

Web & Mobile UI

Conception d'applications web

Loris Gavillet, HEIG-VD, 02.2022, v1.3

Enseignants

Cours

Phase 1

Loris Gavillet

loris.gavillet@heig-vd.ch

Semaine 1 -> Semaine 6

Phase 2

Nicolas Chablotz

nicolas.chablotz@heig-vd.ch

Semaine 7 -> Semaine 12

Objectifs

Cours

- Être capable de réaliser des applications Web progressive (Progressive Web App) en intégrant un design préalablement conçu
- Manipuler différentes API's pour la réalisation d'interfaces web modernes
- Mettre en place des patrons de conception (design patterns)

Contenu

Cours

- De HTML5 vers un standard HTML vivant
- Responsive design, Media Queries
- Data-attributes, custom elements
- API HTML: LocalStorage, History, Service Workers, ...
- Offline mode
- Début templating

Déroulement

Cours

Phase 1 (Semaine 1-6)

- Introduction au cours
- 1/3 Théorie - 2/3 Pratique
- Live coding - Corrections entre les cours
- Setup des outils de base
- Projet sur 9 “jours” de cours
- Examen sur 4 périodes

Technologies utilisées

Cours

Phase 1

- Webpack
- HTML / CSS / JS
- API HTML
- Templating
- PWA / Service workers
- Intro aux frameworks

Repository et informations générales

Cours

<https://github.com/lgavillet/webmobui-22>

Projet

Concept Projet



Spotlified

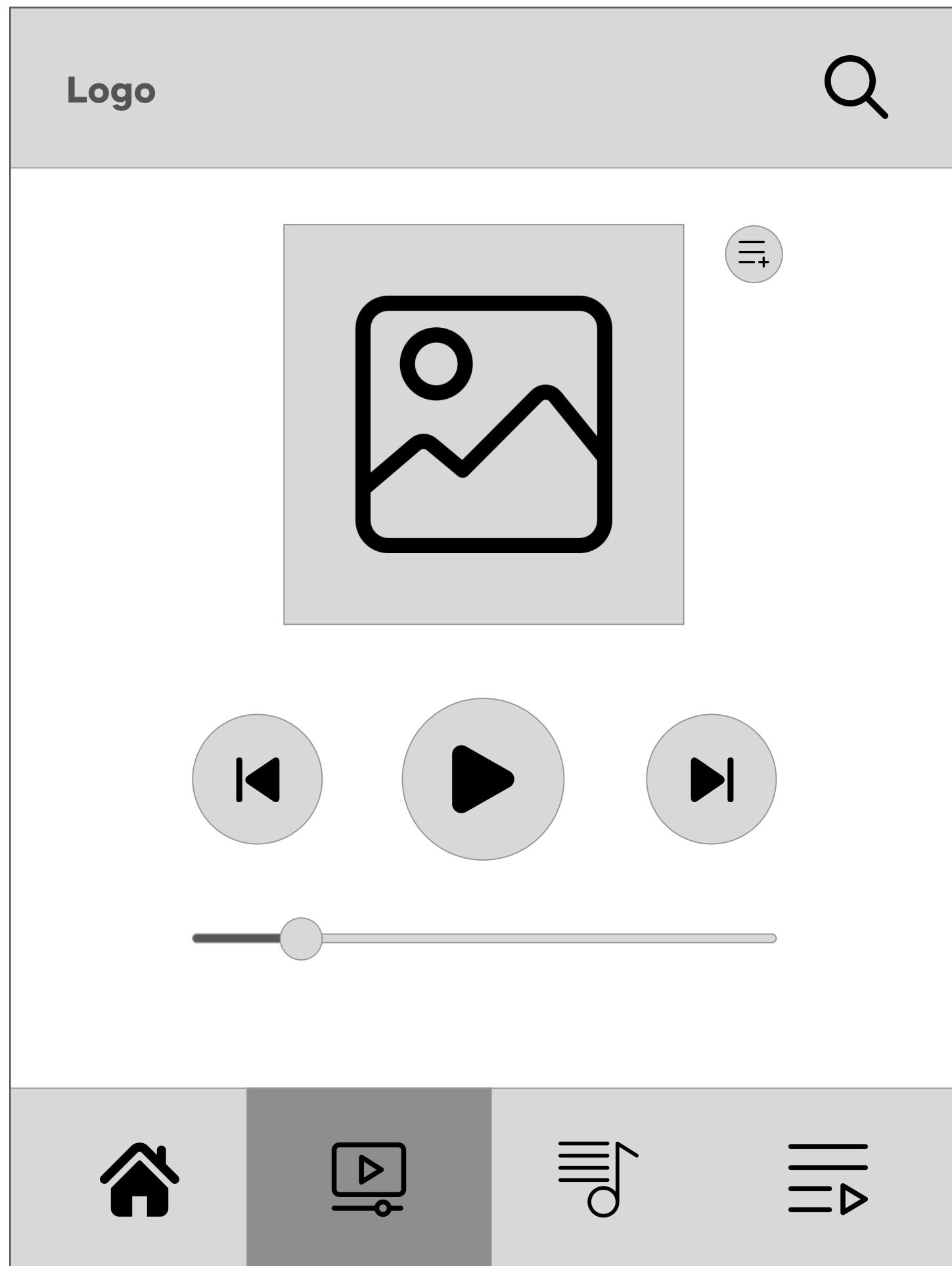
Simplified spotify

Concept Projet

Spotify version simplifiée

3 modèles :

- Chansons
- Artistes
- Playlists



Features

Projet

- Homepage
- Liste des artistes
- Liste des chansons par artiste
- Lecteur audio
- Champ de recherche
- Gestion online/offline
- Gestion des playlists
- Caching
- Installation de l'application

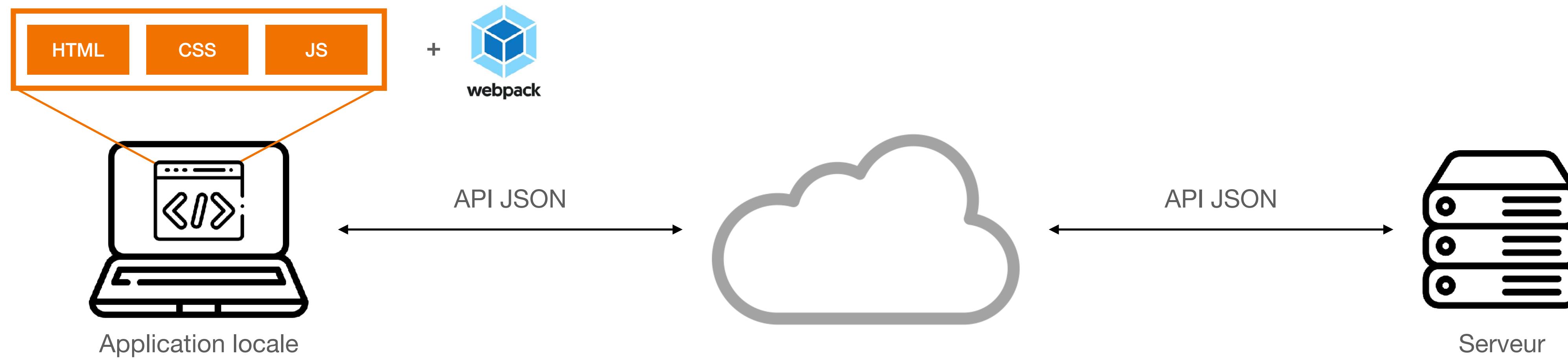
Prototype

Projet

<https://invis.io/DG12CQJQ9QPF#/463974083> Home

Architecture

Projet



Composants ?

Projet

- Projet webpack vide
- Squelette HTML
- Styles CSS structurels
- Icônes
- Routeur pour les pages web
- Client pour l'API JSON
- Lecteur audio
- Popover pour playlists
- Détection online/offline
- Local storage pour les playlists
- Manifest PWA
- Caching
- Service worker

Packages

What is it?

Packages



Définition

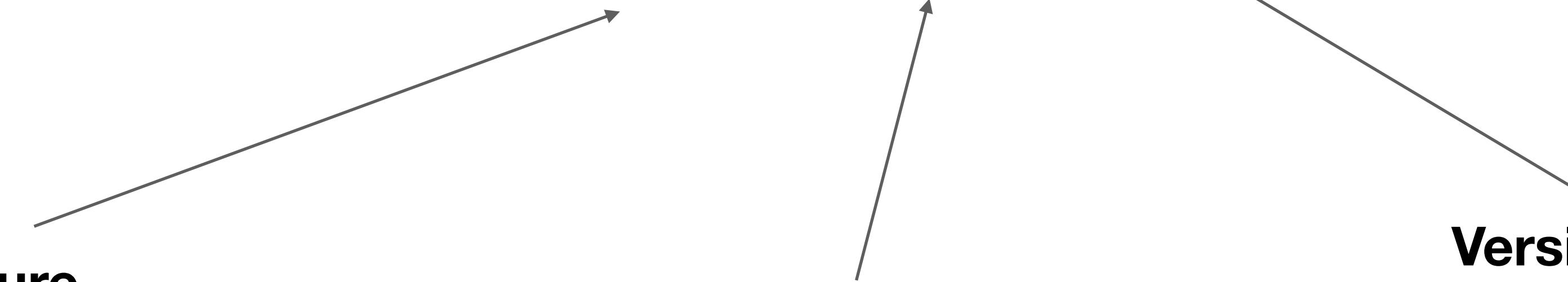
Package

- Bout de code réutilisable
- Expose des fonctionnalités
- Intégrable dans une application
- Versionable

Versioning - Semantic versioning

Package

v16.2.6



Version majeure

Le numéro de version MAJEUR quand il y a des changements non rétrocompatibles,

Version mineure

Le numéro de version MINEUR quand il y a des ajouts de fonctionnalités rétrocompatibles

Version patch

Le numéro de version de CORRECTIF quand il y a des corrections d'anomalies rétrocompatibles

Choix d'un package

Package

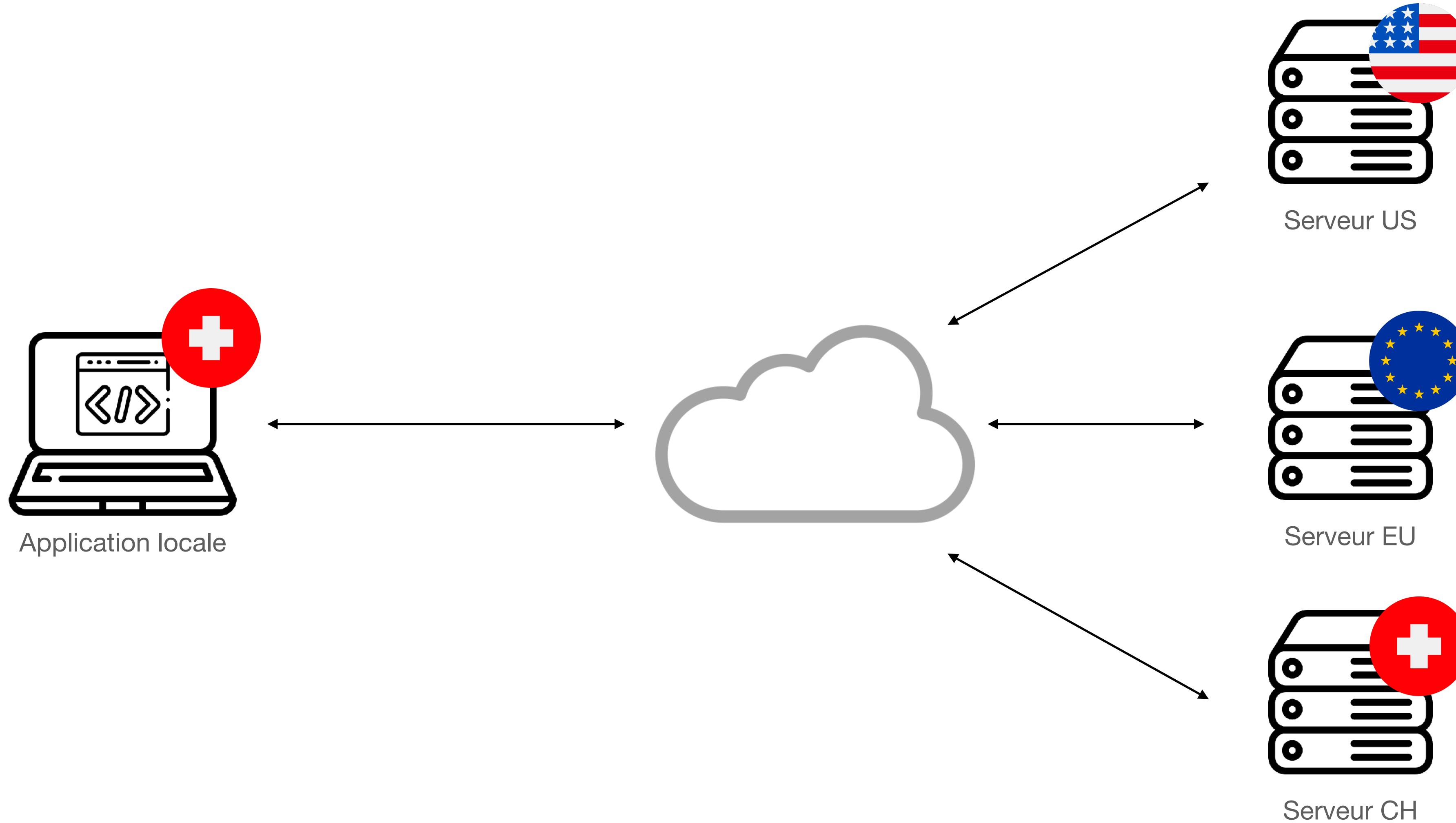
- Vérifier la version -> v0.x.x -> à bannir
- Nombre d'utilisateurs ?
- Maintenu ? Date dernier patch ?
- Dépendances ?

Intégration Package

1. Copy-paste old school / Téléchargement
2. CDN - Content Delivery Network
3. Package manager

CDN - Content Delivery Network

Package



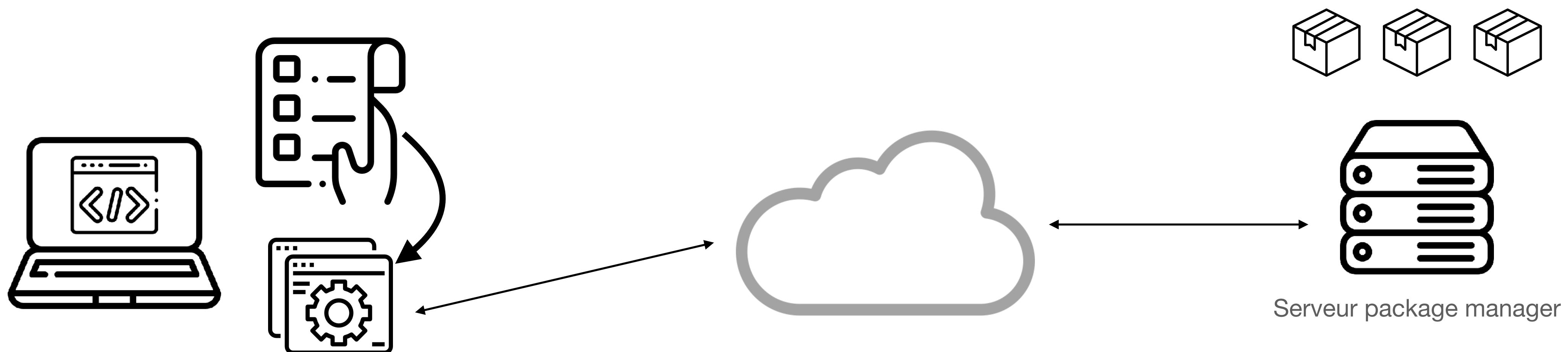
CDN - Content Delivery Network

Package

```
<link
  rel="stylesheet"
  href="https://fonts.googleapis.com/css2?family=Material+Icons"
/>
```

Package manager

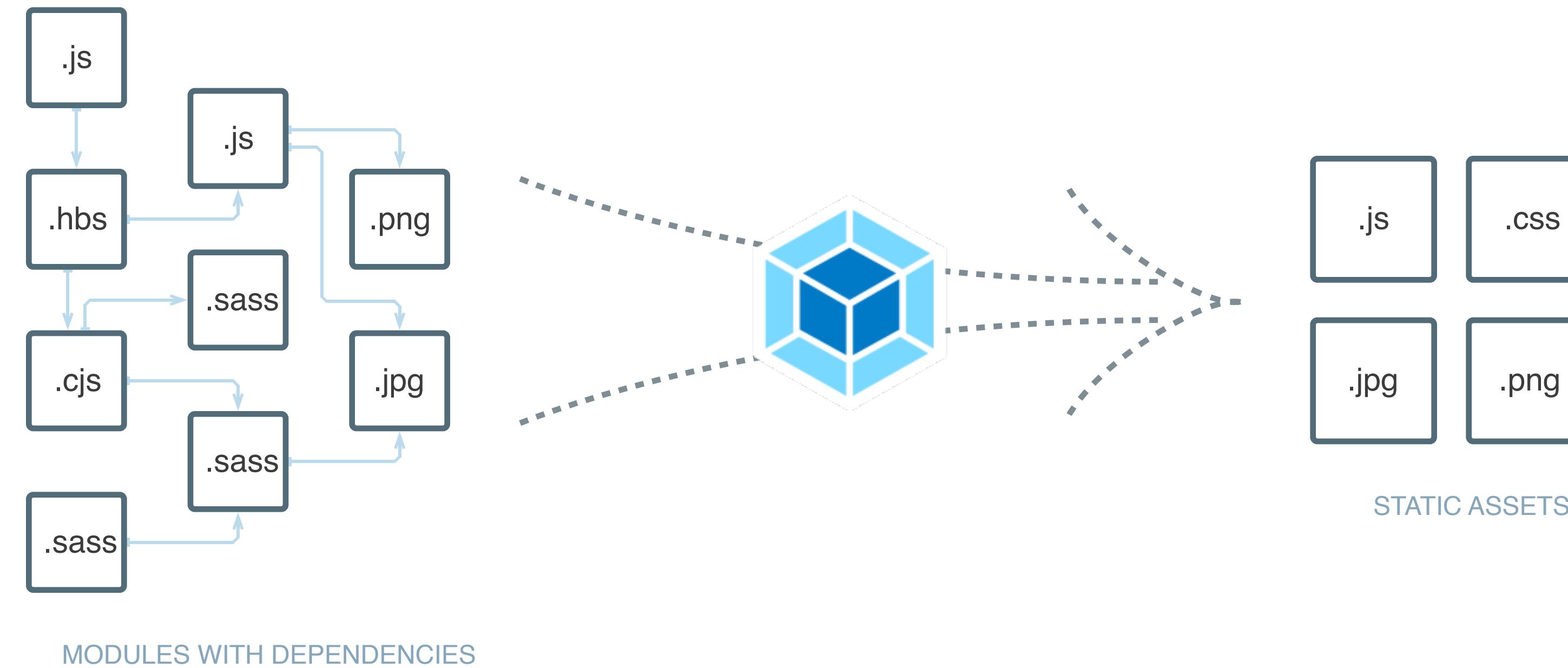
Package



Webpack

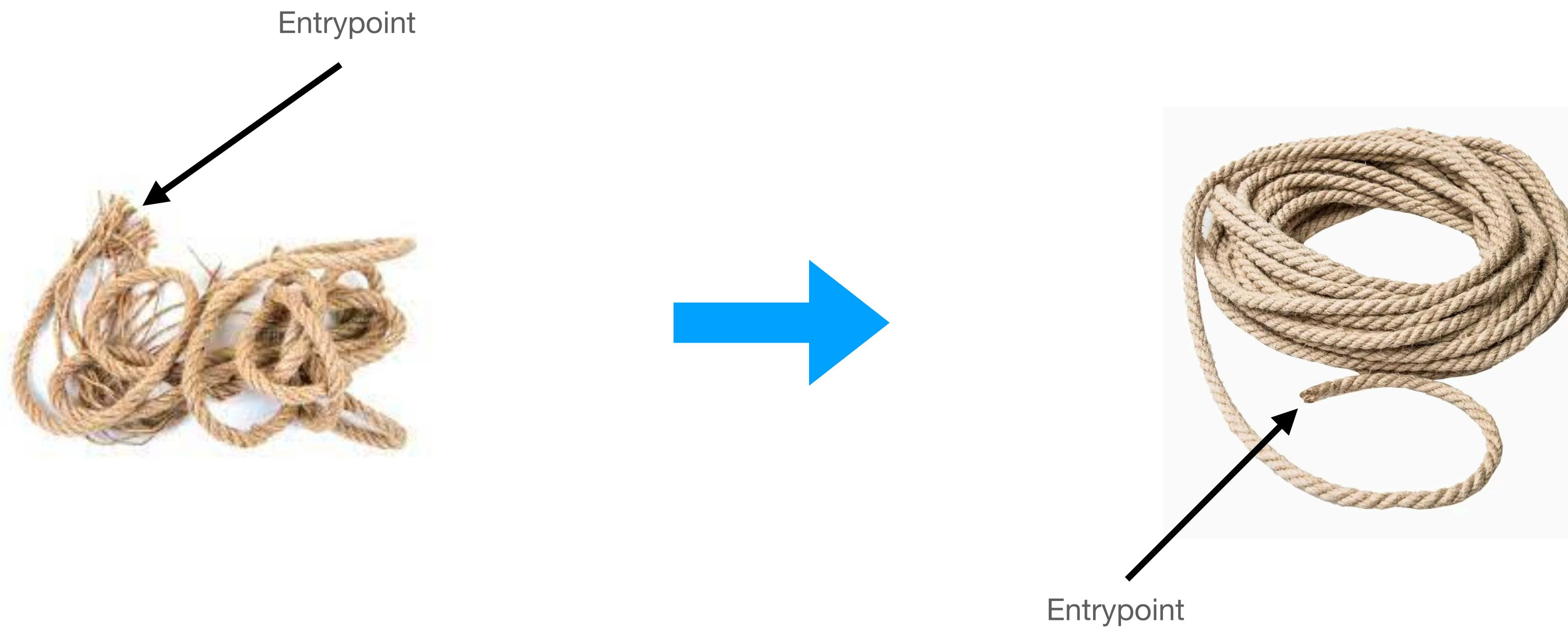
Concept

Webpack



Concept

Webpack



Concept

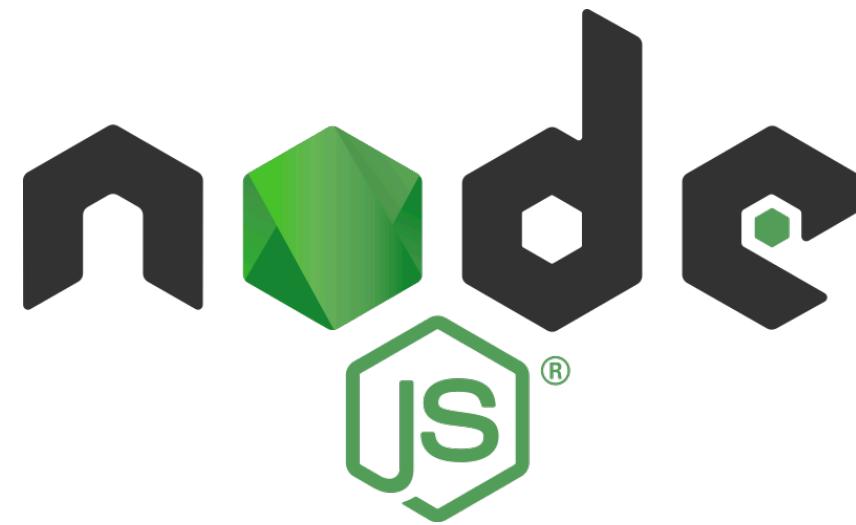
Webpack

“webpack is a module bundler. Its main purpose is to bundle JavaScript files for usage in a browser, yet it is also capable of transforming, bundling, or packaging just about any resource or asset.”

