audio) peuvent s'avérer désagréables pour les utilisateurs et doivent donc être évités dans la mesure du possible.
  - Si vous devez offrir une fonctionnalité de lecture automatique, vous devez la soumettre au choix de l'utilisateur.
  - Cependant, cela peut être utile lors de la création d'éléments médias dont la source sera définie ultérieurement, sous le contrôle de l'utilisateur.
  - Consultez ce guide sur la lecture automatique pour obtenir des informations supplémentaires sur la manière d'utiliser correctement la fonction autoplay.
- controls : Si l'attribut est présent, le navigateur affichera des contrôles pour que l'utilisateur puisse gérer la lecture, le volume, et le déplacement du curseur de lecture.

- crossorigin : Cet attribut à valeur contrainte indique comment le CORS doit être utilisé afin de récupérer la ressource. Les ressources utilisant le CORS peuvent être réutilisées dans un élément <canva> sans corrompre celui-ci. Les valeurs autorisées pour cet attribut sont :
  - anonymous : une requête multi-origine est envoyée sans information d'authentification. Autrement dit, l'en-tête HTTP Origin est envoyé sans cookie, certificat X.509 ou sans authentification HTTP. Si le serveur ne fournit pas d'information d'authentification au site d'origine (sans indiquer l'en-tête Access-Control-Allow-Origin), la ressource sera corrompue (tainted) et son utilisation sera restreinte.
  - use-credentials: une requête multi-origine est envoyée avec une information d'authentification. il envoie l'en-tête HTTP Origin : avec un cookie, un certificat ou une authentification HTTP Basic.

  Lorsque cet attribut n'est pas présent, la ressource est récupérée sans requête CORS et empêche ainsi d'utiliser la ressource dans un <canvas>. Si la valeur fournie est invalide, elle sera considérée comme anonymous. Voir Paramétrage des attributs relatifs au CORS pour plus d'informations.

A vous de jouer sur

CODEPEN

Cliquer sur le logo codepen

La balise <video>

L'élément HTML <video> intègre un lecteur de média qui prend en charge la lecture vidéo dans le document. Vous pouvez également utiliser <video> pour le contenu audio. Cependant l'élément audio peut fournir une expérience utilisateur plus appropriée.

Exemple de syntaxe simple

```
<video controls width="480">
    <source src="https://www.w3schools.com/html/mov_bbb.mp4" type="video/mp4" />
    <source src="https://www.w3schools.com/html/mov_bbb.ogg" type="video/ogg" />
    Télécharger la vidéo
    <a href="https://www.w3schools.com/html/mov_bbb.mp4">MP4</a>
    ou
    <a href="https://www.w3schools.com/html/mov_bbb.ogg">OGG</a>
</video>
```

L'exemple précédent illustre comment utiliser l'élément <video> simplement, à la façon d'un élément <img>. Le chemin vers le média à afficher est fourni via l'attribut src. On peut inclure d'autres attributs afin de spécifier la largeur et la hauteur, la lecture automatique et/ou en boucle, les contrôles affichés, etc. Le contenu fourni entre les balises <video></video> est affiché comme contenu alternatif par les navigateurs qui ne prennent pas en charge l'élément.

Les attributs

À l'instar des autres éléments HTML, cet élément inclut les attributs universels :

- autoplay : Un attribut booléen qui indique que la vidéo doit automatiquement être lancée dès qu'elle peut être jouée sans être arrêtée par le chargement des données.