Concept

Webpack

Basé sur



- NodeJS est un langage de programmation backend, écrit en Javascript
- A ne pas confondre avec le Javascript frontend
- Node est livré avec un gestionnaire de package :
NPM (Node Package Manager)

Concept

Webpack

Basé sur



- Webpack est un package de NodeJS
- S'installe comme tout autre package avec
`> npm install webpack`

Concept

Webpack

- Webpack est un package de NodeJS
- S'installe comme tout autre package avec

```
> npm install webpack
```

Code

Pour le cours

Webpack

- Uniquement packaging en javascript et css
- Serveur web avec Hot reloading
- Pré-configuré et prêt à l'emploi

<https://webpack.js.org/>

Structure de base officielle

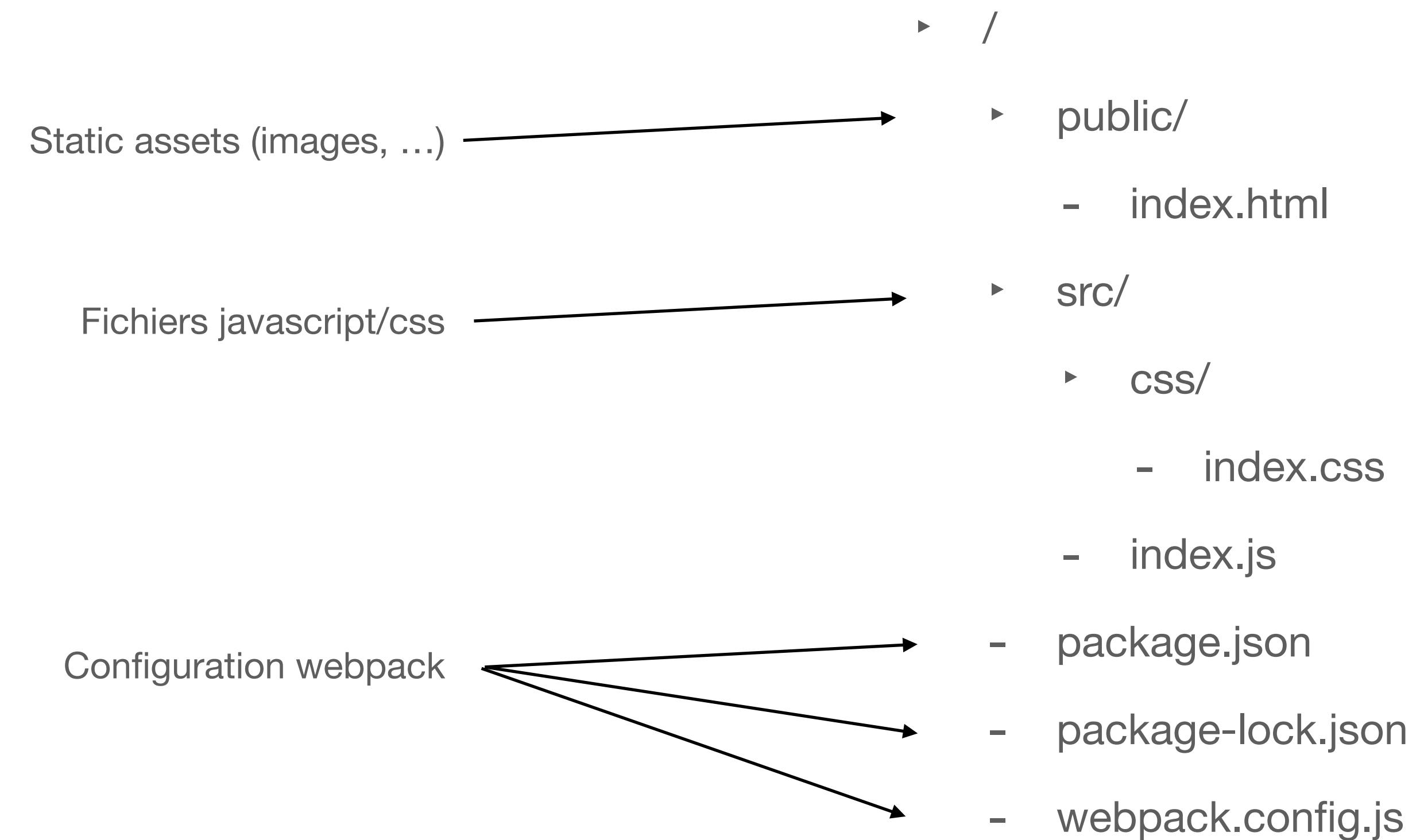
Webpack

- ▶ /
 - ▶ dist/
 - bundle.js
 - ▶ src/
 - ▶ ...
 - index.js
 - index.html
 - package.json
 - package-lock.json
 - webpack.config.js

<https://webpack.js.org/>

Structure pour le cours

Webpack



Installation

Webpack

Installation

- Installer Node (si pas déjà fait)
<https://nodejs.org/en/download/> Version LTS
- Créer un dossier pour le code et copier le projet vide :
[https://github.com/lgavillet/webmobui-22/tree/master/
Projet vide](https://github.com/lgavillet/webmobui-22/tree/master/Projet%20vide)
- Ouvrir l'invite de commande dans votre dossier projet
`> npm install`

<https://webpack.js.org/>

Utilisation Webpack

Démarrer le serveur de dev

```
> npm run start
```

<https://webpack.js.org/>

Node installé ?

Webpack

```
node --version
```

Go!

HTML / CSS / JS

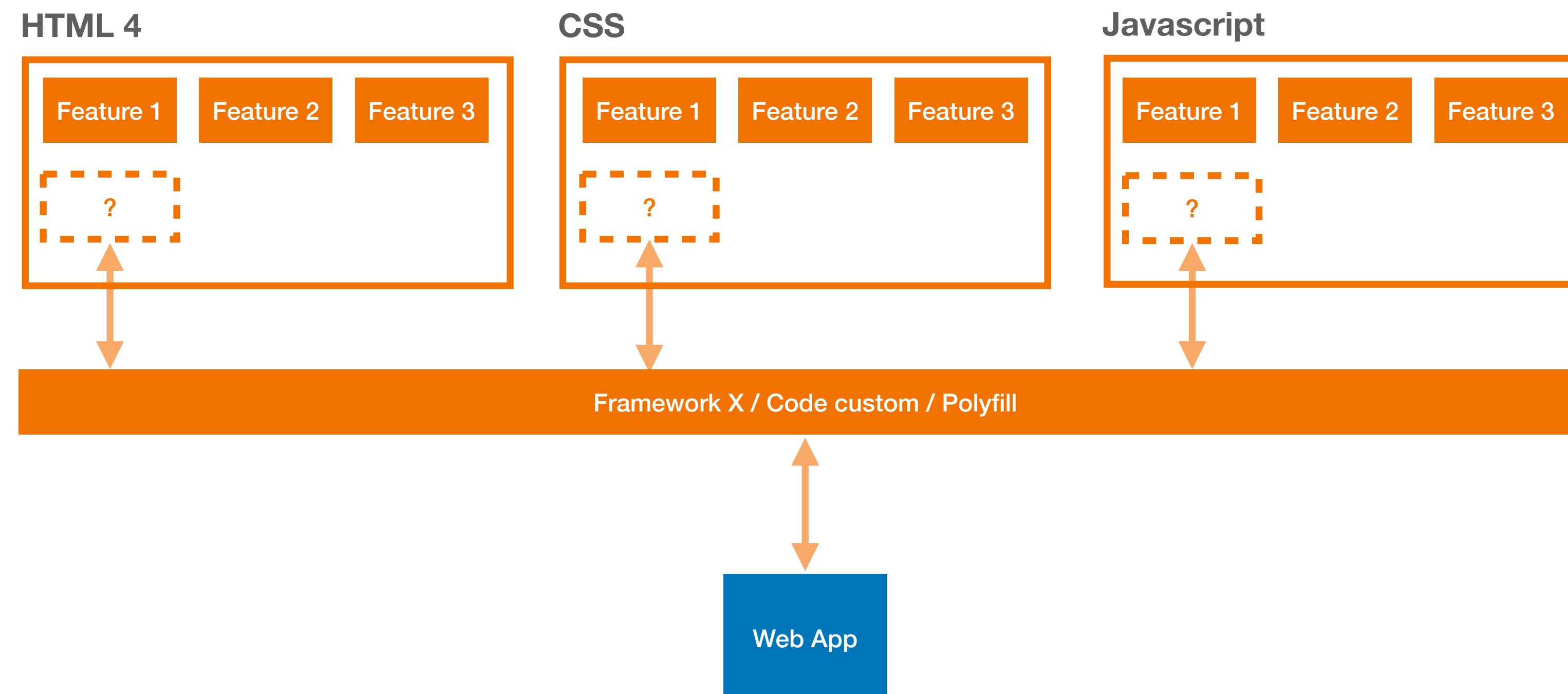
Nouveau standard HTML

HTML / CSS / JS

“Vers une fin des dépendances et une collaboration inter-langages”

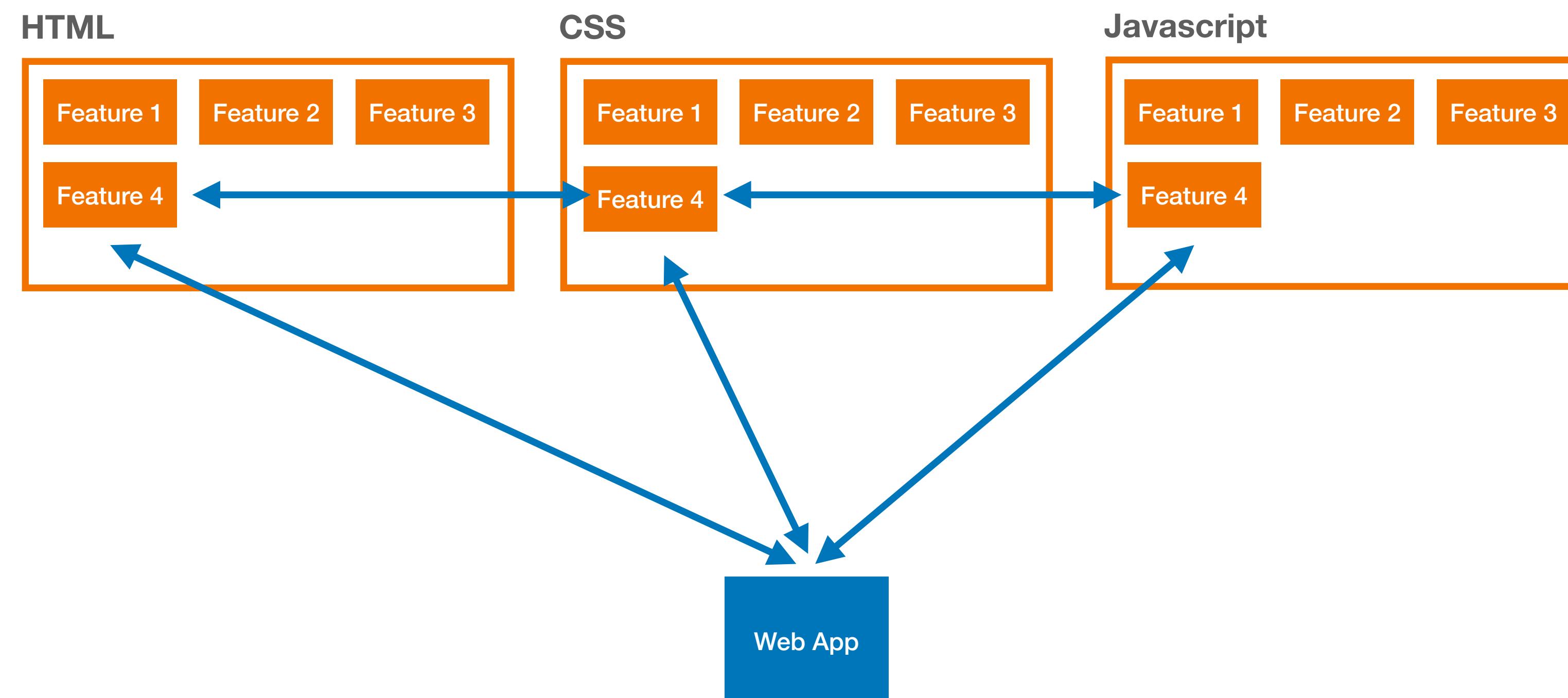
Standardisations des use cases courants

HTML / CSS / JS



Standardisations des use cases courants

HTML / CSS / JS



Polyfill

HTML / CSS / JS

- Un “Polyfill” est le terme générique donné à un bout de code permettant de combler une feature manquante dans un browser
- Permet la retrocompatibilité
- Le plus connu : Babel Polyfill

Standardisations des use cases courants

HTML / CSS / JS

- Validations de formulaires
- Lecteur audio/vidéo
- Eléments complexes (progress bar, ...) -> Shadow DOM !
- Sélectionner des éléments impaires
- Etc...

Nouveau standard HTML

HTML / CSS / JS

- Amélioration de HTML 4
- Fortement lié à Javascript
- Nouveaux éléments sémantiques (header, main, section, article, ...)
- Form inputs améliorés (types d'inputs, claviers mobiles, validations, transformations)
- APIs (géolocalisation, local storage, history, push, service workers, ...)
- Distribution automatique des éléments
- Custom elements

W3C vs WHATWG

HTML / CSS / JS

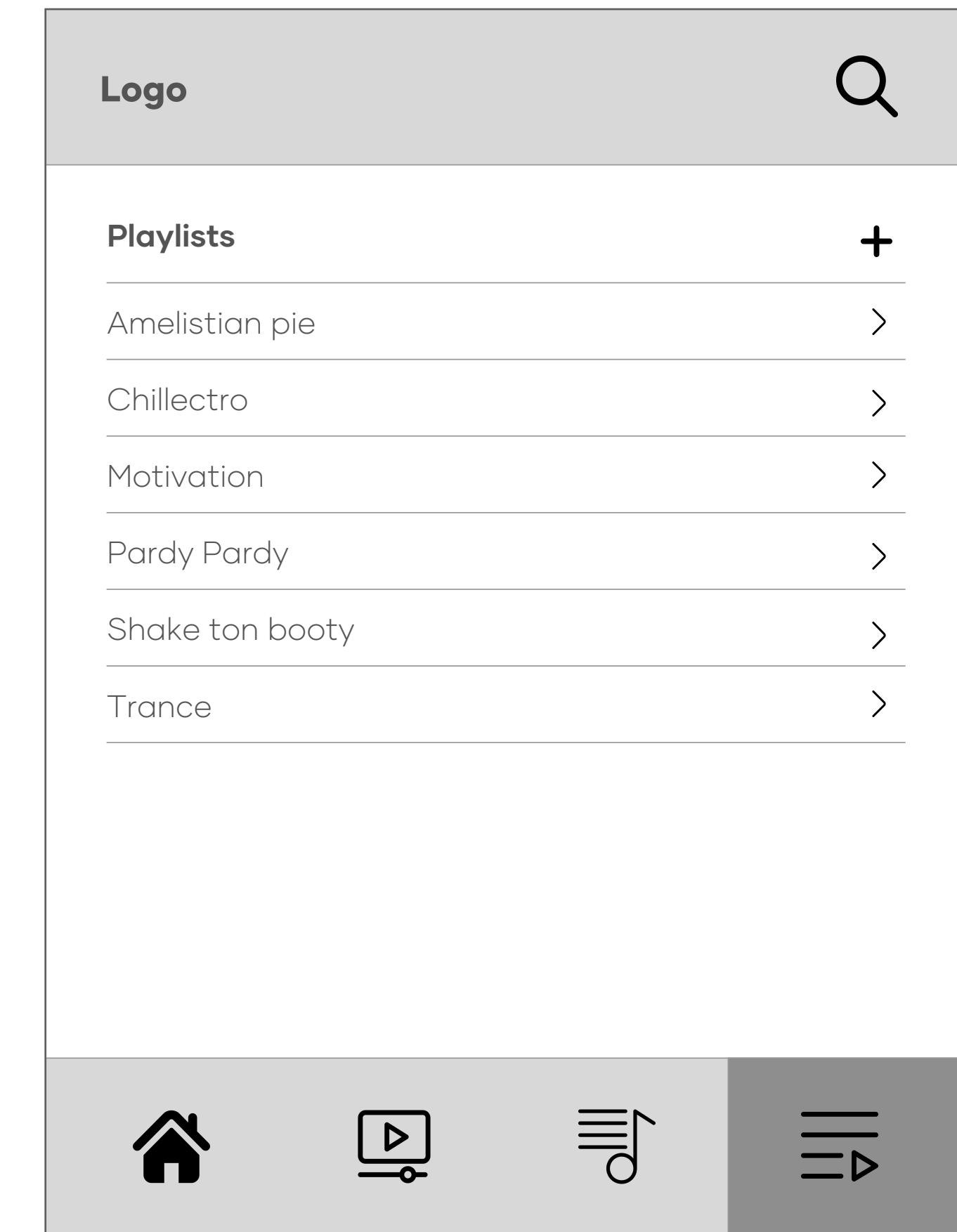
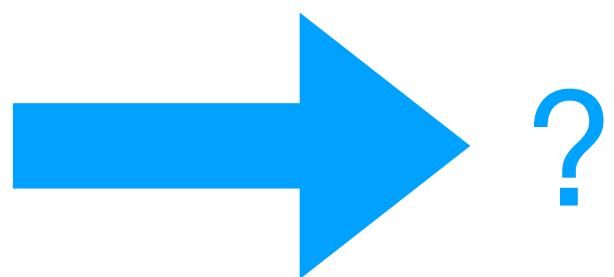
- W3C (**World Wide Web Consortium**)
Acteur historique...
- WHATWG (**Web Hypertext Application Technology Working Group**)
Collaboration non officielle des différents navigateurs web

HTML

Eléments sémantiques

HTML

```
<header />  
<nav />  
<main />  
<section />  
<footer />  
<article />
```



Custom elements

HTML

Here's the magic !



```
<my-super-custom-element />
```

Extended custom elements

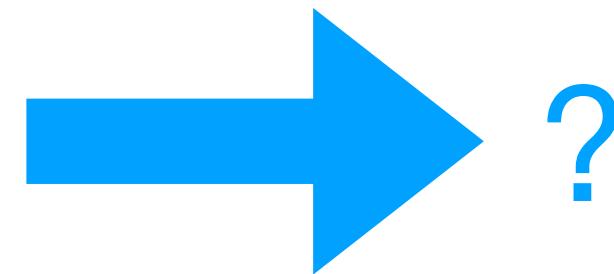
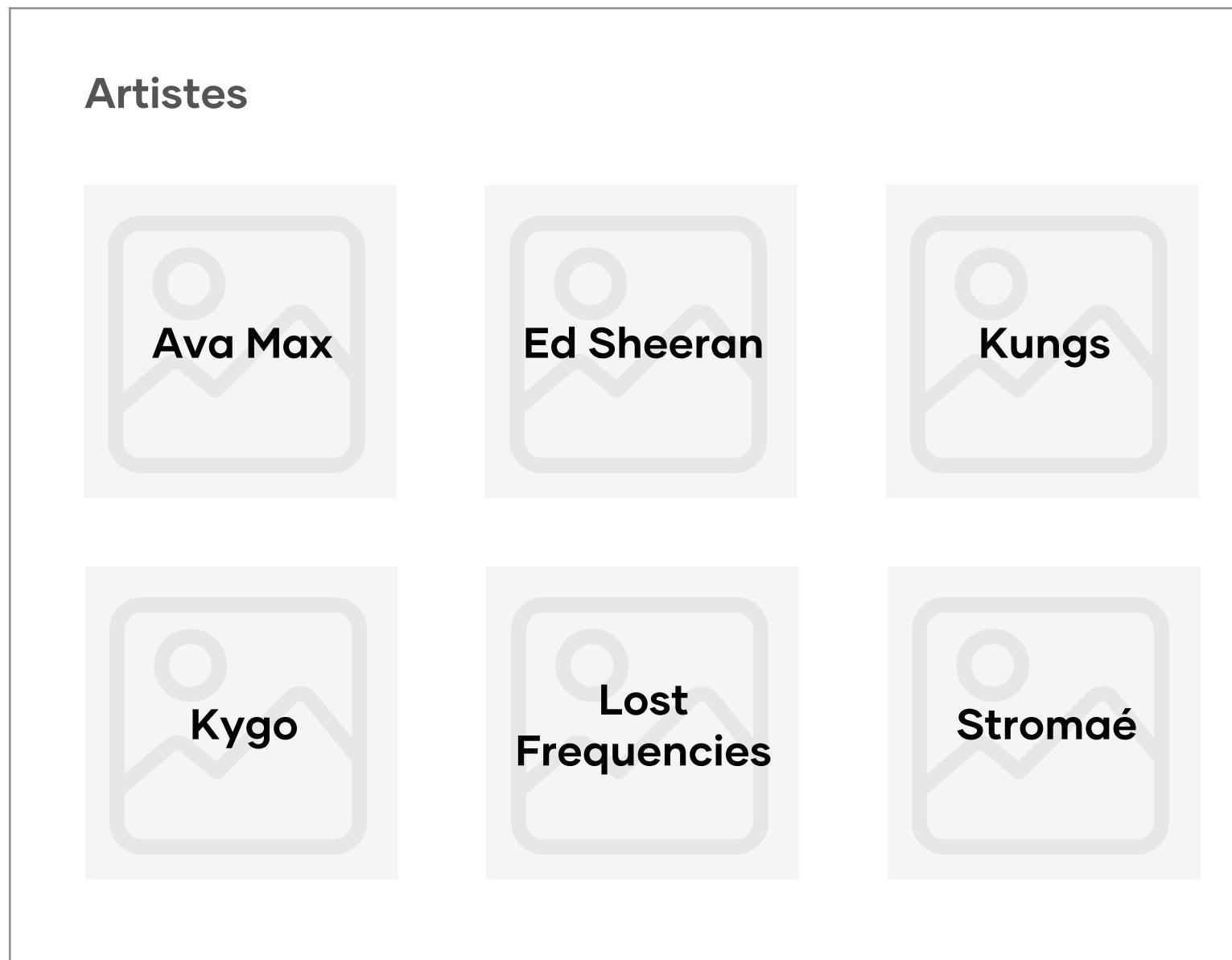
HTML

```
<mega-button />
```

```
const MegaButton = document.registerElement('mega-button', {  
  prototype: Object.create(HTMLButtonElement.prototype),  
  extends: 'button'  
});
```

Custom elements

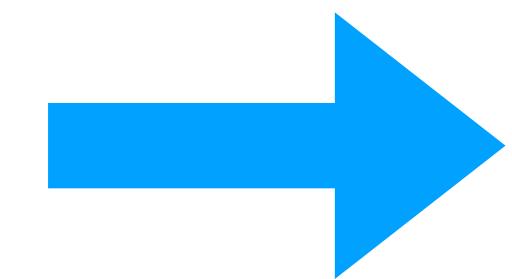
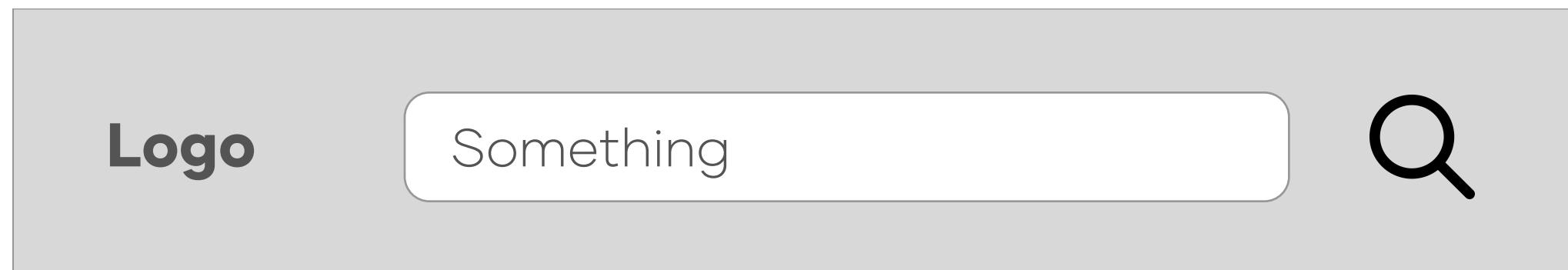
HTML



<artist-cover>
...
</artist-cover>

Inputs on steroids

HTML

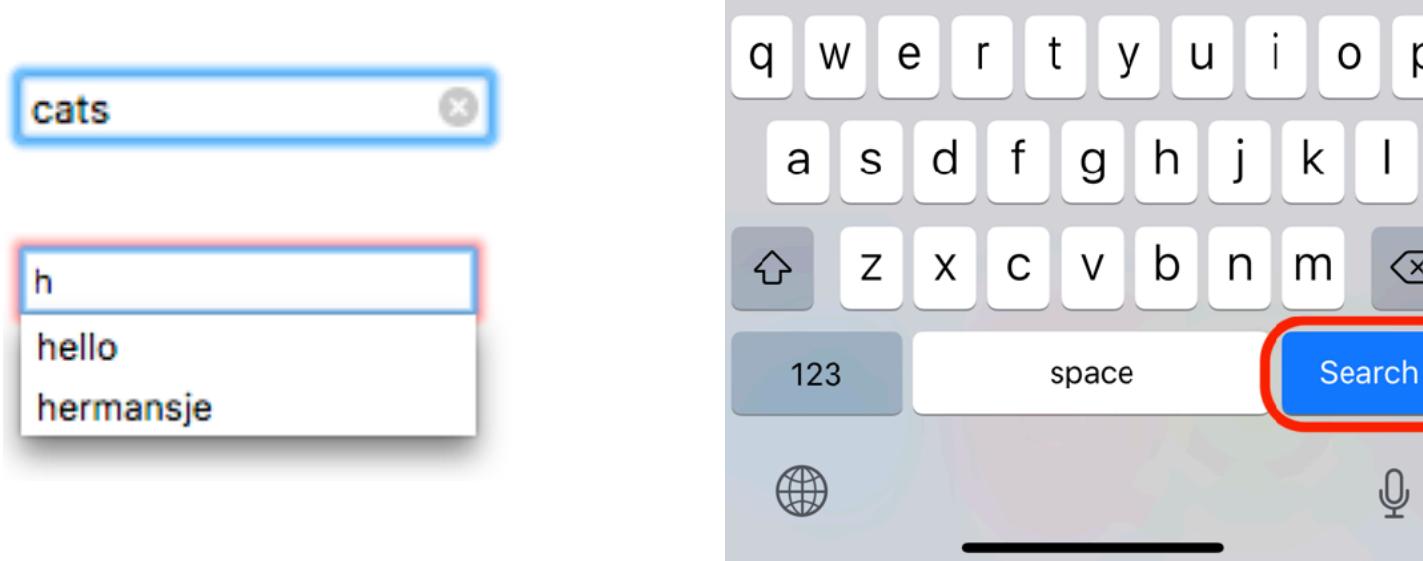


?

```
<input type="search" />
```

Inputs on steroids

HTML



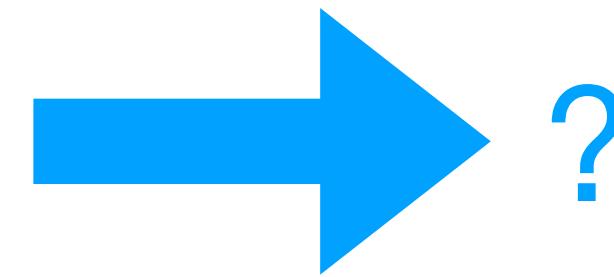
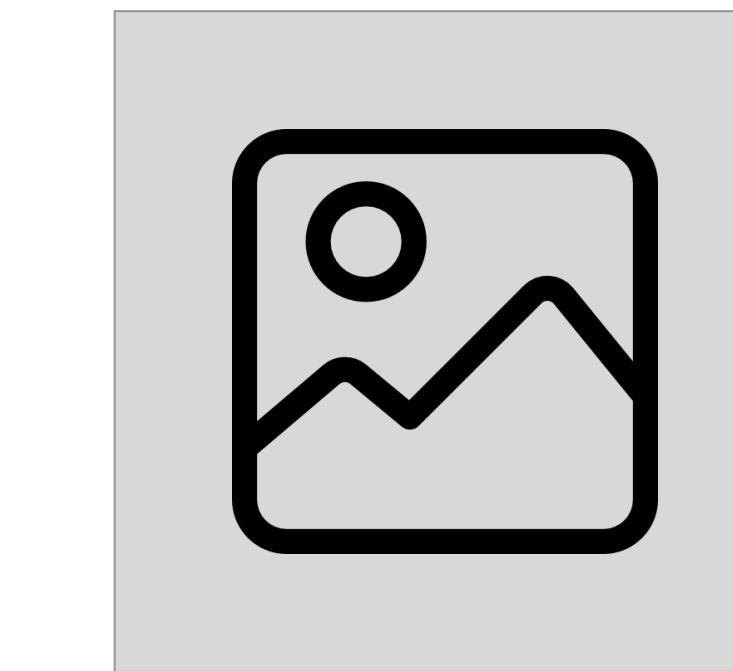
```
<input  
  type="search"  
  spellcheck="false"  
  autocapitalize="false"  
  autofocus  
/>
```



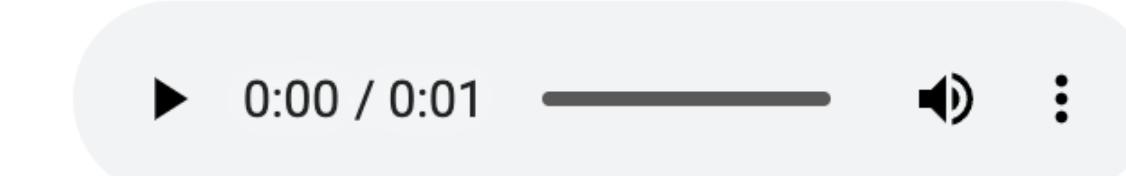
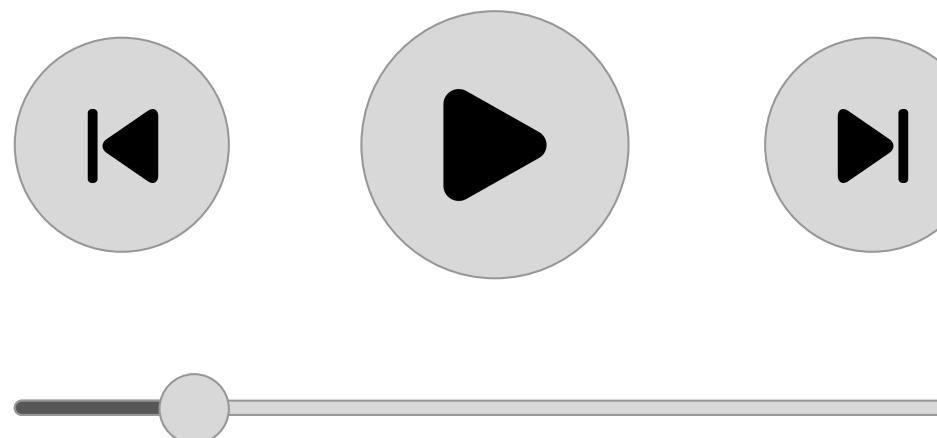
autofocus et autofocus="true" sont sémantiquement équivalents

Audio element

HTML



? <**audio** **src**=“<https://...>” />



Audio element

HTML

Presque...

```
<audio  
  id="audio-player"  
  src="https://..."  
/>
```

```
#audio-player {  
  display: none;  
}
```

Shadow DOM

HTML and a bit more...

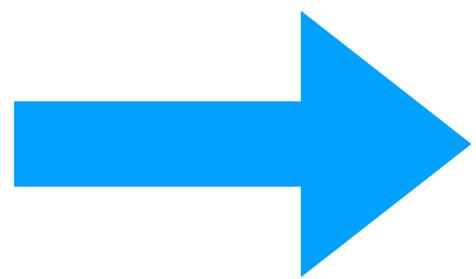
- “Black box”
- Encapsulation et abstraction du DOM contenu dans un élément
- Principalement utilisé par les browsers pour abstraire des éléments complexes
- Exemple: <**audio** />



Shadow DOM

HTML and a bit more...

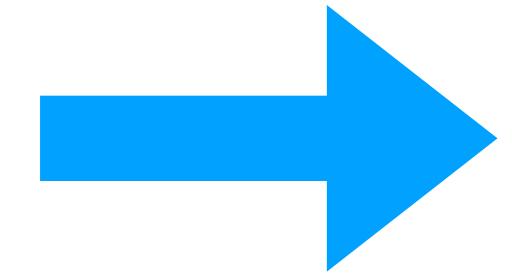
```
<!DOCTYPE html>
<html>
  ><head>...</head>
  ><body>
    .. ><audio src="http://...."></audio> == $0
  </body>
</html>
```



```
<!DOCTYPE html>
<html>
  ><head>...</head>
  ><body>
    .. ><audio src="http://...."> == $0
      >#shadow-root (user-agent)
        ><div pseudo="-webkit-media-controls" class="phase-pre-ready state-no-source">flex
          ><div pseudo="-webkit-media-controls-overlay-enclosure">
            ><input pseudo="-internal-media-controls-overlay-cast-button" type="button" aria-label="play on remote device" style="display: none;">
              >#shadow-root (user-agent)
                </input>
              </div>
            ><div pseudo="-webkit-media-controls-enclosure">flex
              ><div pseudo="-webkit-media-controls-panel" style="display: none;">
                ><input type="button" pseudo="-webkit-media-controls-play-button" aria-label="play" class="pause" disabled style="display: none;">...
                  ><div aria-label="elapsed time: 0:00" pseudo="-webkit-media-controls-current-time-display" style="display: none;">0:00</div>
                  ><div aria-label="total time: / 0:00" pseudo="-webkit-media-controls-time-remaining-display" style="display: none;">/ 0:00</div>
                ><input type="range" step="any" pseudo="-webkit-media-controls-timeline" max="NaN" min="0" aria-label="audio time scrubber 0:00 / 0:00" aria-valuetext="elapsed time: 0:00" disabled>...
                  ><div pseudo="-webkit-media-controls-volume-control-container" class="closed" style="display: none;">...
                    ><input type="button" pseudo="-webkit-media-controls-fullscreen-button" aria-label="enter full screen" style="display: none;">...
                    ><input type="button" aria-label="show more media controls" title="more options" pseudo="-internal-media-controls-overflow-button" style="display: none;">...
                  </div>
                ><div role="menu" aria-label="Options" pseudo="-internal-media-controls-text-track-list" style="display: none;">...
                  ><div role="menu" aria-label="Options" pseudo="-internal-media-controls-playback-speed-list" style="display: none;">...
                    ><div pseudo="-internal-media-controls-overflow-menu-list" role="menu" class="closed" style="display: none;">
                      ><label pseudo="-internal-media-controls-overflow-menu-list-item" role="menuitem" tabindex="0" aria-label="Play " class="animated-1" style="display: none;">...
                        ><label pseudo="-internal-media-controls-overflow-menu-list-item" role="menuitem" tabindex="0" aria-label="enter full screen Full screen " style="display: none;">...
                          ><label pseudo="-internal-media-controls-overflow-menu-list-item" role="menuitem" tabindex="0" aria-label="download media Download " style="display: none;">...
                            ><label pseudo="-internal-media-controls-overflow-menu-list-item" role="menuitem" tabindex="0" aria-label="Mute " style="display: none;">...
                            ><label pseudo="-internal-media-controls-overflow-menu-list-item" role="menuitem" tabindex="0" aria-label="play on remote device Cast " style="display: none;">...
                            ><label pseudo="-internal-media-controls-overflow-menu-list-item" role="menuitem" tabindex="0" aria-label="show closed captions menu Captions " style="display: none;">...
                            ><label pseudo="-internal-media-controls-overflow-menu-list-item" role="menuitem" tabindex="0" aria-label="show playback speed menu Playback speed " class="animated-0" style="display: none;">...
                          </div>
                        </div>
                      </div>
                    </div>
                  </div>
                </div>
              </div>
            </div>
          </div>
        </div>
      </div>
    </div>
  </body>
</html>
```

Progress v1

HTML

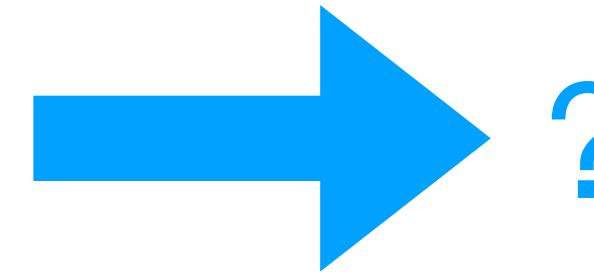


?

```
<progress value="20" max="100" />
```

Progress v2

HTML



```
<input  
    type="range"  
    value="20"  
    min="0"  
    max="100"  
/>
```

Data attributes

HTML

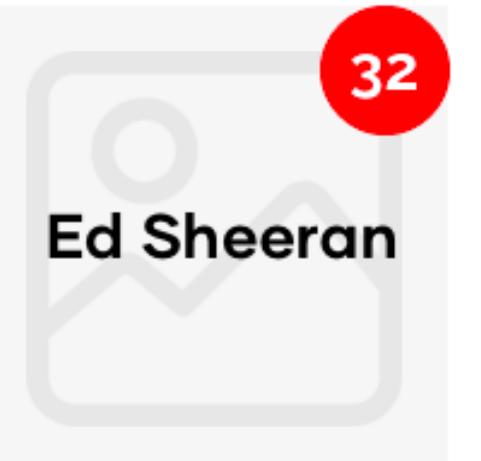
```
<artist-cover  
  data-id="1"  
  data-song-count="32"  
  data-thumbnail-url="https://..."  
>  
  Ed Sheeran  
</artist-cover>
```

Data attributes

HTML and a bit more...

```
<artist-cover  
  data-id="1"  
  data-song-count="32"  
  data-thumbnail-url="https://...">  
  Ed Sheeran  
</artist-cover>
```

```
artist-cover {  
  /*_/_!\ Bientôt..._*/  
  background-image: attr(data-image url);  
}  
  
artist-cover:before {  
  /*_/_Possible! Content only _*/  
  content: attr(data-song-count);  
  position: absolute;  
  top: 0;  
  right: 0;  
  border-radius: 50%;  
  background-color: #ff0000;  
  color: #fff;  
  text-align: center;  
}
```



Limitations...

HTML and a bit more...

```
<input type="date" />
```

Code

css

A dramatic scene of a volcanic eruption at night. A large, dark mountain peak is erupting, with bright orange and yellow lava flows cascading down its sides and filling the foreground. The sky is filled with dark, billowing smoke and ash. In the foreground, several human skeletons are scattered across the lava field, their arms raised as if in despair or warning. The overall atmosphere is one of destruction and hellish torment.

Here be dragons...

Browser defaults

CSS

- Chaque browser dispose d'une feuille de style interne par défaut
- Sensé être normé et cross-browser compatible
- En réalité... Pas vraiment.
- Utilisation de feuilles de style CSS reset

CSS reset/reboot/normalize/younameit css

- Permet de charger des styles par défaut consistants
- Assurance d'une compatibilité cross-browser à 100%
- Pléthore de versions online
- La plus répandue <https://necolas.github.io/normalize.css/>
(utilisée par Bootstrap, Twitter, GitHub, ...)

CSS icons

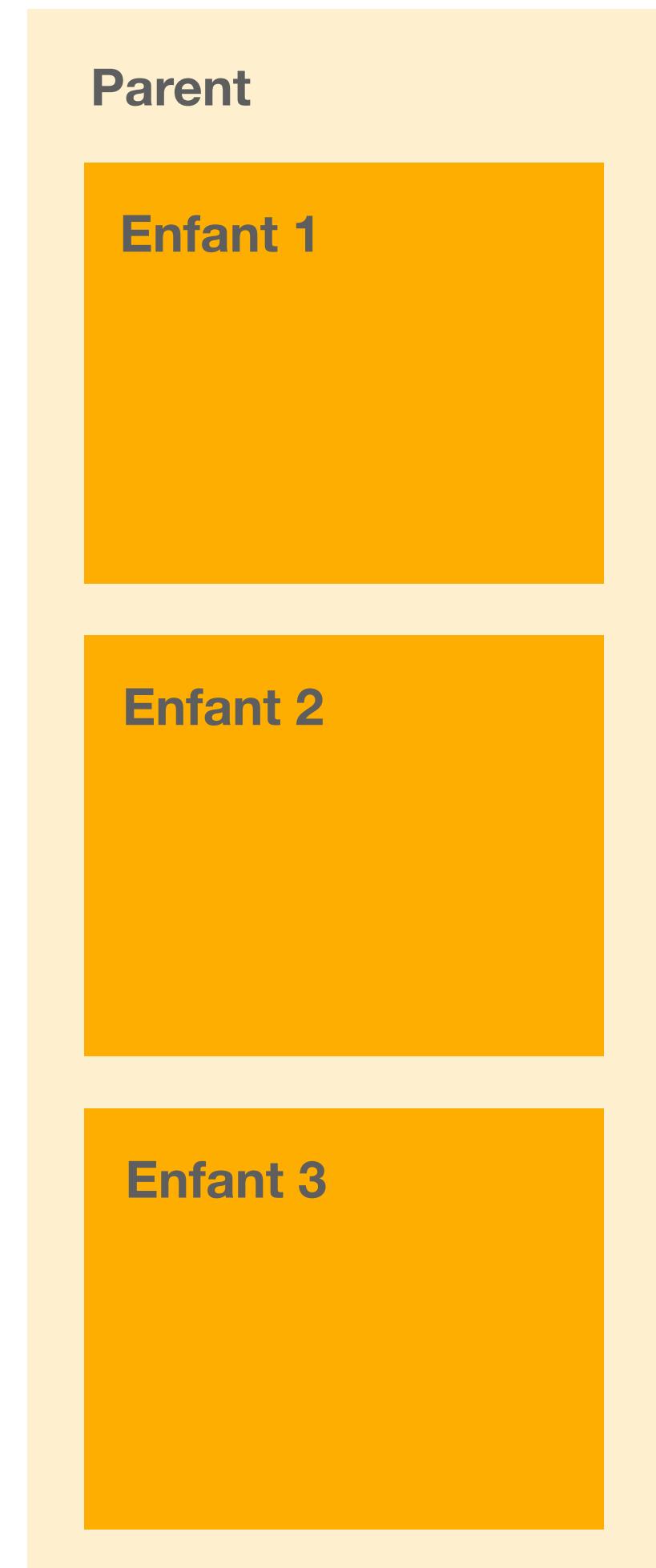
css

- Actuellement Google Material icons
- Intégré en mode CDN
- <https://fonts.google.com/icons>
- Exemple : <face>
- Libre à vous !

Flexboxes

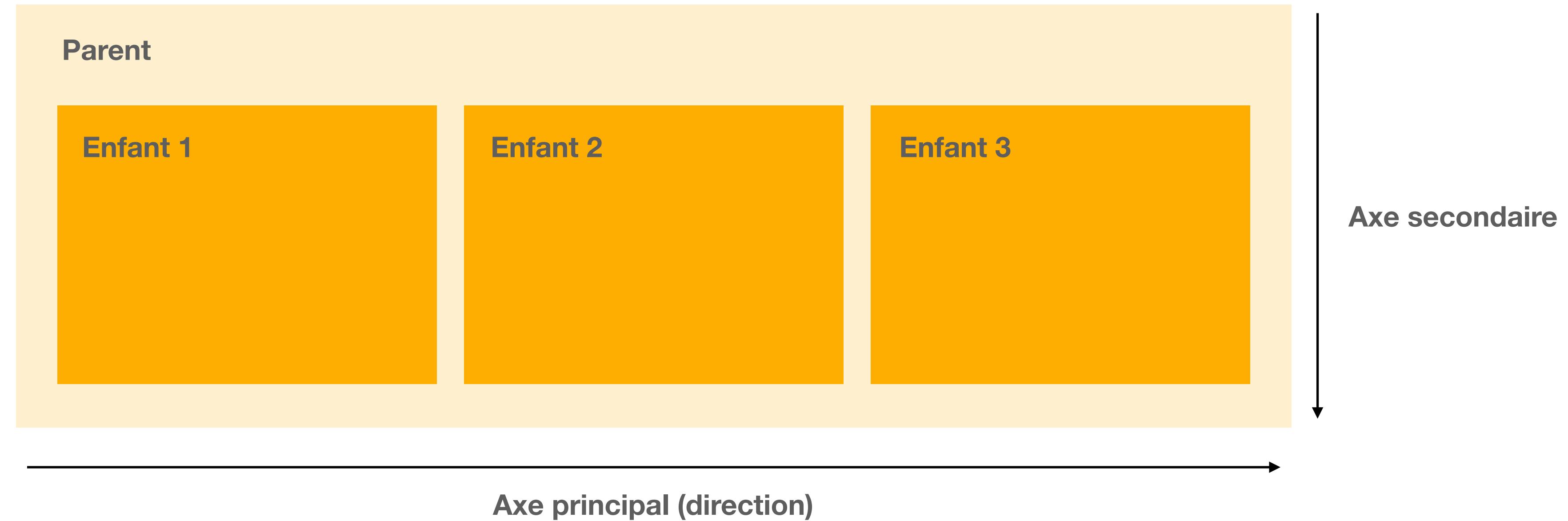
CSS

- Permet de distribuer l'espace entre des éléments, au sein d'un parent
- Unidimensionnel - Ligne ou colonne
- Gère également l'alignement et le dimensionnement
- Comparable à des “float” gérées par le parent
- Très fortement lié au concept de “responsive design”