  Note :

  Les navigateurs modernes bloquent l'audio (ou les vidéos avec une piste audio non assurée) de la lecture automatique. Les sites qui jouent automatiquement l'audio peuvent être une expérience désagréable pour les utilisateurs. Consultez notre guide sur la lecture automatique pour plus d'informations. Pour désactiver la vidéo automatique, autoplay="false" ne fonctionnera pas. La vidéo sera automatiquement lue si l'attribut est présent dans le tag <video>. Pour supprimer la lecture automatique, l'attribut doit être complètement supprimé.
- controls : Si cet attribut est présent, le navigateur affichera des contrôles pour permettre à l'utilisateur de contrôler la lecture de la vidéo, le volume et la mise sur pause. L'attribut controlslist, lorsqu'il est indiqué, aide le navigateur à choisir les contrôles à afficher pour la manipulation du média lorsque l'attribut controls est utilisé. Les valeurs autorisées pour cet attribut sont nodownload, nofullscreen et noremoteplayback. On utilisera l'attribut disablePictureInPicture afin de désactiver ce mode et les contrôles associés.

- crossorigin : Cet attribut à valeur contrainte indique comment le CORSdoit être utilisé afin de récupérer la ressource. Lesressourcesutilisant le CORSpeuvent être réutilisées dans un élément <canvas> sanscorrompre celui-ci. Lesvaleurs autorisées pour cet attribut sont :
  - anonymous : une requête multi-origine est envoyée sansinformation d'authentification. Autrement dit, l'en-tête HTTPOrigin est envoyé sanscookie, certificat X.509 ou sansauthentification HTTP.Sile serveur ne fournit pas d'information d'authentification au site d'origine (sansindiquer l'en-tête Access-Control-Allow-Origin), la ressource seracorrompue (tainted) et son utilisation serarestreinte.
  - use-credentials: une requête multi-origine est envoyée avecune information d'authentification. il envoie l'en-tête HTTPOrigin : avecun cookie, un certificat ou une authentification HTTPBasic.Si le serveur ne fournit pas d'informations d'authentification au site d'origine (c'est-à-direen n'envoyant pas l'en-tête HTTPAccess-Control-Allow-Credentials :, la vidéo seracorrompue et son utilisation serarestreinte

  Lorsque cet attribut n'est pas présent, la ressourceest récupérée sansrequête CORSet empêche ainsi d'utiliser la ressourcedans un <canvas>. Si la valeur fournie est invalide, elle seraconsidérée comme anonymous. Voir Paramétragedes attributs relatifs au CORSpour plus d'informations.
- disablepictureinpicture : Empêchele navigateurde suggérer un menu contextuel pour la superpositiond'une image/vidéo ("Picture-in-picture")ou de demander l'activationautomatique pour la superposition du média.
- disableremoteplayback : Un attribut booléen utilisé pour désactiverla capacitéde lecture à distancedans les appareils qui sont attachésà l'aide de technologies câblées (HDMI, DVI, etc.) et sansfil (Miracast, Chromecast, DLNA, AirPlay, etc.).
- height : Lahauteur de la zone où afficher la vidéo, exprimée en pixels CSS(en valeur absolue uniquement ; pas de pourcentages).

- loop : Un attribut booléen, qui, lorsqu'il est présent, indique que la vidéo doit être jouée en boucle.
- muted : Un attribut booléen qui indique s'il faut couper le son contenu dans la vidéo. Si cet attribut est utilisé, le son sera coupé au lancement de la vidéo. Sa valeur par défaut est false, ce qui signifie que l'audio sera lu lorsque la vidéo sera lue.
- playsinline : Un attribut booléen qui indique que la vidéo doit être jouée en incise, c'est-à-direau sein de la zone de lecture de l'élément.

  Note :

  L'absence de cet attribut n'implique pas que la vidéo sera lancée en plein écran.
- poster : Une URL qui contient une vignette à afficher tant que la vidéo est en cours de téléchargement. Si cet attribut n'est pas utilisé, rien n'est affiché jusqu'à ce que la première image de la vidéo soit disponible, ensuite, c'est cette image qui est affichée comme vignette sur la vidéo.
- preload : Cet attribut à valeur contrainte est une indication destinée au navigateur sur la meilleure façon de charger la vidéo. Il peut prendre l'une des valeurs suivantes :
  - none - la vidéo ne doit pas être préchargée
  - metadata - seules les métadonnées de la vidéo (sa durée par exemple) sont récupérées
  - auto - le fichier entier peut être téléchargé, même si l'utilisateur ne s'en sert pas
  - "" (chaîne vide) synonyme de la valeur auto La valeur par défaut peut être différente selon le navigateur. La spécification conseille d'utiliser la valeur metadata.