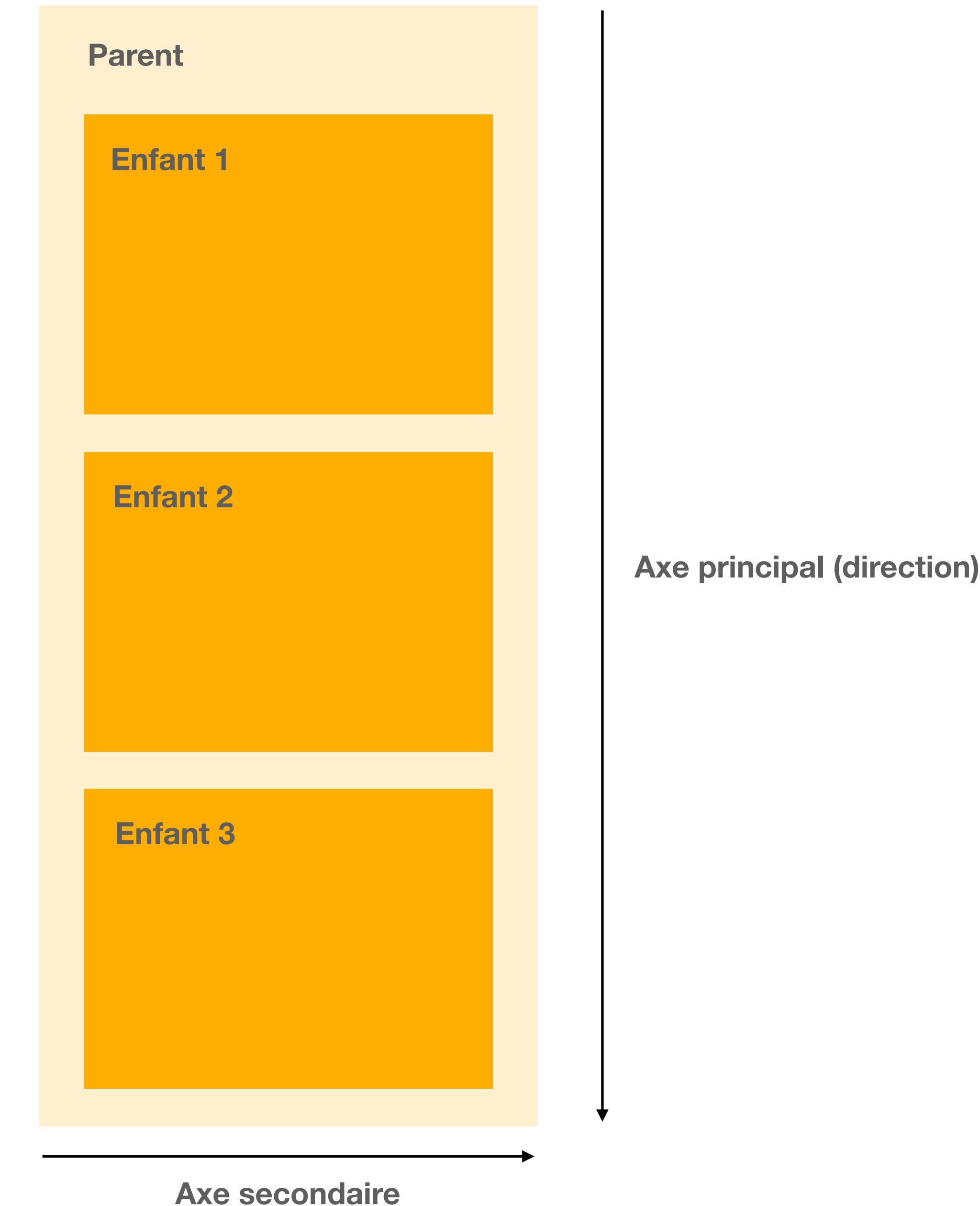
Flexboxes

CSS



Flexboxes

CSS



Flexboxes - Propriétés CSS

Propriétés du parent

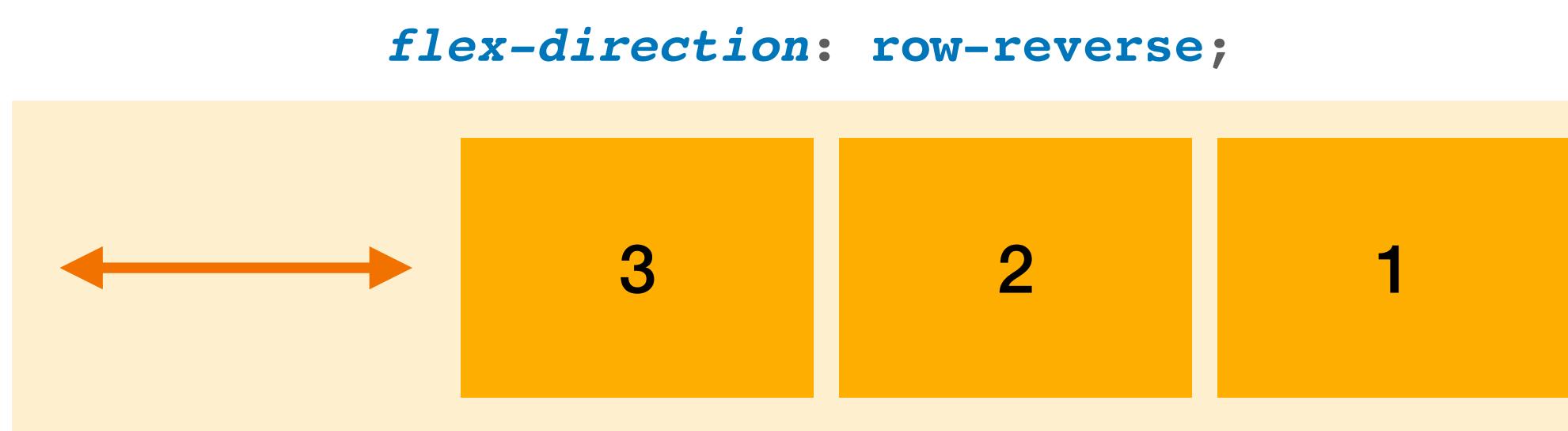
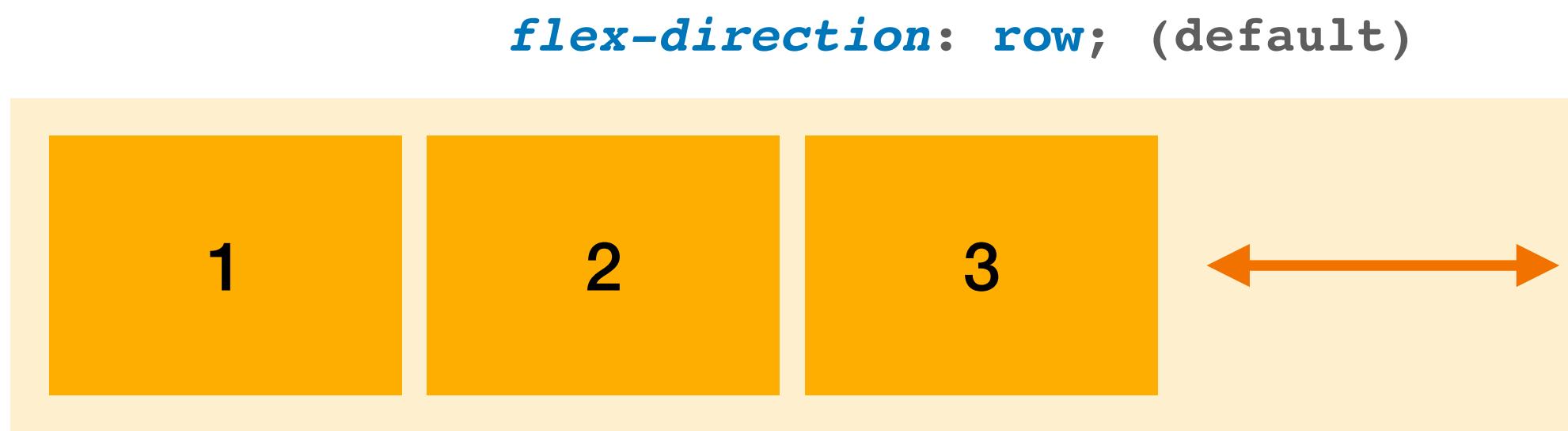
- Direction (`flex-direction`)
- Alignement axe principal (`justify-content`)
- Alignement axe secondaire (`align-items`)
- Multi-lignes (`flex-wrap`)
- Alignement multi-lignes (`align-content`)
- Ecartement (`gap`)

Propriétés des enfants

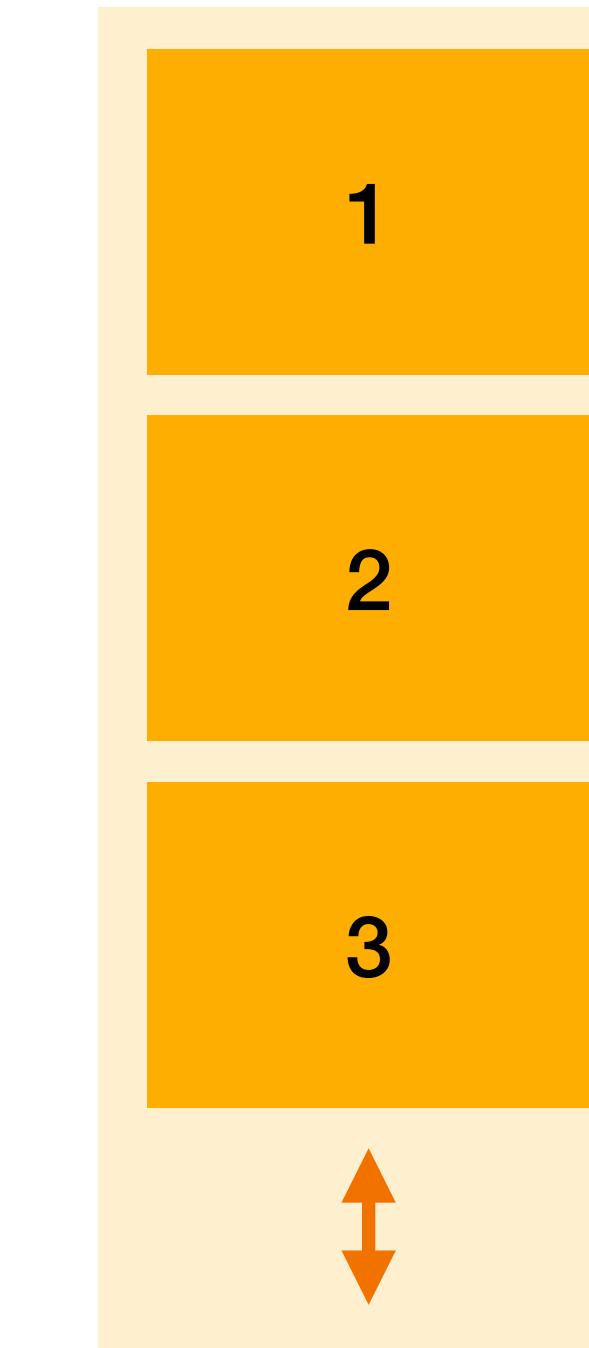
- Taille de base (`flex-basis`)
- Taille du rétrécissement (`flex-shrink`)
- Taille de l'agrandissement (`flex-grow`)
- Overrides des propriétés du parent (`...-self`)

Flexboxes - Direction (flex-direction)

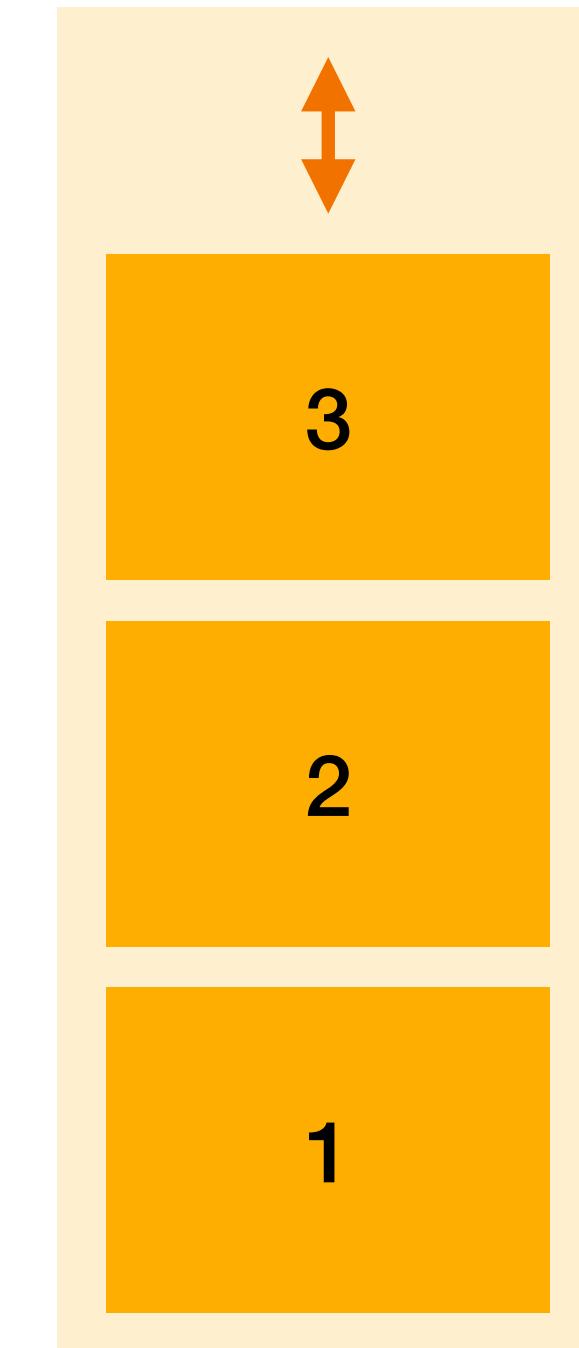
CSS



flex-direction: column;



flex-direction: column-reverse;

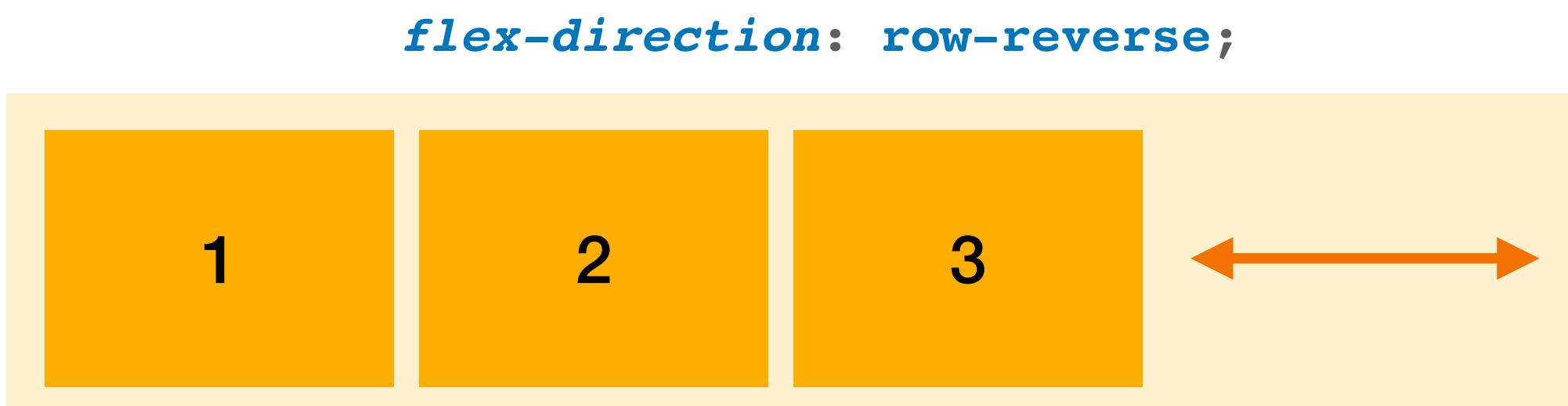
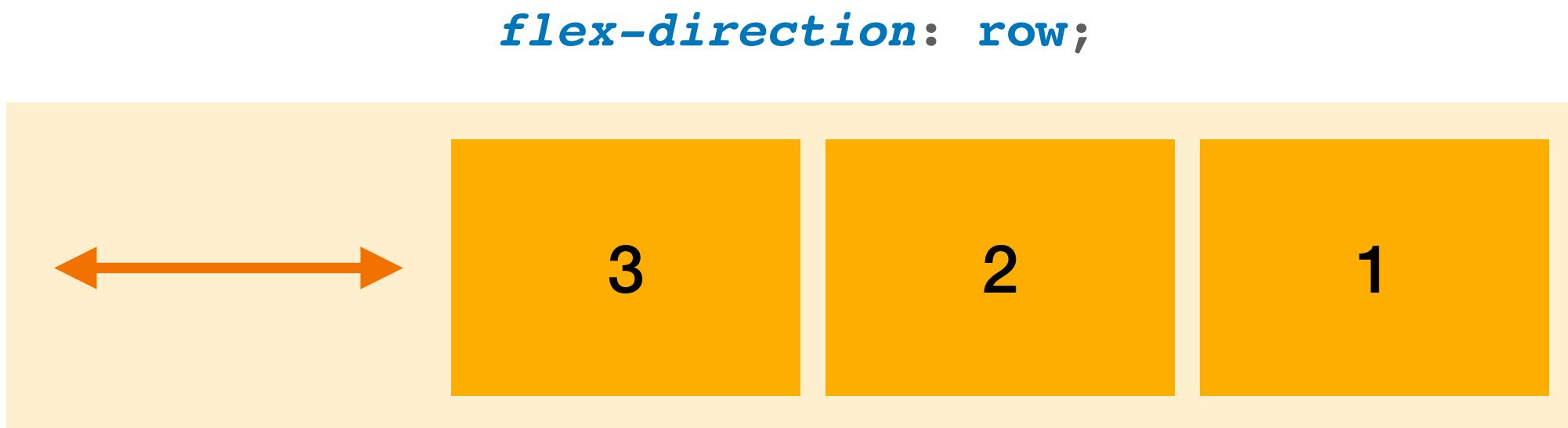


Flexboxes - Direction rtl

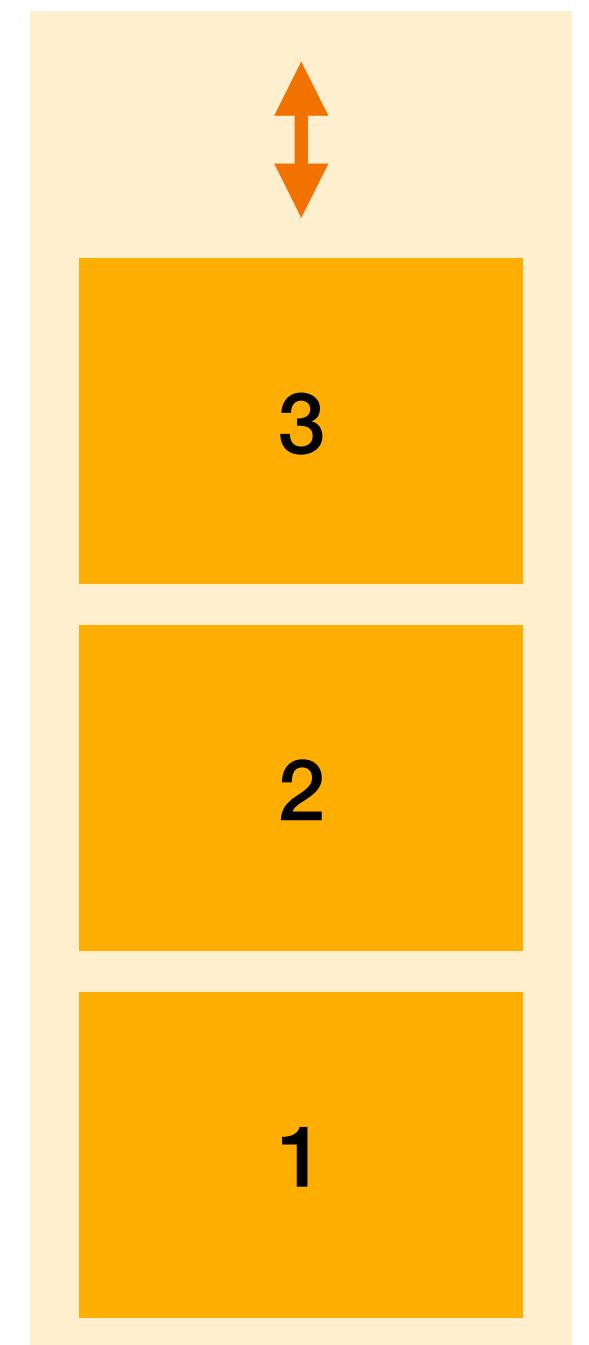
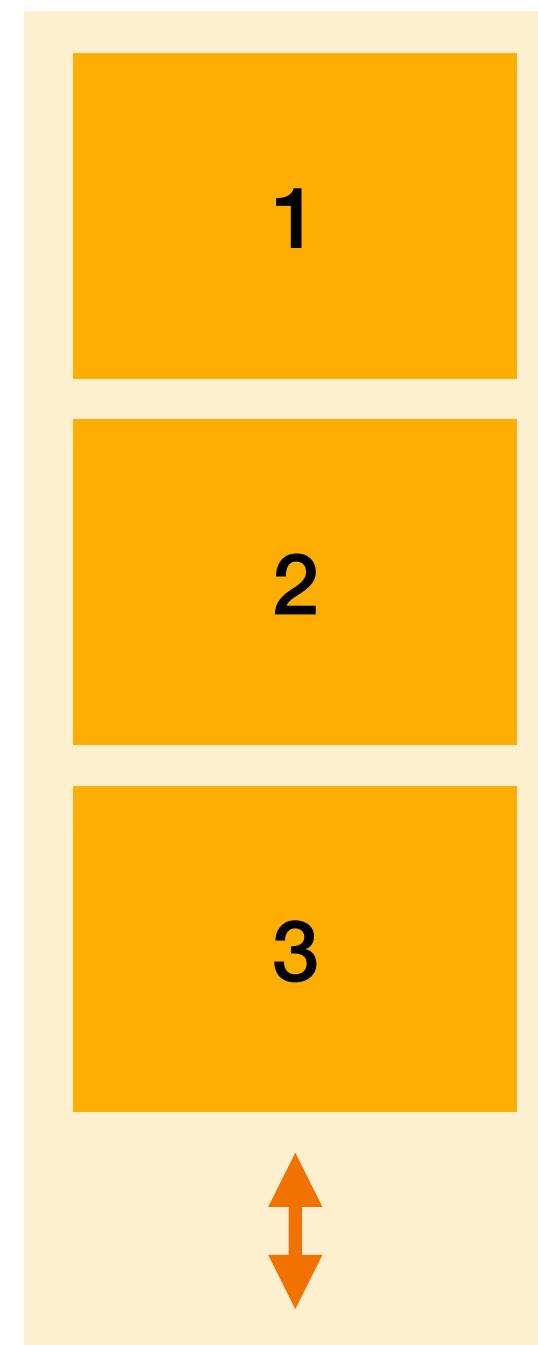
CSS



En écriture droite-gauche (arabe) -> `direction: rtl;`



`flex-direction: column;` `flex-direction: column-reverse;`



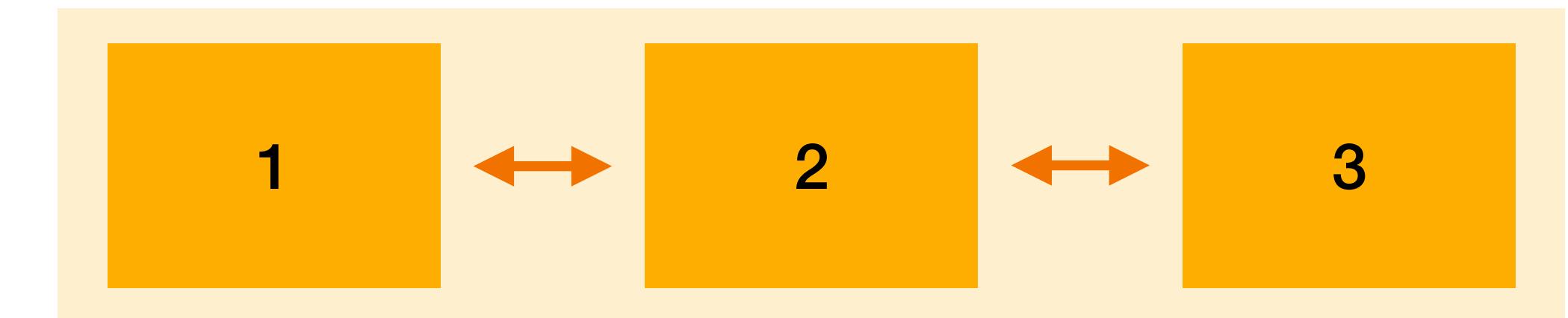
Flexboxes - Alignement axe principal (justify-content)

CSS

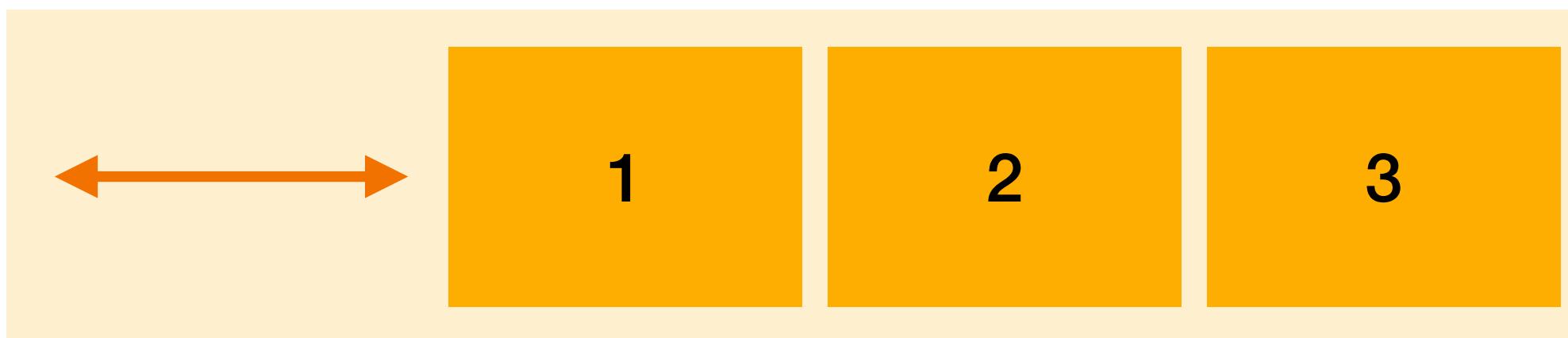
justify-content: flex-start; (default)



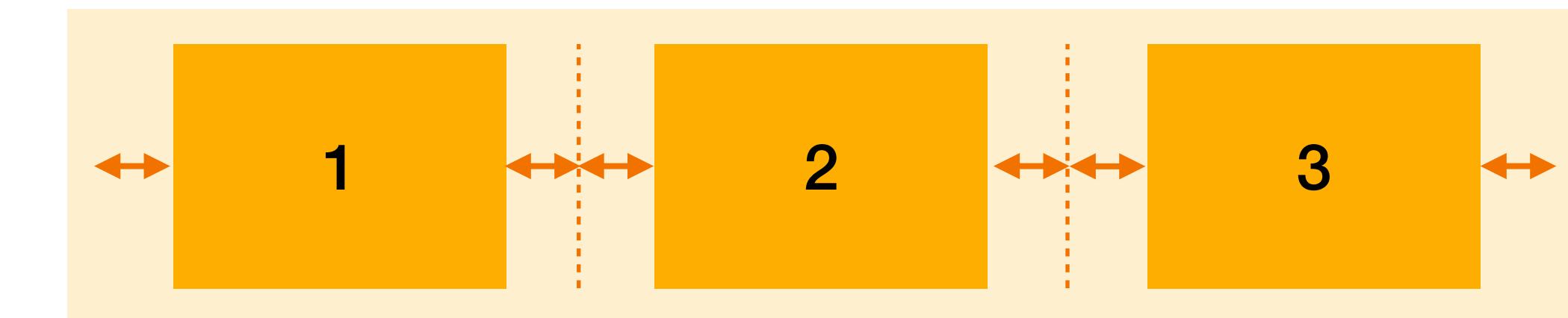
justify-content: space-between;



justify-content: flex-end;



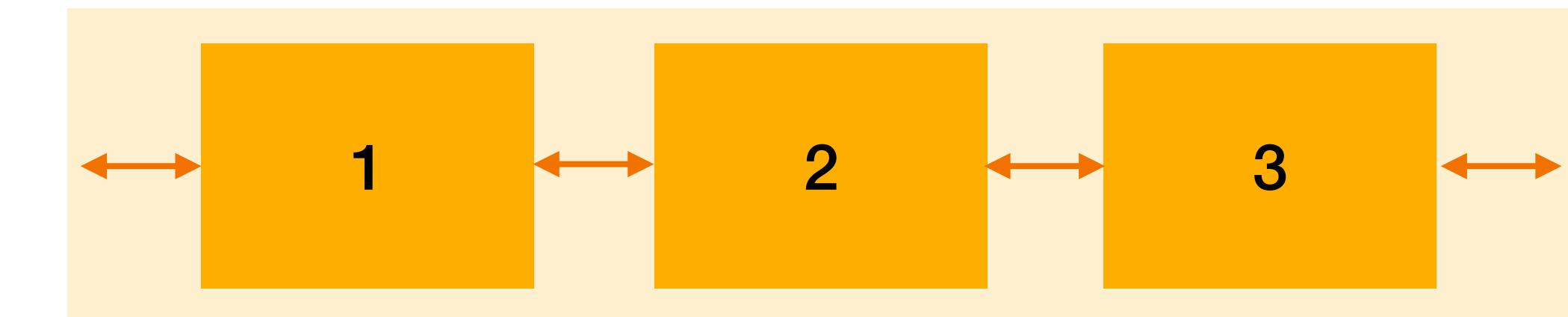
justify-content: space-around;



justify-content: center;



justify-content: space-evenly;



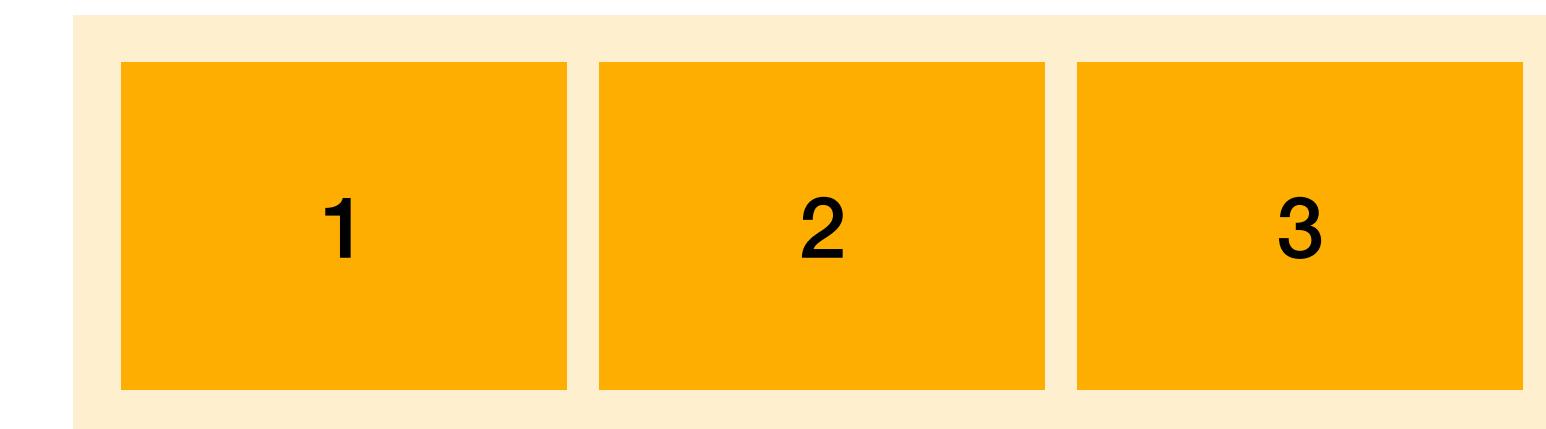
Flexboxes - Alignement axe secondaire (align-items)

CSS

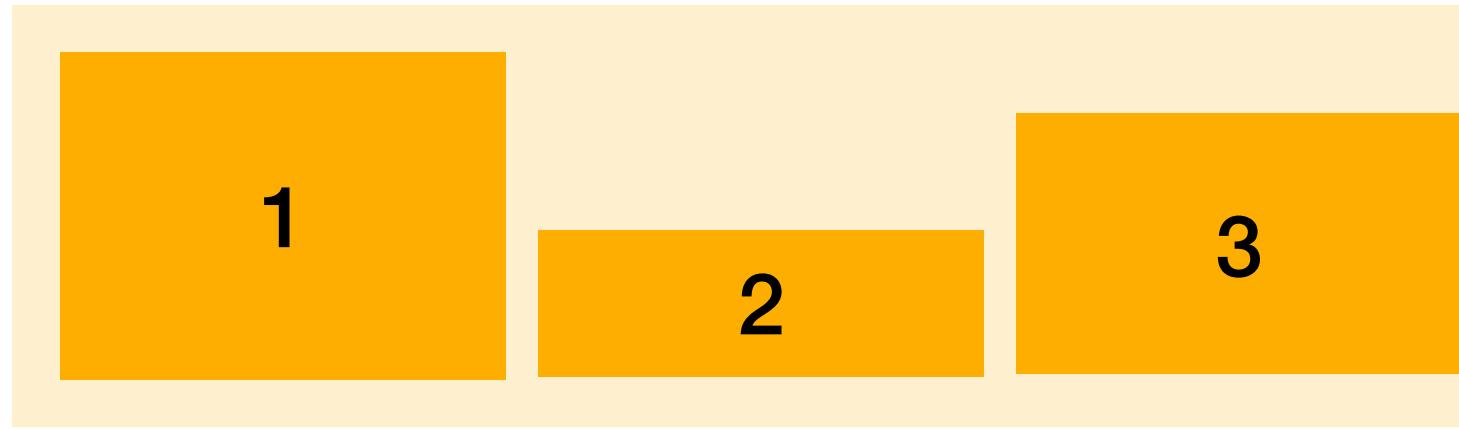
`align-items: flex-start;`



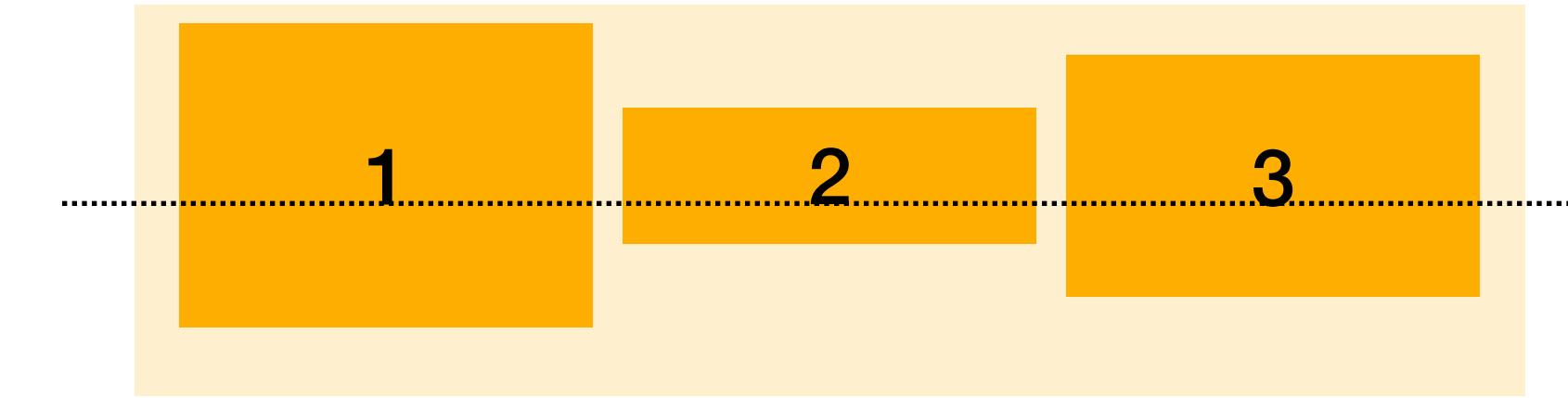
`align-items: stretch; (default)`



`align-items: flex-end;`



`align-items: baseline;`



`align-items: center;`



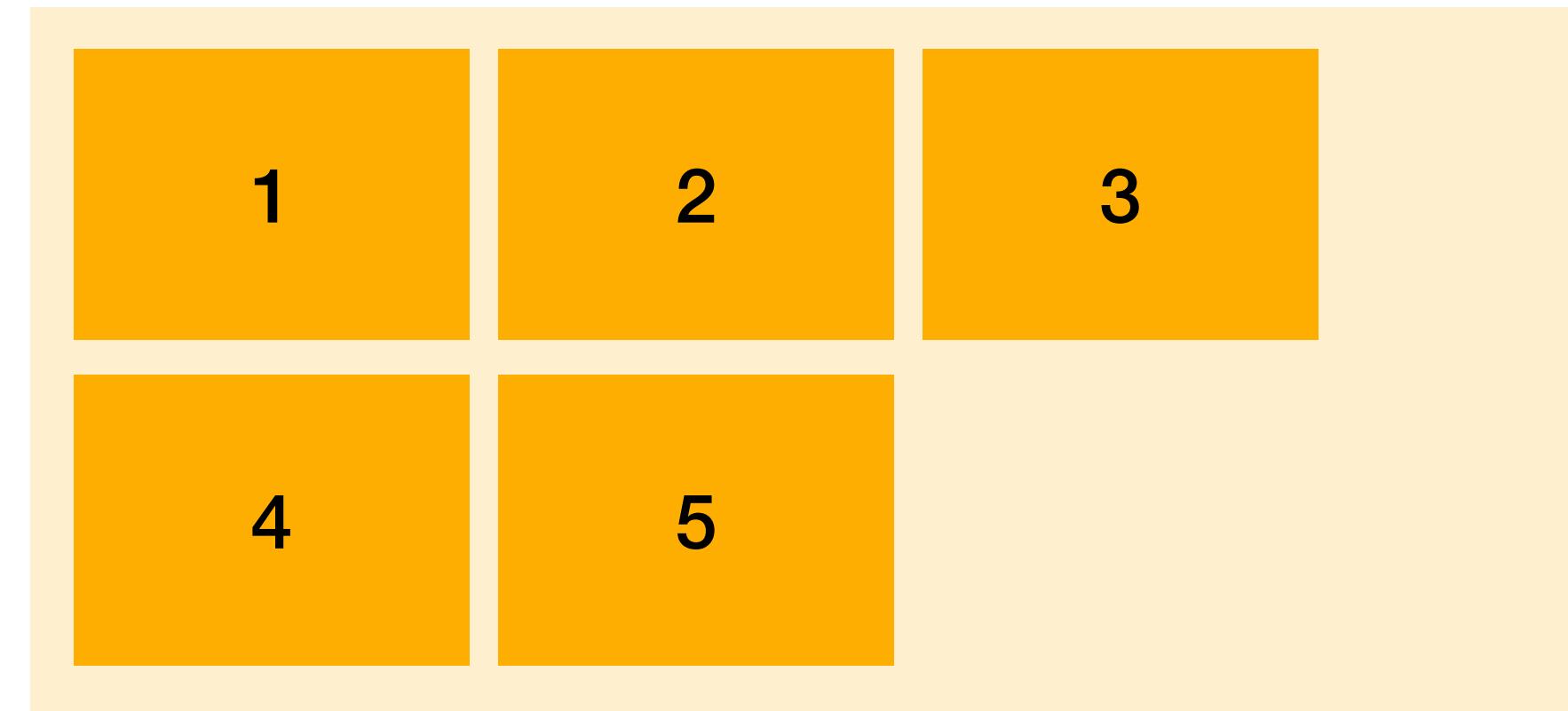
Flexboxes - Multi-lignes (wrap)

CSS

`flex-wrap: nowrap; (default)`



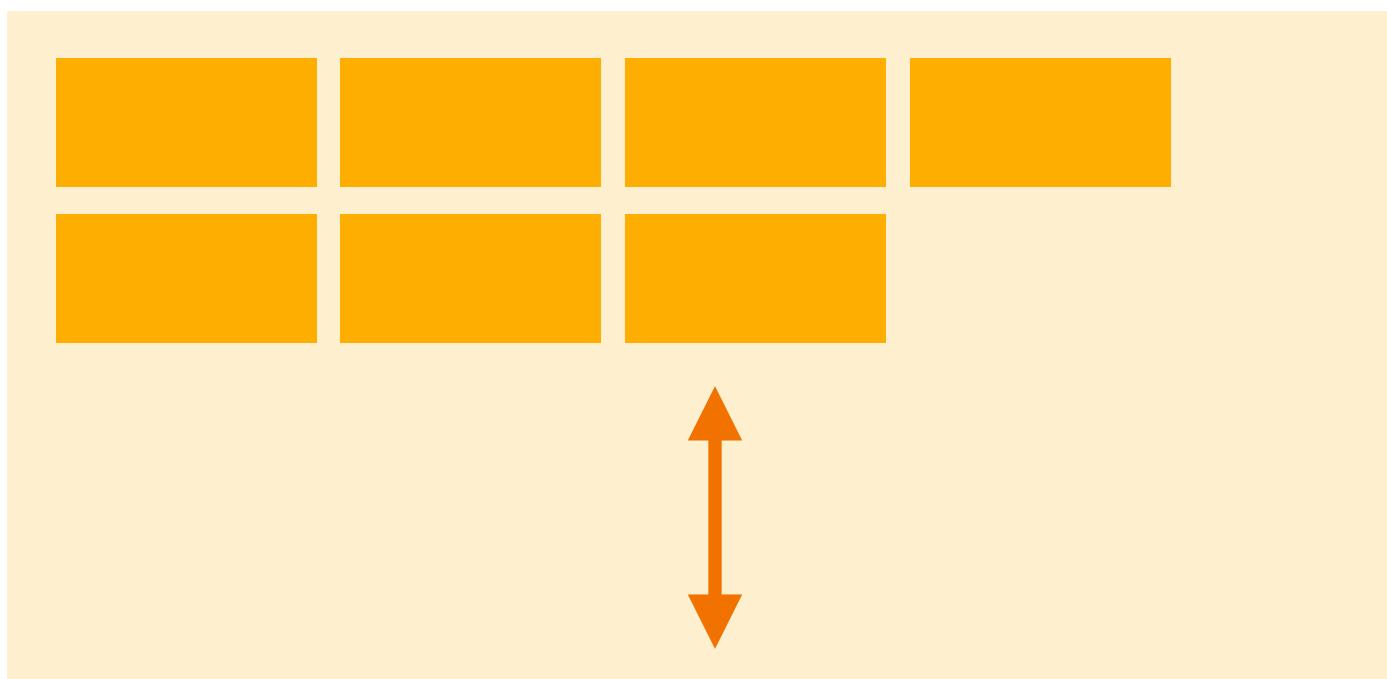
`flex-wrap: wrap;`



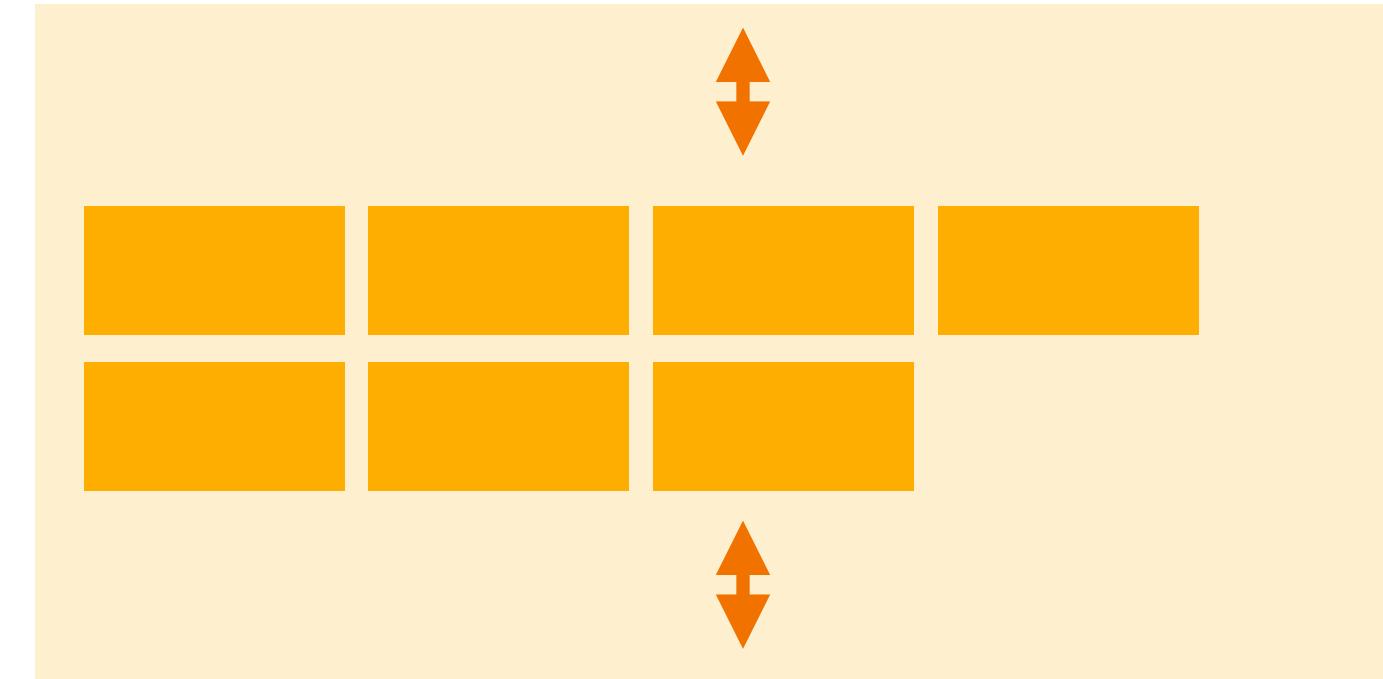
Flexboxes - Alignement multi-lignes (align-content)

CSS

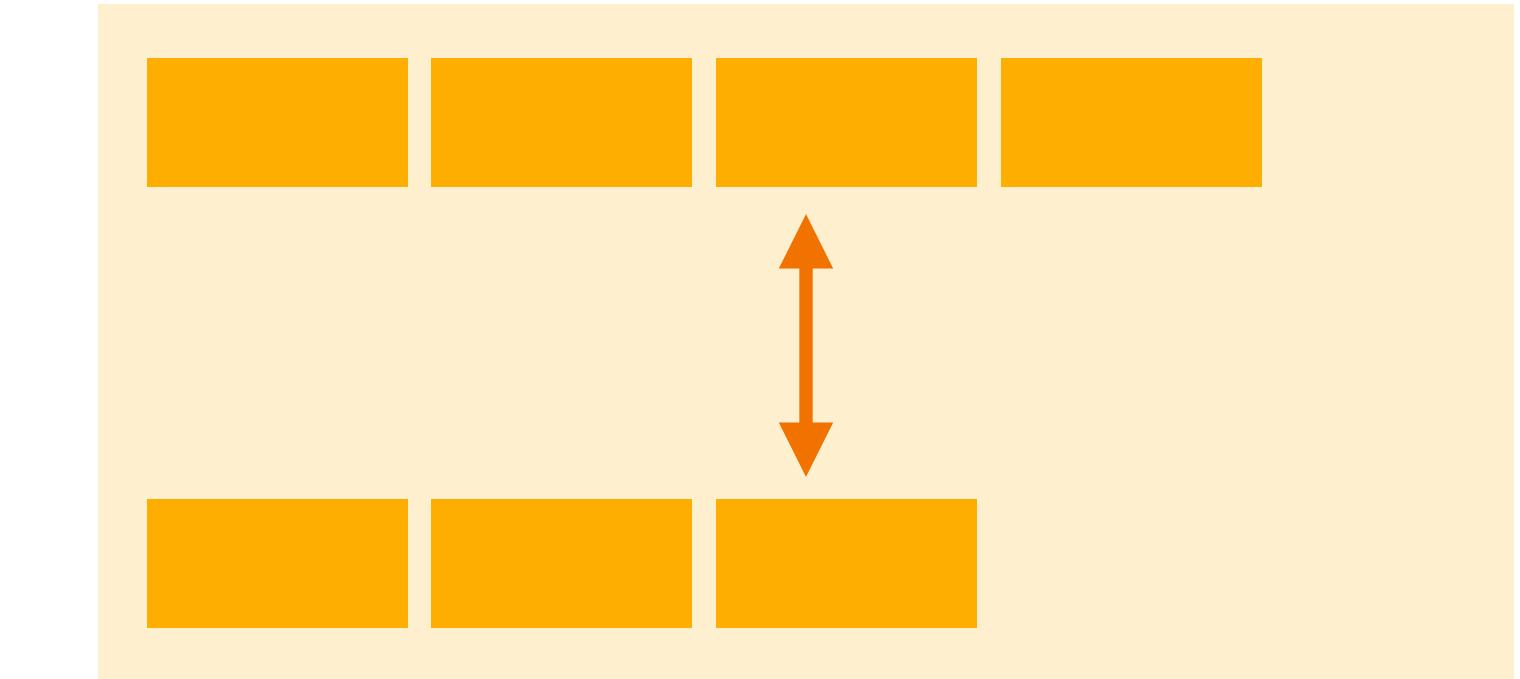
align-content: flex-start;