  Note :

  L'attribut autoplay a la priorité sur preload. Si autoplay est défini, le navigateur doit nécessairementtélécharger la vidéo pour la lancer. Cet attribut est simplement une indication, la spécification ne force pas le navigateur à respecter la valeur de cet attribut.

- src : L'URL de la vidéo à intégrer. Cet attribut est optionnel, l'élément <source> peut également être utilisé dans l'élément <video> afin d'indiquer la vidéo à intégrer.
- width : La largeur de la zone où afficher la vidéo, exprimée en pixels CSS (en valeur absolue uniquement ; pas de pourcentages).

Notes d'utilisation

Les navigateurs ne prennent pas en charge l'ensemble des formats vidéo. Vous pouvez fournir plusieurs sources grâce à des éléments <source>. Le navigateur utilisera la première ressource dont il connaît le format :

```
<video controls>
  <source src="maVideo.mp4" type="video/mp4" />
   <source src="maVideo.webm" type="video/webm" />
 <p>
    ## Notes d'utilisation
    Les navigateurs ne prennent pas en charge l'ensemble des formats vidéo.
    Votre navigateur ne prend pas en charge les vidéos HTML5. Voici
    <a href="maVideo.mp4">un lien pour télécharger la vidéo</a>.
  </p>
</video>
```

Lorsque vous utilisez des éléments <source>, le navigateur tente de charger chaque source séquentiellement. Si une source échoue (par exemple, en raison d'une URL non valide ou d'un format non pris en charge), la source suivante est tentée, etc. Un événement error se déclenche sur l'élément <video> après que toutes les sources ont échoué ; Les événements error ne sont pas déclenchés sur chaque élément individuel <source>. Nous proposons un guide des types de fichiers médias substantiels et approfondi, le guide des codecs pris en charge pour la vidéo. Il y a également un guide disponible pour les codecs audio qui peuvent être utilisés avec eux.

A vous de jouer sur

Cliquer sur le logo codepen

La balise <img>

L'élément HTML <img> intègre une image dans le document. C'est un élément vide (void element) qui n'a pas de balise de fermeture et ne peut pas contenir de contenu textuel. Il est fondamental pour l'affichage d'images sur le web.

Exemple de syntaxe simple

```
<img src="https://www.w3schools.com/tags/img_girl.jpg" alt="Girl in a jacket" width="500" height="600">
```

L'élément <img> est l'un des éléments les plus utilisés en HTML et est essentiel pour :

- Afficher des photos et illustrations
- Montrer des logos et icônes
- Présenter des graphiques et diagrammes
- Améliorer l'esthétique et la compréhension du contenu

Les attributs

Cet élément inclut les attributs universels :

- src (Attribut obligatoire) : Spécifie l'URL de l'image à afficher. C'est l'attribut le plus important de l'élément <img>.
- alt : Fournit un texte alternatif pour l'image. Ce texte est affiché si l'image ne peut pas être chargée et est lu par les lecteurs d'écran.
- width : La largeur intrinsèque de l'image en pixels (sans unité). Définir width et height permet d'éviter le layout shift pendant le chargement de la page.
- height : La hauteur intrinsèque de l'image en pixels (sans unité).