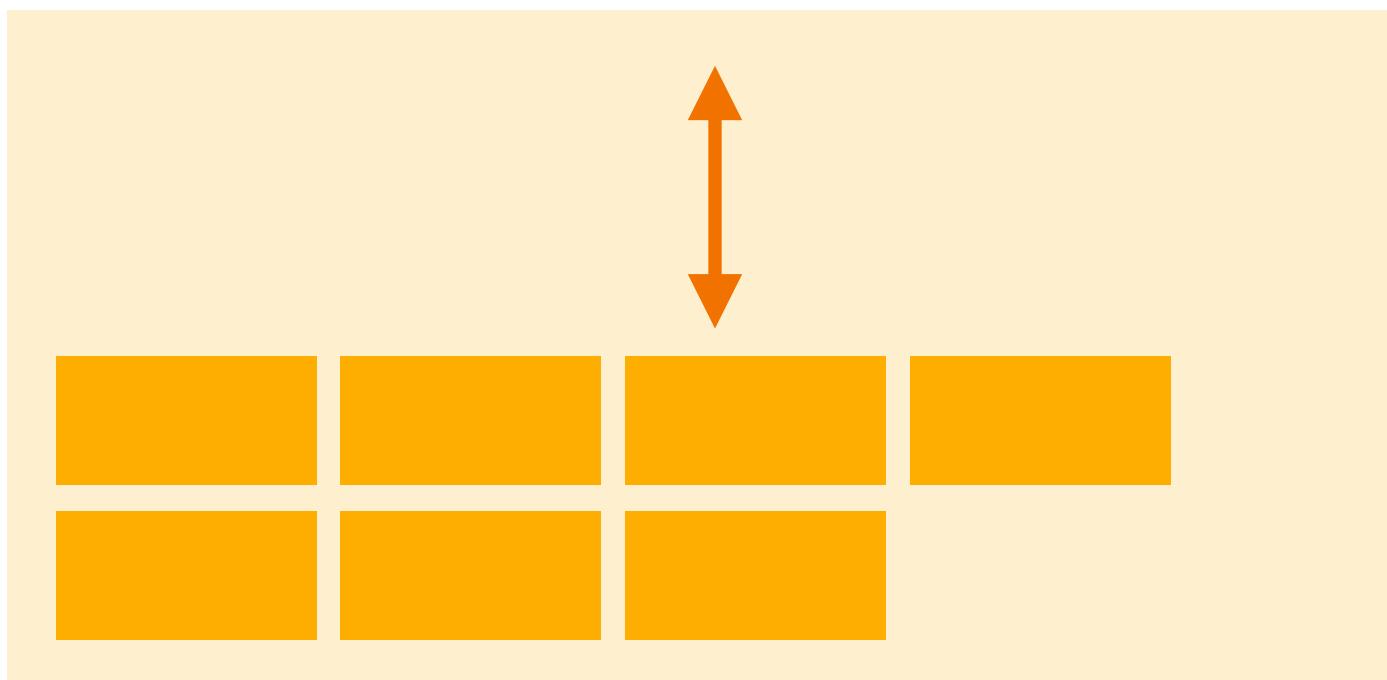
align-content: center;



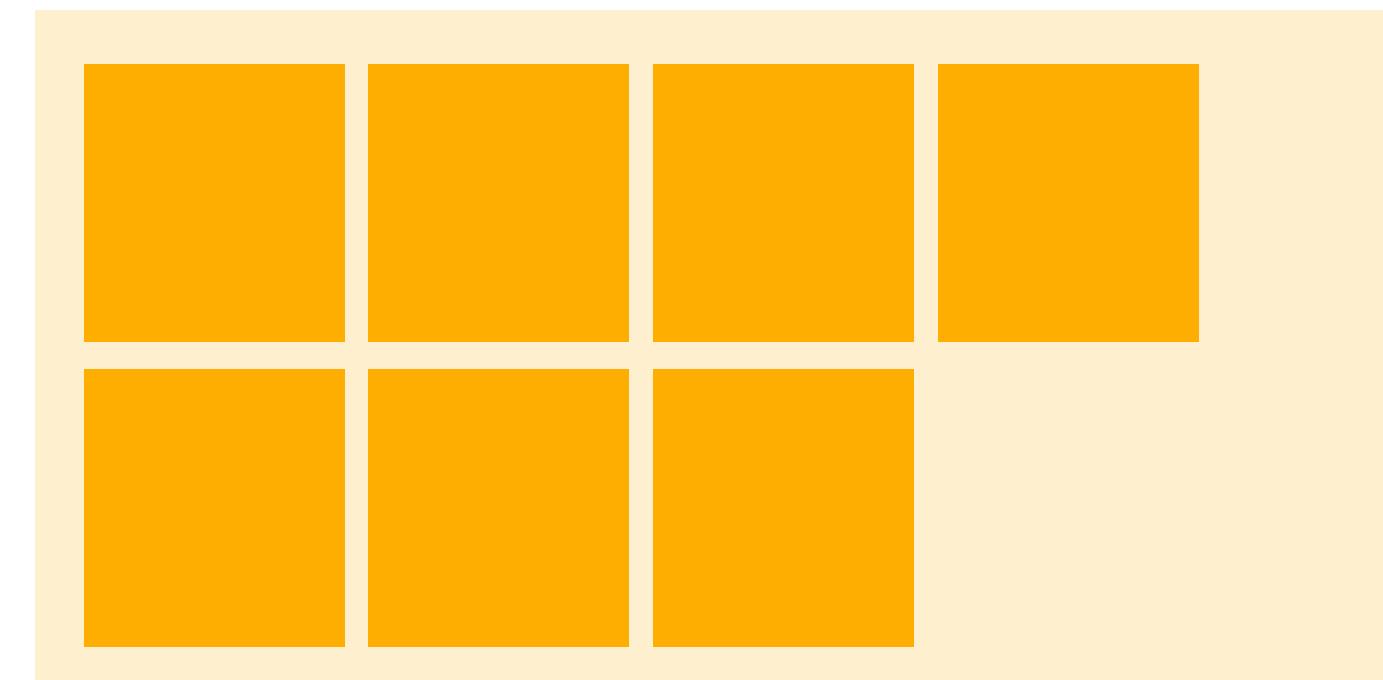
align-content: space-between;



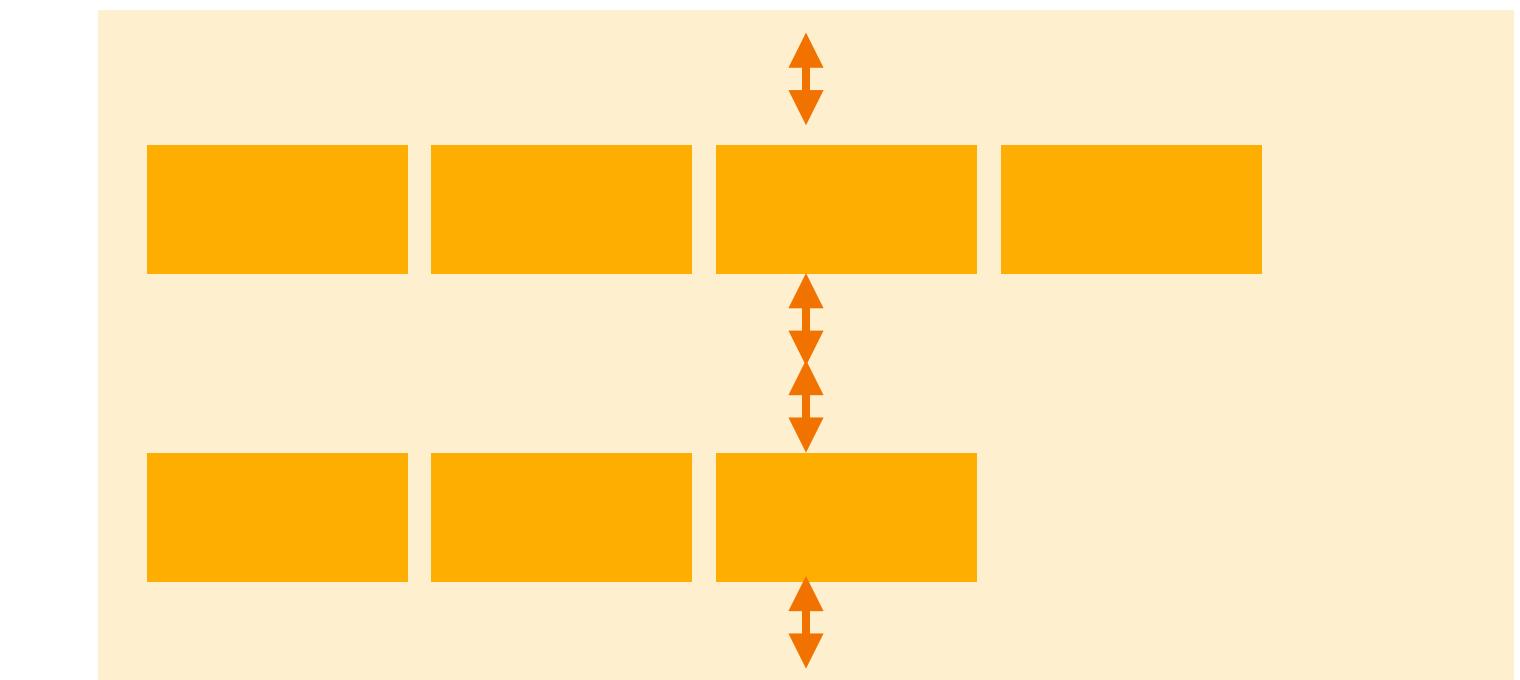
align-content: flex-end;



align-content: stretch;

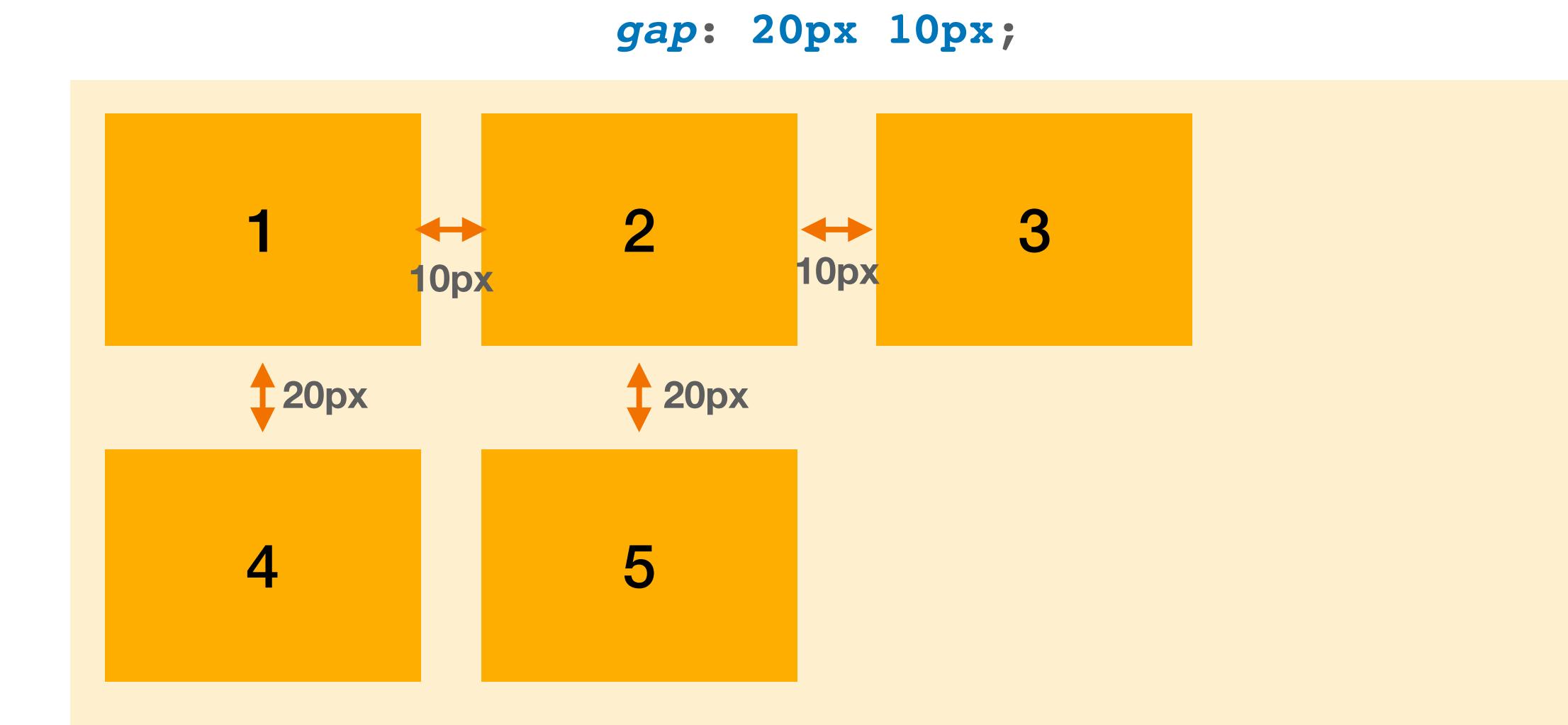
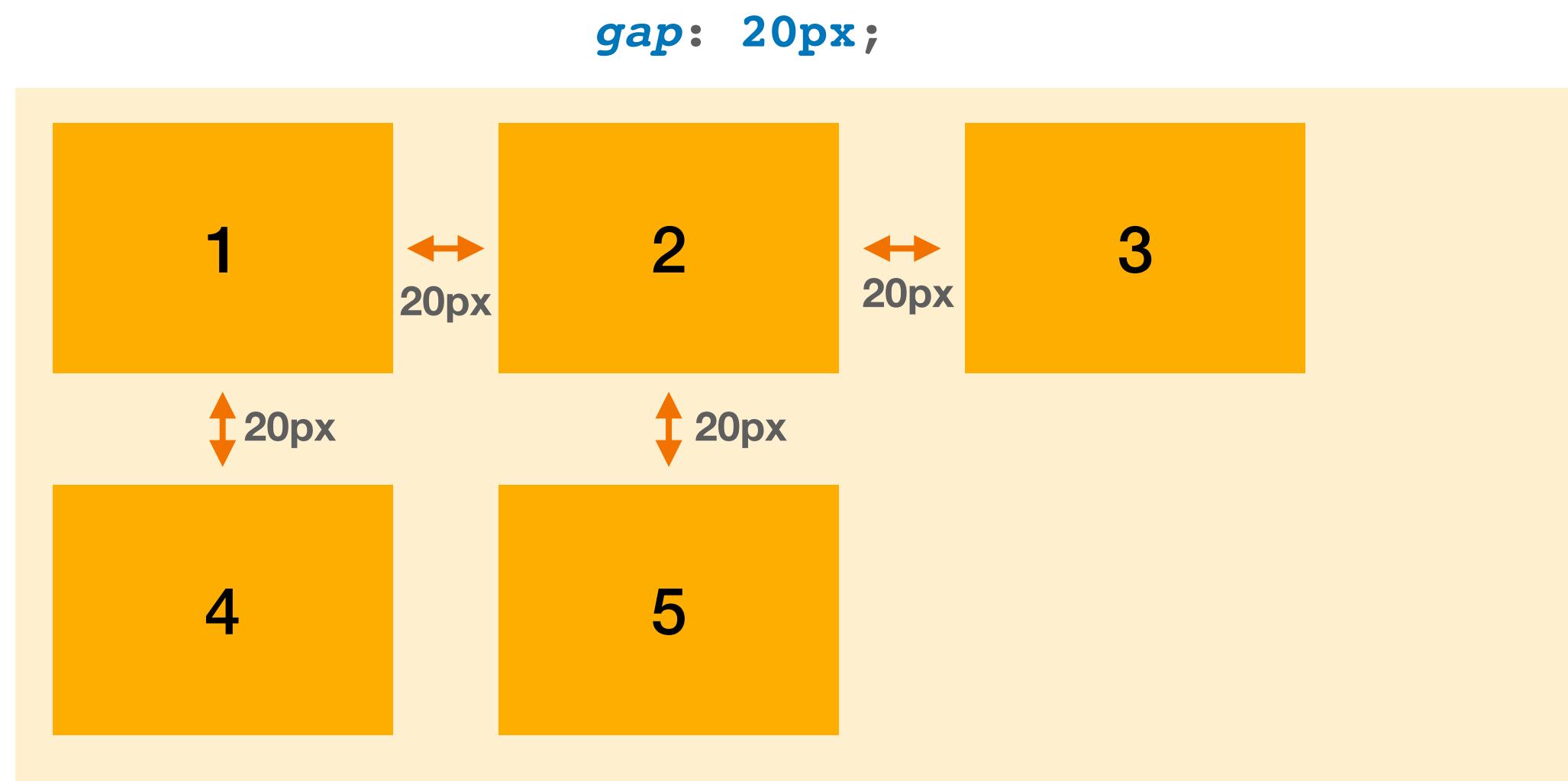


align-content: space-around;



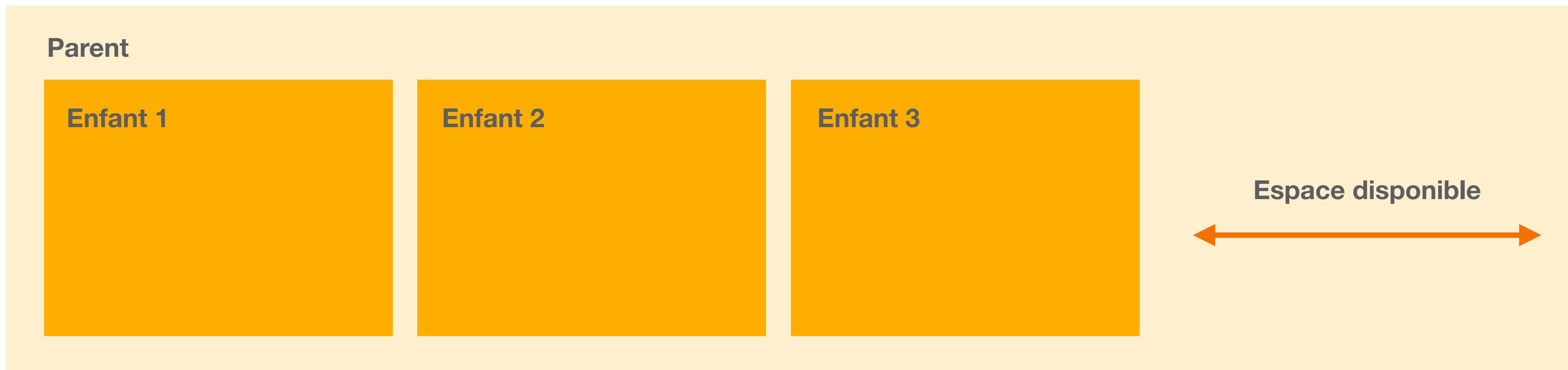
Flexboxes - Ecartement (gap)

css



Flexboxes - Sizing

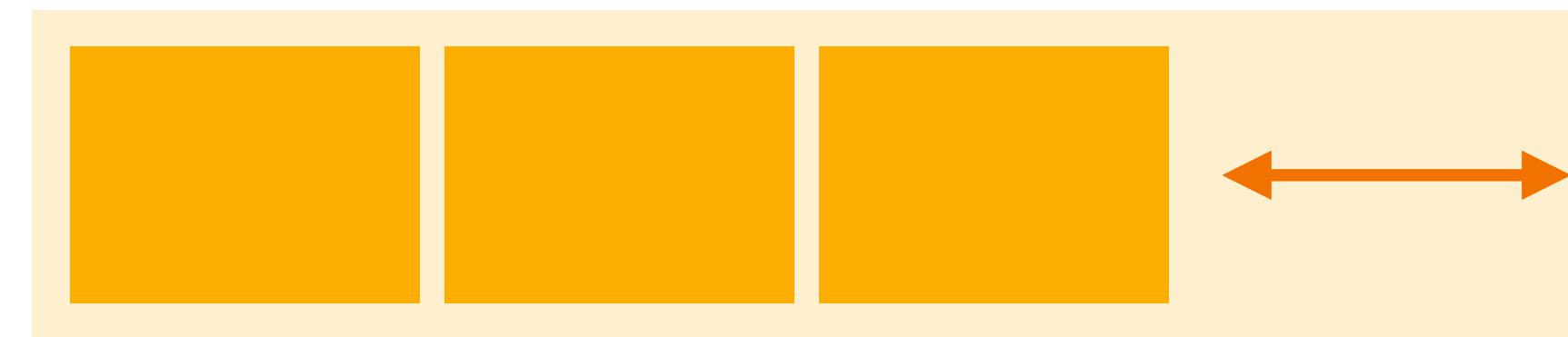
CSS



Flexboxes - Sizing & responsiveness

CSS

flex-basis: [px|%|auto|...];



Rétrécissement

flex-shrink: entier;



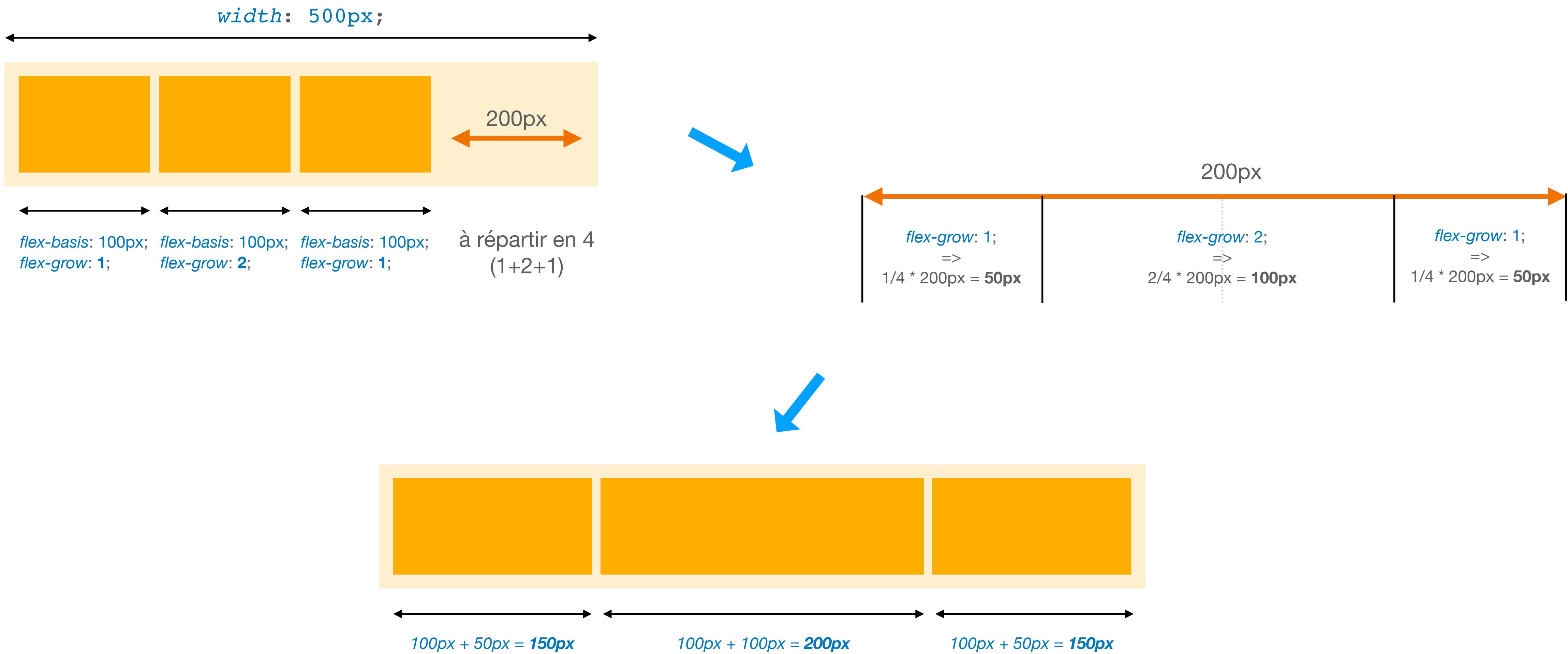
Agrandissement

flex-grow: entier;



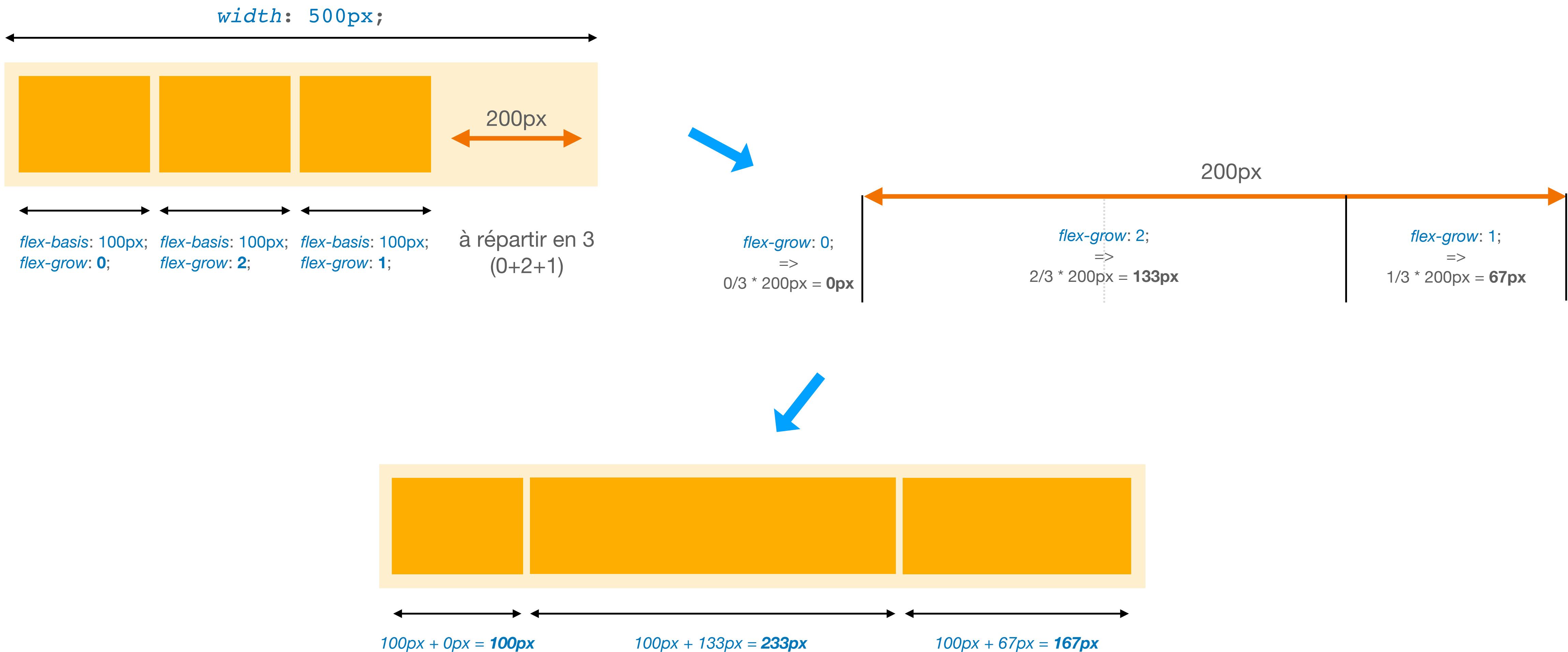
Flexboxes - Sizing flex-grow

CSS

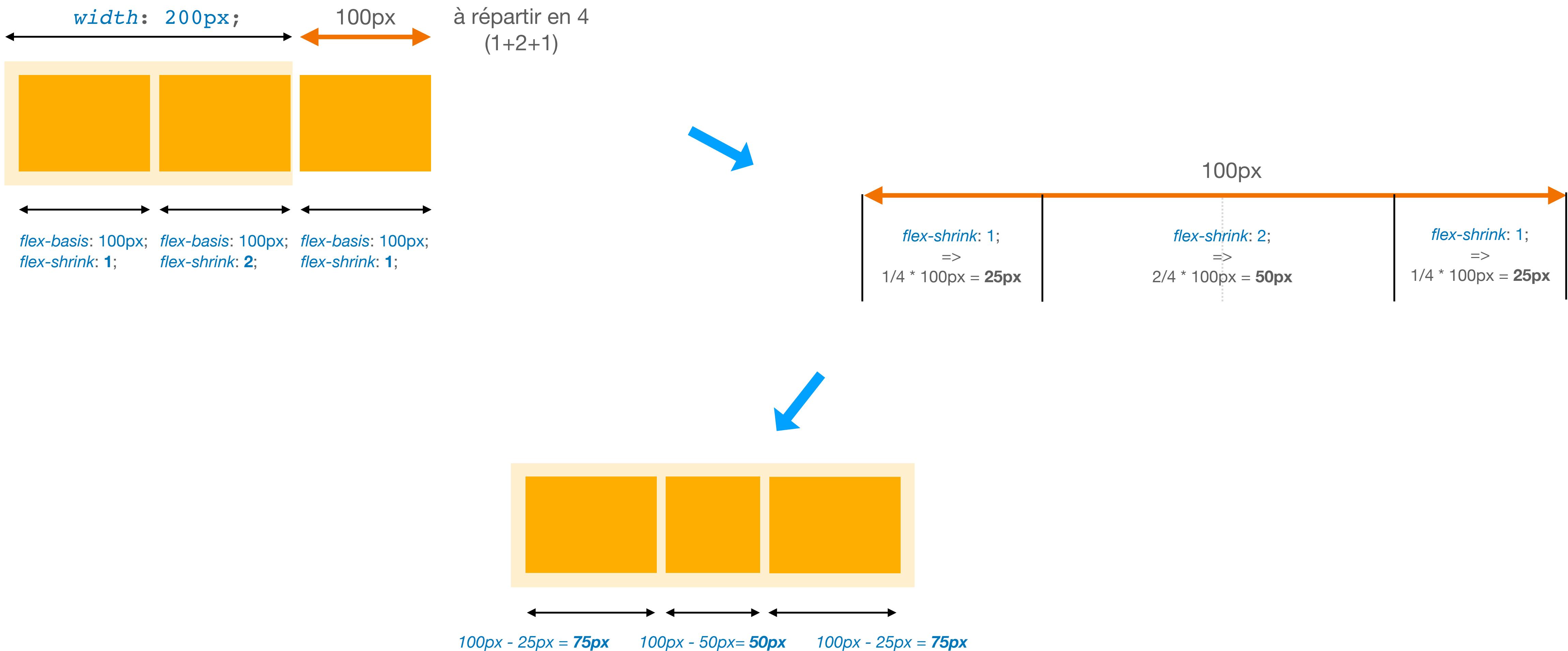


Flexboxes - Sizing flex-grow

CSS



Flexboxes - Sizing flex-shrink CSS



Flexboxes - Sizing flex-basis & sizing

CSS



Flex-basis hérite du width (si défini) ou “auto” par défaut



Par défaut, un élément flex enfant a un min-width défini à 0 et non “auto” comme les autres éléments (resp. height selon direction)



Flex-grow vaut 0 par défaut

Cela implique que, par défaut, il ne peut être plus petit que son contenu et overflow son parent



Flex-shrink vaut 0 par défaut

Pour autoriser l’overflow, définir “min-width: auto” sur l’enfant

Flexboxes - Overrides parents

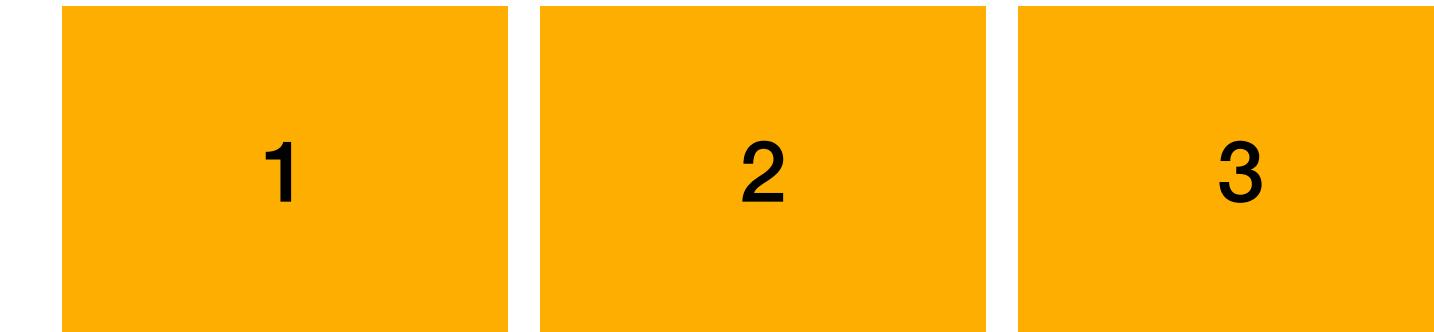
CSS

- `align-self` (overrides `align-items`)
- `justify-self` (overrides `justify-content`)

Flexboxes - Order

CSS

```
<div style="display: flex">  
  <div style="order: 2">2</div>  
  <div style="order: 3">3</div>  
  <div style="order: 1">1</div>  
</div>
```

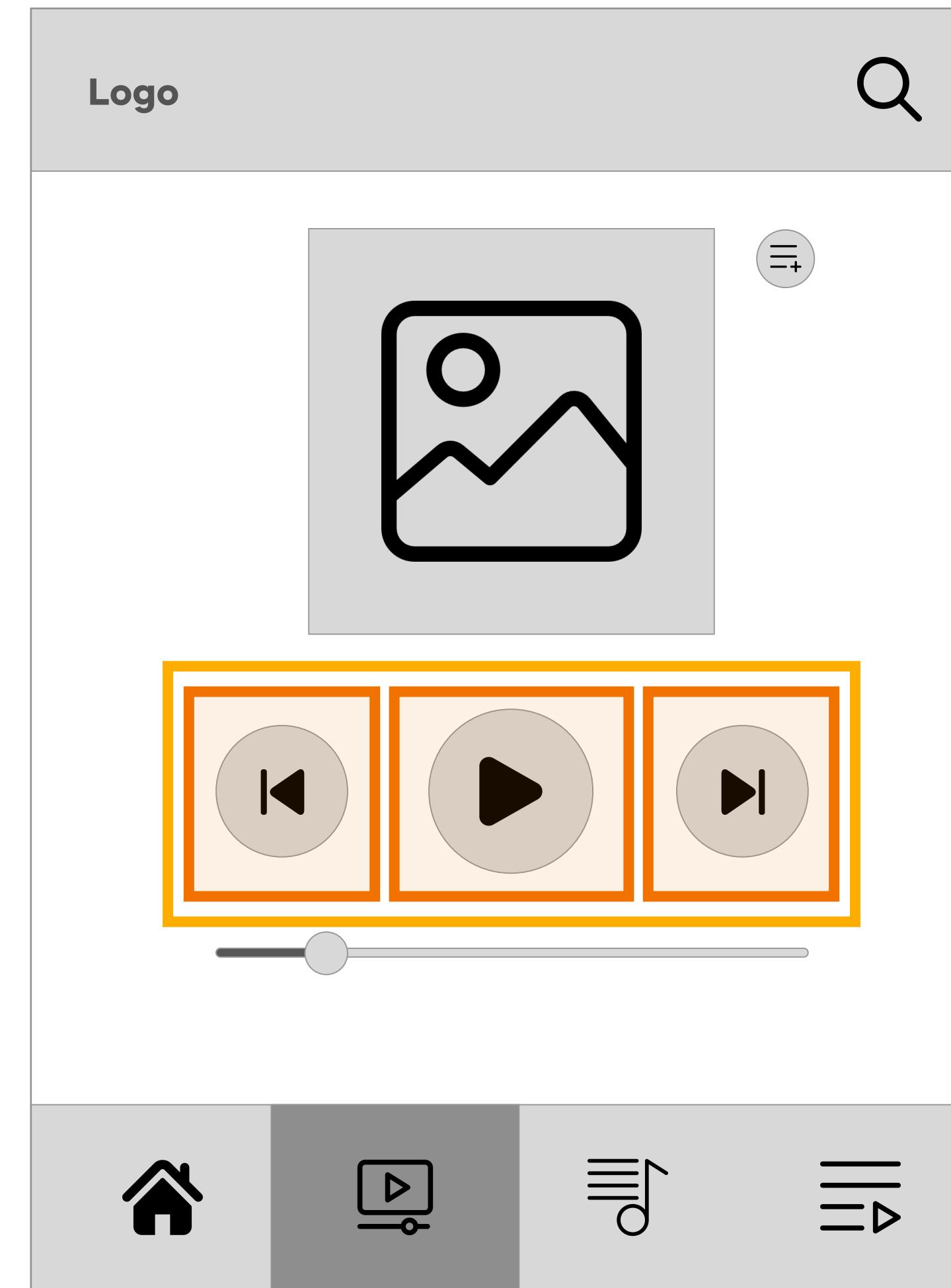


Flexboxes dans le projet

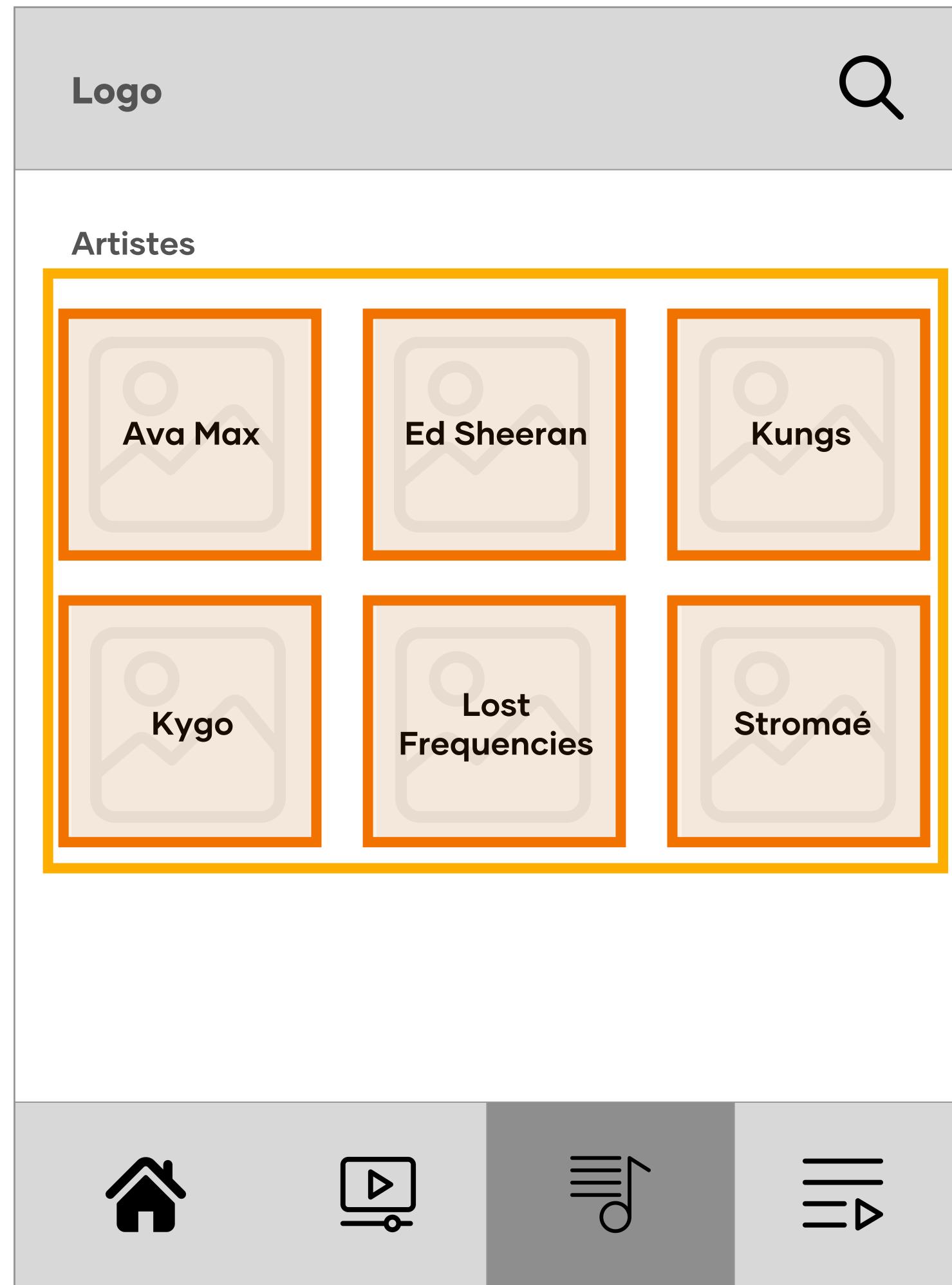
CSS



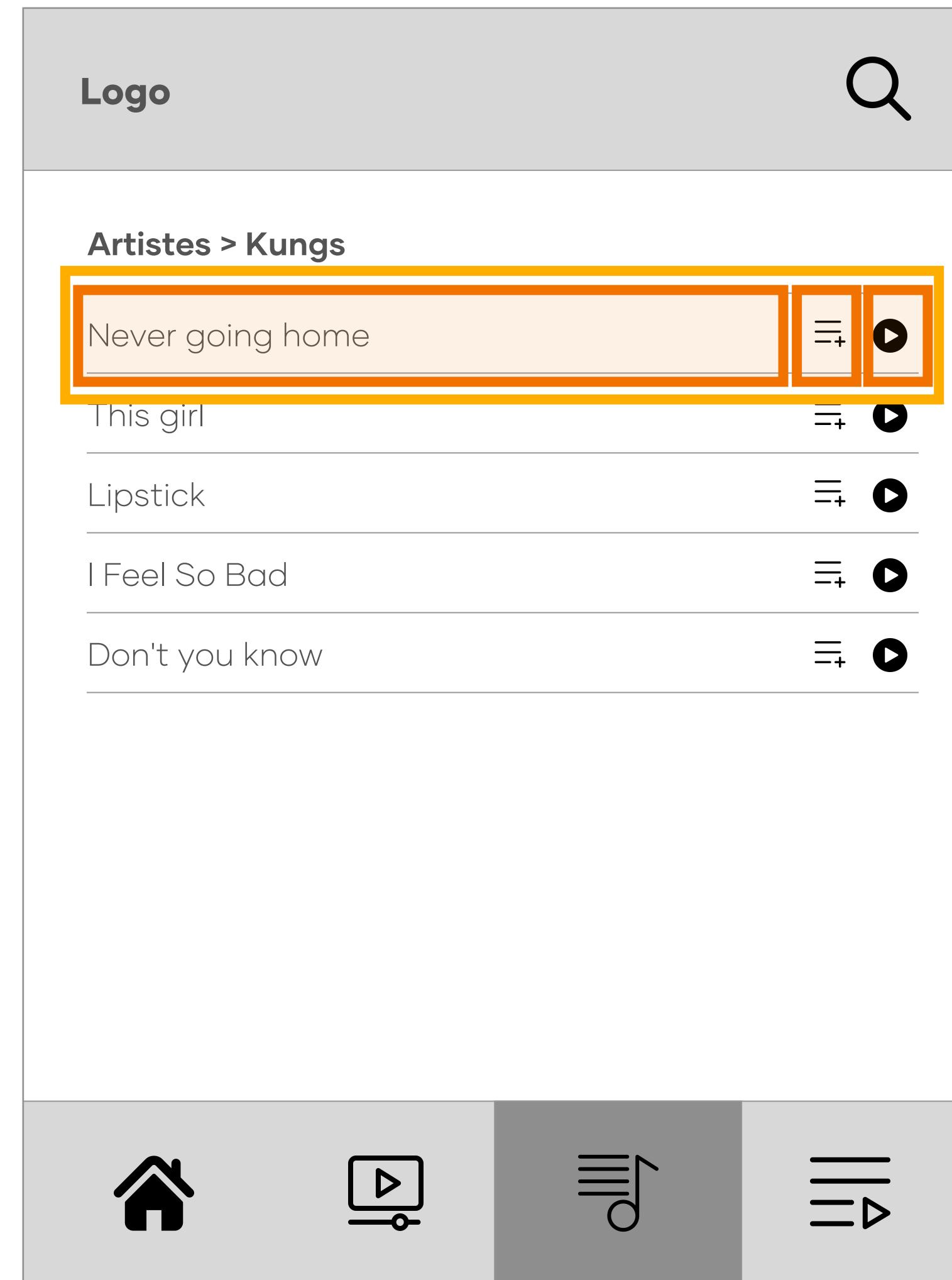
Flexboxes dans le projet css



Flexboxes dans le projet CSS



Flexboxes dans le projet CSS



Code

En parlant de CDN...

Tips & tricks

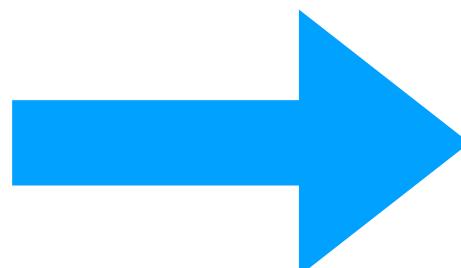
Super CDN pour des images **placeholder**, peu importe votre projet :

placekitten.com

Usage: [http://placekitten.com/\[width\]/\[height\]](http://placekitten.com/[width]/[height])

```

```



Unités CSS

CSS

Deux types d'unités :

- Unités absolues - px, cm, pt, in, ...
- Unités relatives - %, em, rem, vw, vh, ...