  Note :

  Les valeurs width et height définissent le ratio d'aspect de l'image, pas sa taille d'affichage. Utilisez CSS pour contrôler la taille d'affichage
- loading : Indique comment le navigateur doit charger l'image. Valeurs possibles :
  - eager : Charge l'image immédiatement, quelle que soit sa position dans la page (comportement par défaut)
  - lazy : Diffère le chargement de l'image jusqu'à ce qu'elle soit proche du viewport
- decoding : Fournit une indication au navigateur sur la façon de décoder l'image. Valeurs possibles :
  - sync : Charge l'image immédiatement, quelle que soit sa position dans la page (comportement par défaut)
  - async : Décode l'image de manière asynchrone pour réduire le délai de présentation d'autres contenus
  - auto : Pas de préférence pour le mode de décodage (par défaut)
- fetchpriority : Fournit une indication sur la priorité relative à accorder lors de la récupération de l'image. Valeurs possibles : high, low, auto

- srcset : Définit un ensemble d'images sourcesque le navigateur peut choisir en fonction de la densité de pixels ou de la largeur de la fenêtre.
- sizes : Définit un ensemble de tailles d'image pour différentes conditions de mise en page. Utilisé conjointement avec srcsetet des descripteurs de largeur (w).
- crossorigin : Indique si les requêtes CORSdoivent être utilisées lors de la récupération de l'image. Valeurs possibles : anonymous, use-credentials
- referrerpolicy : Définit quelle information de référence envoyer lors de la récupération de l'image. Valeurs : no-referrer, no-referrer-when-downgrade, origin, origin-when-cross-origin, same-origin, strict-origin, strict-origin-when-cross-origin, unsafe-url
- usemap : Associe l'image à une image map (carte cliquable) définie par un élément <map>. La valeur doit être le symbole # suivi du nom de l'élément <map>.
- ismap : Indique que l'image fait partie d'une image map côté serveur.Lorsque l'utilisateur clique sur l'image, les coordonnées du clic sont envoyées au serveur.

Formats d'image supportés

- JPEG (.jpg, .jpeg) : Photos, images complexes.
- PNG (.png) : Images avec transparence, graphiques.
- GIF (.gif) : Animations simples (préférer WebP/AVIF).
- SVG (.svg) : Logos, icônes, graphiques vectoriels.
- WebP (.webp) : Excellent compromis qualité/taille.

- AVIF (.avif) : Meilleure compression que WebP.
- ICO (.ico) : Favicons.
- BMP (.bmp) : À éviter (fichiers lourds).

A vous de jouer sur

CODEPEN

Cliquer sur le logo codepen

La balise <picture>

L'élément HTML <picture> est un conteneur utilisé pour spécifier plusieurs sources pour un élément <img> qu'il contient. Le navigateur choisira la source la plus appropriée en fonction de la résolution de l'écran, de la taille de la fenêtre, du format d'image supporté, et d'autres critères. Cet élément permet de fournir des images responsives et optimisées pour différents contextes d'affichage.

Exemple de syntaxe simple

```
<picture>
  <source media="(min-width:650px)" srcset="https://www.w3schools.com/tags/img_pink_flowers.jpg">
  <source media="(min-width:465px)" srcset="https://www.w3schools.com/tags/img_white_flower.jpg">
  <img src="https://www.w3schools.com/tags/img_orange_flowers.jpg" alt="Flowers" style="width:auto;">
</picture>
```

L'élément <picture> est particulièrement utile pour :

- L'artdirection : Afficher des images différentes selon le contexte
- L'optimisation des performances : Servir des formats modernes aux navigateurs compatibles
- La gestion des écrans haute résolution (Retina, 4K, etc.)
- L'économie de bande passante : Charger uniquement la version appropriée de l'image

Règles importantes :

- L'élément <img> est obligatoire et doit être le dernier enfant
- Les éléments <source> sont évalués dans l'ordre d'apparition
- La première source compatible est sélectionnée.
- L'attribut alt doit être sur l'élément <img>

Les attributs (sur <source>)

Cet élément utilise principalement des éléments <source> qui supportent les attributs suivants :

- srcset (Attribut requis) : Spécifie l'URL de l'image à utiliser dans différents contextes. Peut contenir plusieurs images avec descripteurs de taille ou de densité.
- type : Le type MIME de l'image. Permet au navigateur de choisir immédiatement une source sans télécharger l'image pour vérifier son format. Types MIME courants pour images :
  - image/avif : Format AVIF (très moderne, excellente compression).
  - image/webp : Format WebP (moderne, bonne compression).

    - image/jpeg : Format JPEG(classique,universellement supporté).
    - image/png : Format PNG (classique,transparence).
    - image/svg+xml : Format SVG(vectoriel)
- media : Spécifie une media query que le navigateur évalue pour déterminer si cette source doit être utilisée. Si la media query n'est pas satisfaite, le navigateur passe à la source suivante.
- sizes : Spécifie la taille d'affichage prévue de l'image pour différentes conditions de mise en page. Utilisé avec srcsetet des descripteurs de largeur (w) pour aider le navigateur à choisir l'image appropriée.
- width et height : Définissentles dimensions intrinsèques de l'image. Utiles pour éviter le reflow et améliorer les performances.

Cas d'usage courants

- Fourniture de formats modernes avec fallback : Servir des formats modernes optimisés (AVIF, WebP) aux navigateurs qui les supportent, avec un fallback JPEG/PNGpour les anciens navigateurs.
- Images responsive selon la taille de l'écran : Servir différentes versions d'une image selon la largeur de la fenêtre pour optimiser la bande passanteet le temps de chargement.
- Art Direction (images différentes selon le contexte) : Afficher des compositions d'image différentes selon le format d'écran. Parexemple, un recadrage différent pour mobile vs desktop.
- Support des écrans haute densité (Retina): Fournir des images en résolution standard et haute définition.

- Images selon l’orientation de l’appareil

A vous de jouer sur

CODEPEN

Cliquer sur le
logo codepen

2. API Multimédia getUserMedia()

L'APIMediaDevices.getUserMedia() permet d'accéder aux périphériques multimédia de l'utilisateur (webcam, microphone) depuis le navigateur. C'est une API moderne et puissante pour créer des applications de visioconférence, de capture photo/vidéo, d'enregistrement audio, etc.

Qu'est-ce que MediaDevices ?

MediaDevices est une interface qui donne accès aux périphériques média connectés comme :

- Caméras (webcams intégrées ou externes)
- Microphones (intégrés ou externes)
- Partage d'écran (avec getDisplayMedia())

Prérequis

Pour des raisons de sécurité, getUserMedia() ne fonctionne que sur HTTPS (en production) et localhost (en développement), pas sur HTTP (bloqué par les navigateurs).
Le navigateur demande toujours l'autorisation à l'utilisateur avant d'accéder à la caméra ou au micro.
getUserMedia() est compatible avec la plupart des navigateurs.

Exemple de syntaxe simple

```
navigator.mediaDevices.getUserMedia(constraints)
  .then(function(stream) {
    // Succès : utiliser le flux média
  })
  .catch(function(error) {
    // Erreur : gérer le refus ou l'absence de périphérique
  });
```

Paramètres (constraints)

L'objet constraints est un objet crucial qui spécifie les exigences pour l'audio et la vidéo. Il doit contenir au moins une clé parmi audio ou video avec une valeur true ou un objet de configuration détaillé.

- video : Si true, demande l'accès à la caméra vidéo par défaut. Si c'est un objet, il permet de spécifier des exigences détaillées comme la résolution, le taux d'images ou la caméra spécifique.
- audio : Si true, demande l'accès au microphone par défaut. Si c'est un objet, il permet de spécifier des exigences détaillées comme l'annulation d'écho ou le microphone spécifique.

L'utilisation d'objets pour les contraintes permet de garantir que le flux obtenu répond à des critères spécifiques. Le navigateur essaiera de satisfaire les contraintes au mieux.

Les contraintes de résolution (vidéo) spécifient les dimensions idéales, minimales ou maximales.

- width et height : Largeur et hauteur souhaitées ou requises.
- minWidth et minHeight : Exige la résolution minimale que vous pouvez accepter. Si le périphérique ne peut pas fournir cette résolution ou plus, la promesse sera rejetée avec une erreur.
- maxWidth et maxHeight : Définit la résolution maximale que vous souhaitez obtenir, même si le périphérique peut faire mieux.
- aspectRatio : La valeur souhaitée ou requise pour la largeur divisée par la hauteur (width / height).
- frameRate : Spécifie le nombre d'images vidéo que la caméra doit capturer et fournir par seconde, mesuré en images par seconde (FPS).

Les contraintes de périphérique (audio et vidéo) ciblent un appareil spécifique si l'utilisateur en a plusieurs.

- deviceId : Utilise l'appareil avec l'ID spécifié (obtenu via enumerateDevices()).
- facingMode (Vidéo uniquement): Demande la caméra arrière (pour les appareils mobiles). Les valeurs possibles sont "user" (frontal) ou "environment" (arrière).

Valeur de retour Promise<MediaStream>

La méthode renvoie une Promise qui se résout ou est rejetée.

- Résolution (Succès): Le Promise est résolu avec un objet MediaStream.
  - Le MediaStream contient un ou plusieurs objets MediaStreamTrack (pistes) pour l'audio et/ou la vidéo.
  - Ces pistes peuvent être ensuite associées à des éléments <video> ou <audio> pour l'affichage, ou utilisées dans des API comme WebRTC (RTCPeerConnection).
- Rejet (Échec): Le Promise est rejeté avec un objet DOMException décrivant pourquoi l'accès a échoué.

Le cas d'utilisation le plus courant est d'afficher le flux vidéo sur une page.

```
const videoElement = document.getElementById('localVideo');
navigator.mediaDevices.getUserMedia({ video: true, audio: false })
  .then(function(stream) {
    // Attache le MediaStream à l'élément vidéo
    videoElement.srcObject = stream;
    // Lance la lecture (nécessaire pour certains navigateurs)
    videoElement.play();
  })
  .catch(function(err) {
    console.error("Échec de l'accès à la caméra : ", err);
  });
```

La gestion des erreurs est cruciale pour informer l'utilisateur des problèmes d'accès au média.

- NotAllowedError : L'utilisateur a refusé l'autorisation dans la boîte de dialogue du navigateur. Demander à l'utilisateur de réinitialiser les permissions dans les paramètres du navigateur.
- NotFoundError : Aucune caméra ou microphone n'a été trouvé sur l'appareil. Vérifier si les périphériques sont branchés ou si les pilotes sont installés.
- NotReadableError : Le périphérique est en cours d'utilisation par une autre application ou est inaccessible. Fermer les autres applications utilisant la caméra/le micro.
- ConstraintNotSatisfiedError : Le périphérique trouvé ne peut pas respecter les contraintes demandées (ex. : résolution trop élevée). Relaxer les contraintes (ex. : demander une résolution plus faible).
- SecurityError : Le code est exécuté via HTTP et non HTTPS. Déployer le site sur un serveur sécurisé (HTTPS).

Il est essentiel d'arrêter explicitement toutes les pistes du MediaStream lorsque vous n'en avez plus besoin. Cela libère les ressources matérielles et éteint l'indicateur lumineux de la caméra.

```
// En supposant que 'stream' est l'objet MediaStream
stream.getTracks().forEach(track => {
  track.stop(); // Arrête la piste individuelle
});
```

L'API **getUserMedia()** est un outil puissant pour créer des applications multimédia interactives. Bien qu'elle ne fasse pas partie d'HTML5 pur, elle s'intègre parfaitement avec les éléments **<video>** et **<audio>** pour créer des expériences utilisateur riches.

Exemple 1

Afficher la webcam dans une vidéo

A vous de jouer sur

Cliquer sur le logo codepen

Exemple 2

Capturer une photo depuis la webcam

A vous de jouer sur

Cliquer sur le logo codepen

Exemple 3

Enregistrement audio

A vous de jouer sur

Cliquer sur le logo codepen

2. Les balises multimédias et l’ API Multimédia getUserMedia() en détails

Retrouver les balises multimédias et l’ API Multimédia getUserMedia() sur un site internet dédié avec une documentation détaillée et des exemples de code.