Unités CSS - em css

- 1 em = Taille de police à 100% de l'élément parent
- Si pas d'élément parent, 1 em est égal à la hauteur d'une lettre en taille standard, selon la résolution d'écran

Unités CSS - em

CSS

Pas de parent ? Résolution par défaut. Exemple: 16px

```
<body>  
<div>  
  <p>Hello</p>  
</div>  
</body>
```

```
body {  
  font-size: 1em;  
}  
  
div {  
  font-size: 0.75em;      3/4 du parent -> 9 pixels  
}  
  
p {  
  font-size: 0.75em;  
}
```

Unités CSS - rem

css

- 1 rem = Taille de police à 100% de l'élément root (**Root EM**)
- Si pas d'élément parent, 1 rem est égal à la hauteur d'une lettre en taille standard, selon la résolution d'écran

Unités CSS - rem

css

Pas de parent ? Résolution par défaut. Exemple: 16px

```
<body>  
<div>  
  <p>Hello</p>  
</div>  
</body>
```

```
body {  
  font-size: 1rem;  
}  
  
div {  
  font-size: 0.75rem; 3/4 du root -> 12 pixels  
}  
  
p {  
  font-size: 0.75rem;  
}
```

Unités CSS - rem

css

- Le REM est beaucoup plus utilisé, typiquement pour des design responsive ou dans des frameworks CSS
- En définissant toutes les tailles de polices et marges en rem, d'après l'élément root, une seule adaptation du CSS met à jour l'entier du document

Unités CSS - rem

CSS

```
body {  
    font-size: 16px;  
}  
  
h1 {  
    font-size: 2.5rem;  
    margin-bottom: 1rem;  
}  
  
h2 {  
    font-size: 2rem;  
    margin-bottom: 0.75rem;  
}  
...
```

Unités CSS - vw, vh

CSS

- vw = viewport width
- vh = viewport height
- Très utile pour donner la largeur ou la hauteur de l'écran à un élément, sans devoir chaîner des `height: 100%` sur tous les parents

Variables CSS

CSS

- CSS dispose aussi de variables, comme Javascript par exemple
- Une variable commence toujours par "--"
- Ce sont en fait des custom properties... mais utilisées comme variables !
- Une variable peut être déclarée sur n'importe quel déclaration, mais ne sera accessible que par l'élément et ses enfants

Variables CSS

css

Déclaration

```
body {  
  --text-color: #00ff00;  
}
```

Utilisation

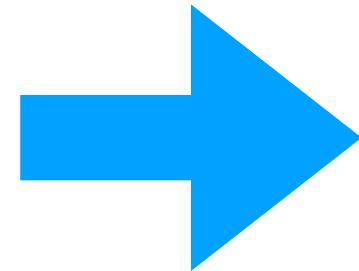
```
h1 {  
  color: var(--text-color);  
}
```

Variables CSS - Portée css

Exemple

```
header {  
  --text-color: #00ff00;  
}
```

```
footer {  
  /* rien. */  
}
```



```
header h1 {  
  color: var(--text-color); /* Yes! */  
}
```

```
footer h1 {  
  color: var(--text-color); /* Nope. Invalide */  
}
```

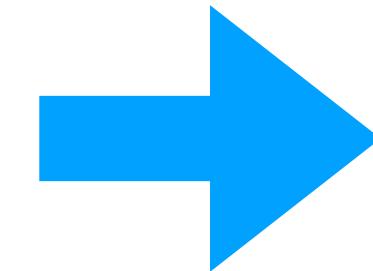
Variables CSS - Déclaration idéale

css

```
/* Utiliser le pseudo element :root */  
  
:root {  
  --text-color: #00ff00;  
}
```

Variables CSS - Variable d'une variable CSS

```
:root {  
  --primary-color: #00ff00;  
  
  --link-color: var(--primary-color)  
}
```



```
--primary-color => #00ff00  
  
--link-color => #00ff00  
}
```

Viewport css

- Le viewport est la zone de la fenêtre dans laquelle le contenu web peut être vu
- Le viewport est souvent plus grand que la zone affichée par le navigateur -> La view

Viewport CSS



https://developer.mozilla.org/fr/docs/Web/HTML/Viewport_meta_tag

Viewport - Déjà vu ? Vue compactée ?

CSS



https://developer.mozilla.org/fr/docs/Web/HTML/Viewport_meta_tag

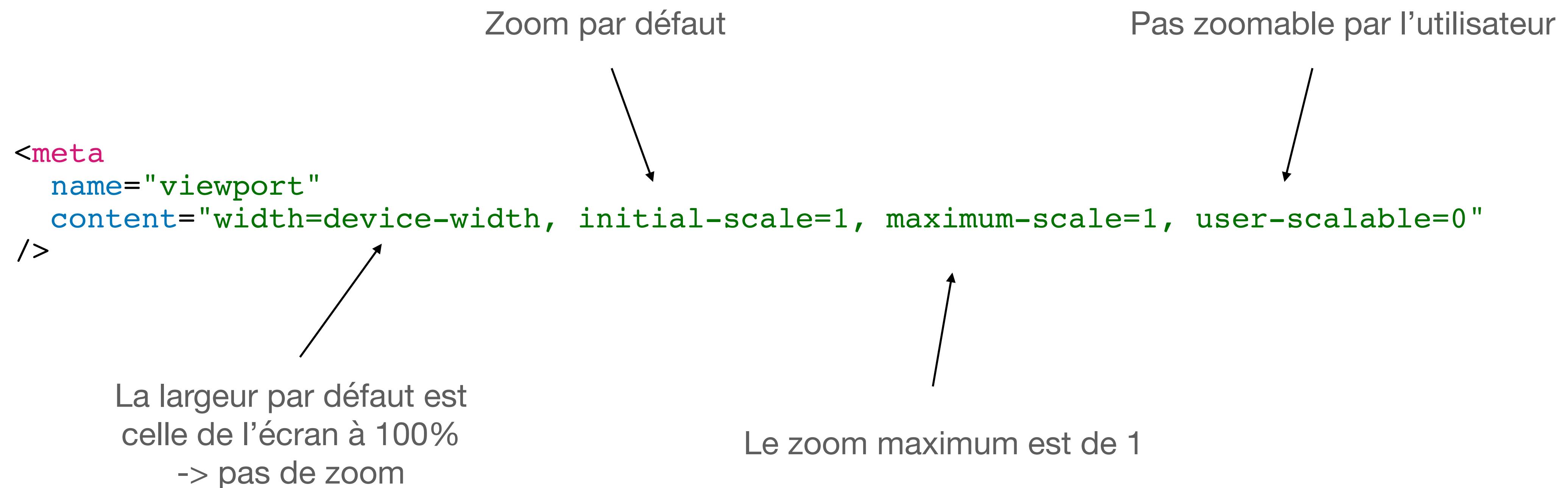
Viewport - Autozoom

CSS

- Les résolutions des versions mobiles sont de plus en plus précises et on ne distingue plus les pixels
- Un mécanisme d'autozoom est utilisé par les navigateurs pour renderer une version plus large de la page et la redimensionner à une résolution plus petite
- Par exemple, si l'écran d'un téléphone mobile a une largeur de 640 pixels, les pages peuvent être affichées dans une fenêtre virtuelle de 980 pixels, puis réduites pour tenir dans l'espace de 640 pixels.

Viewport - La balise meta viewport

CSS



Media queries

CSS

- Design responsive
- Blocs CSS conditionnels liés au type de display ou à sa taille
- Applicable à certains tags HTML via l'attribut `media=`
- Utilisable via Javascript pour tester et surveiller

Media queries - Blocs CSS

```
@media [not|only] mediatype and (mediafeature [and|or|not] mediafeature) {  
    ...  
    CSS-Code;  
    ...  
}
```

mediatype = all | print | screen | speech

mediafeature = [max-width | min-height | orientation | ...]: value

Media queries - Blocs CSS

CSS

Exemple 1

```
.container {  
  display: flex;  
  flex-direction: row;  
}  
...  
  
@media (max-width: 767px) {  
  .container {  
    flex-direction: column;  
  }  
}
```

Exemple 2

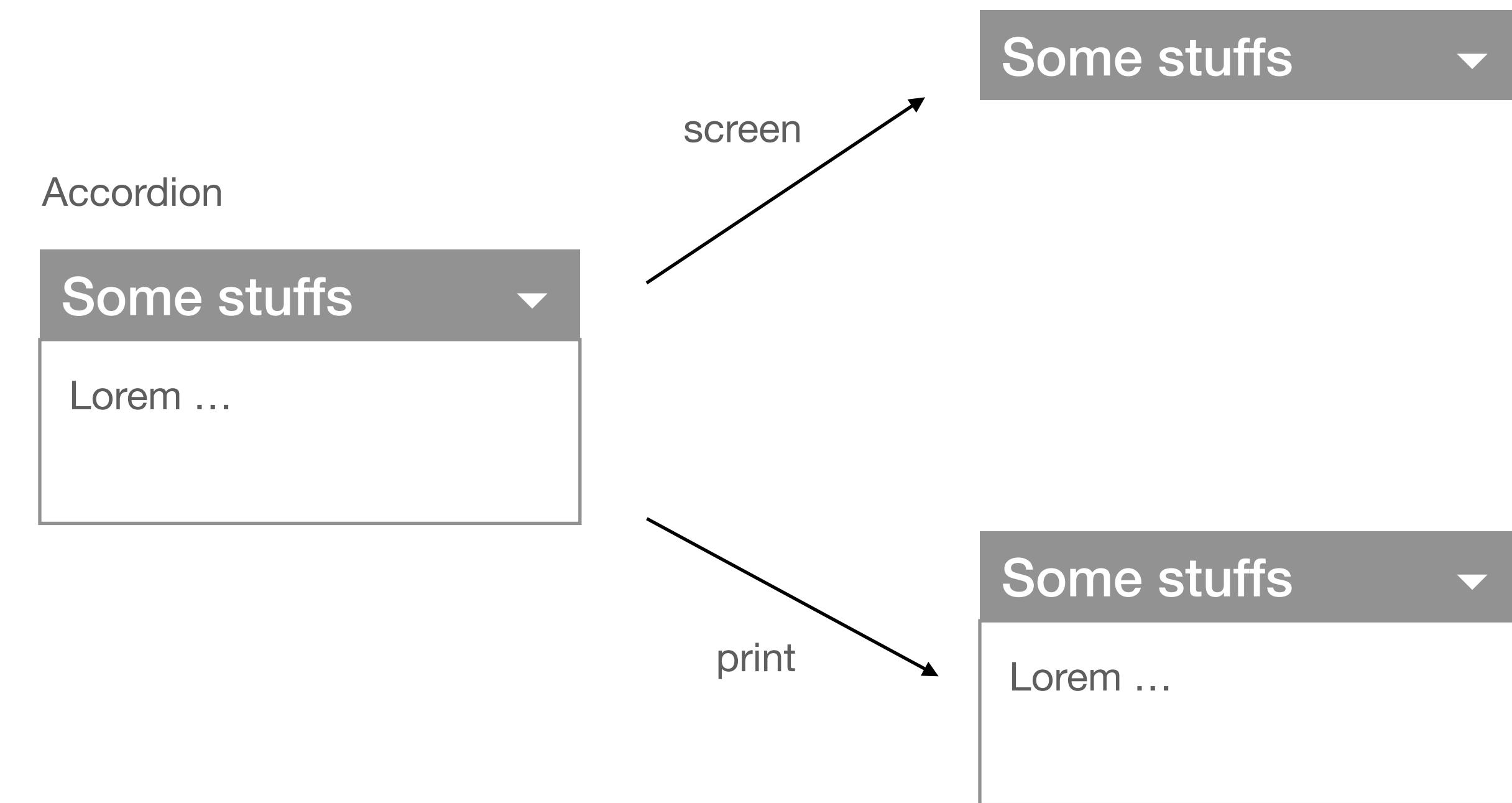
```
body {  
  background-color: white;  
}  
...  
  
@media screen and (prefers-color-scheme: dark) {  
  body {  
    background-color: black;  
  }  
}
```

Media queries - Blocs CSS

CSS

Exemple 3

```
.accordion .accordion-content {  
    display: none;  
}  
...  
  
@media print {  
    .accordion .accordion-content {  
        display: block;  
}
```



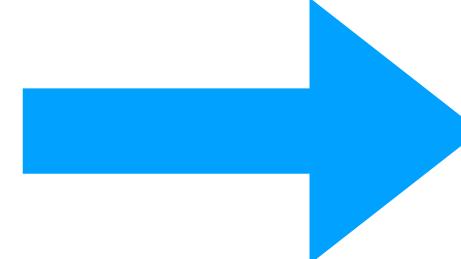
Media queries - Blocs CSS



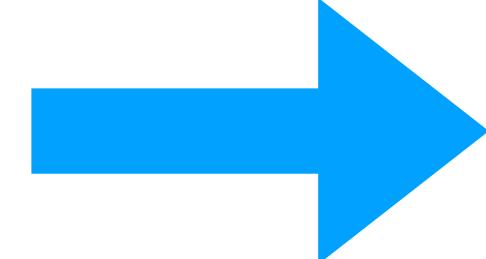
Cumul des règles par qualification des sélecteurs

```
.container {  
  display: flex;  
  flex-direction: row;  
}  
...  
  
@media (max-width: 767px) {  
  .container {  
    flex-direction: column;  
  }  
}
```

Sélecteur interprété
comme



```
.container {  
  display: flex;  
  flex-direction: row;  
}  
...  
  
@media (max-width: 767px) .container {  
  flex-direction: column;  
}
```



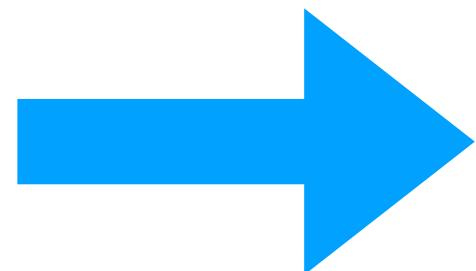
Media queries - Blocs CSS



Cumul des règles par qualification des sélecteurs

```
.container {  
  display: flex;  
  flex-direction: row;  
}  
...  
  
@media (max-width: 767px) .container {  
  flex-direction: column;  
}
```

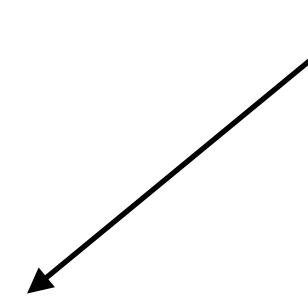
Comparabile à une sur-qualification
du type



```
.container {  
  display: flex;  
  flex-direction: row;  
}  
...  
  
body .container {  
  flex-direction: column;  
}
```

Media queries - Tags HTML CSS

Accepte des media queries



```
<link rel="stylesheet" media="screen and (min-width: 1201px)" href="widescreen.css">
<link rel="stylesheet" media="screen and (max-width: 600px)" href="smallscreen.css">
```



Avantages ? Inconvénients ?

Media queries - Breakpoints

CSS

- **320px – 480px** : Mobile devices
- **481px – 768px** : iPads, Tablets
- **769px – 1024px** : Small screens, laptops
- **1025px – 1200px** : Desktops, large screens
- **1201px and more** : Extra large screens, TV

Media queries - Javascript

CSS

Tester une media query

```
const mql = window.matchMedia("(orientation: portrait)");

if (mql.matches) {
  /* La zone d'affichage/viewport est en portrait */
} else {
  /* La zone d'affichage/viewport est en paysage */
}
```

Media queries - Javascript

CSS

Ajouter un listener sur une media query

```
const mql = window.matchMedia("(orientation: portrait)");
mql.addListener(handleOrientationChange);

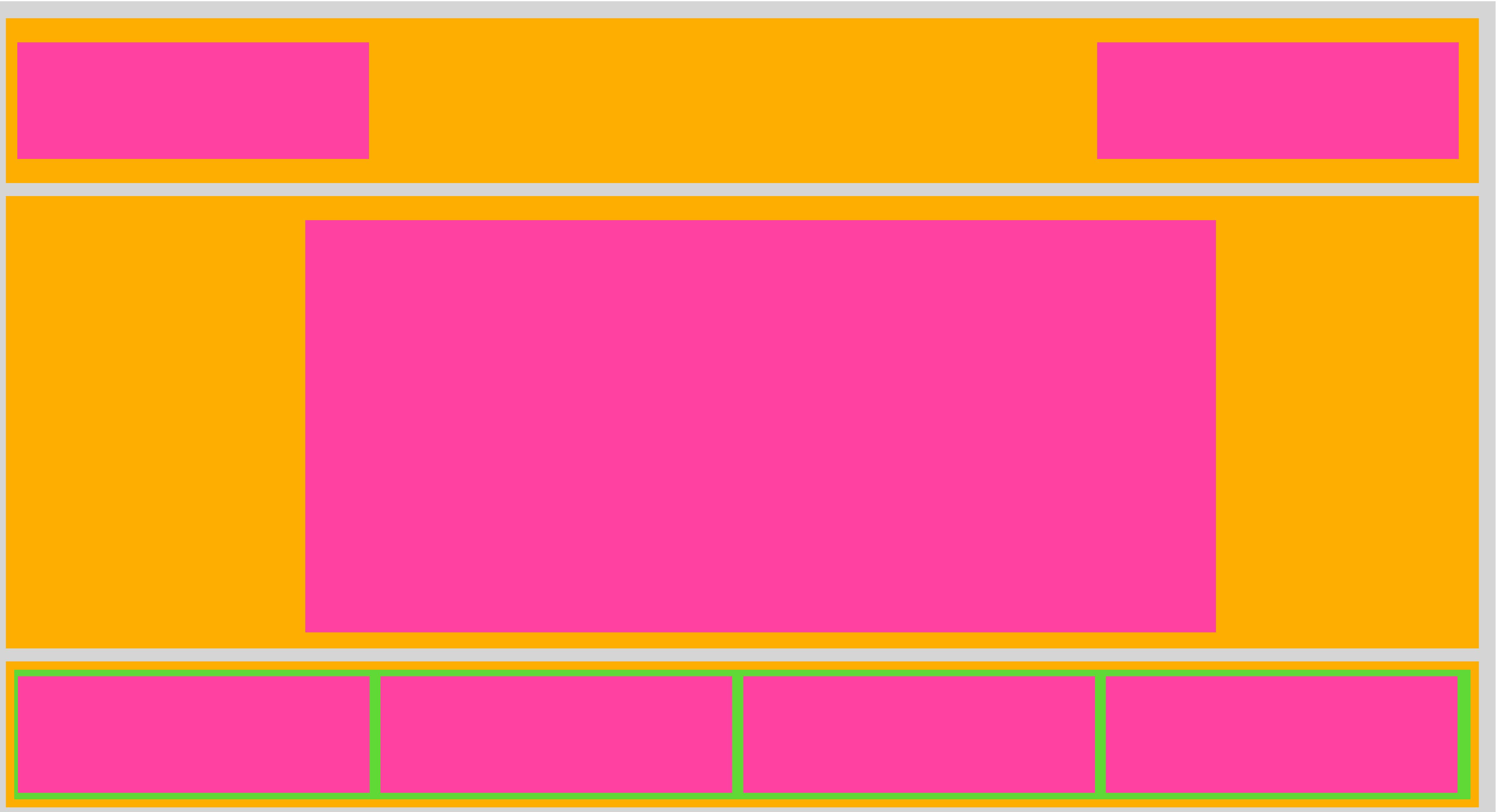
function handleOrientationChange(mql){
    if (mql.matches) {
        /* La zone d'affichage/viewport est en portrait */
    } else {
        /* La zone d'affichage/viewport est en paysage */
    }
}
```

Composants ?

Projet

- ✓ Projet webpack vide
 - Popover pour playlists
 - Détection online/offline
 - Local storage pour les playlists
 - Manifest PWA
 - Caching
 - Service worker
- ✓ Squelette HTML
- ✓ Styles CSS structurels
- ✓ Icônes
- Routeur pour les pages web
- Client pour l'API JSON
- Lecteur audio

Code



Rubber duck debugging

Tips & tricks



https://fr.wikipedia.org/wiki/M%C3%A9thode_du_canard_en_plastique

JS

Single Page Application

JS

- Une Single Page Application est une application web qui réside sur une seule page HTML
- Le javascript va gérer l'affichage des différents éléments et non les pages HTML
- Typiquement utilisé pour des applications PWA

Single Page Application

JS

Comment renderer
les différentes pages ?

Single Page Application

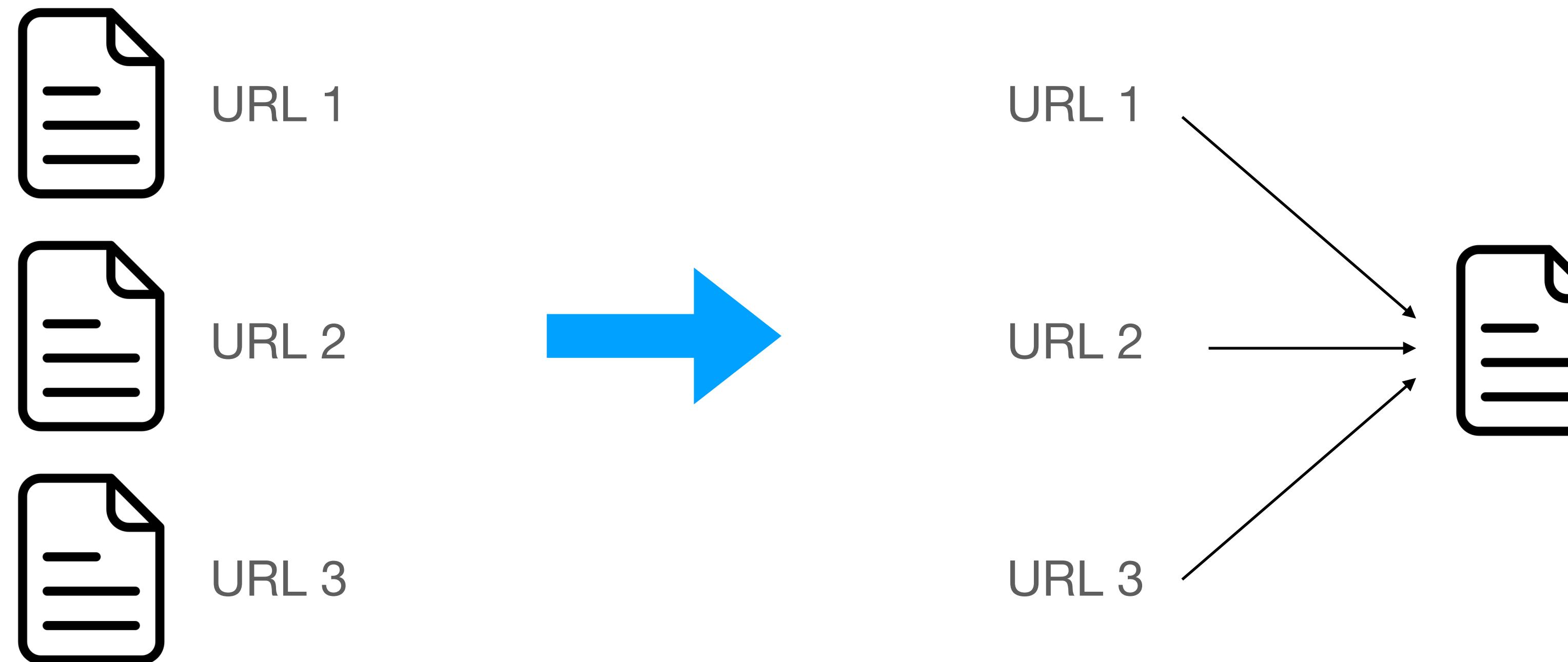
JS

Deux options principales :

- Les vues ou pages sont existantes dans le DOM et sont affichées/cachées par CSS (`display: none`)
- Les vues ou pages sont à chaque fois renderée au complet par le javascript. Typiquement un custom element ou une classe javascript avec une méthode `render`

Single Page Application - Routing

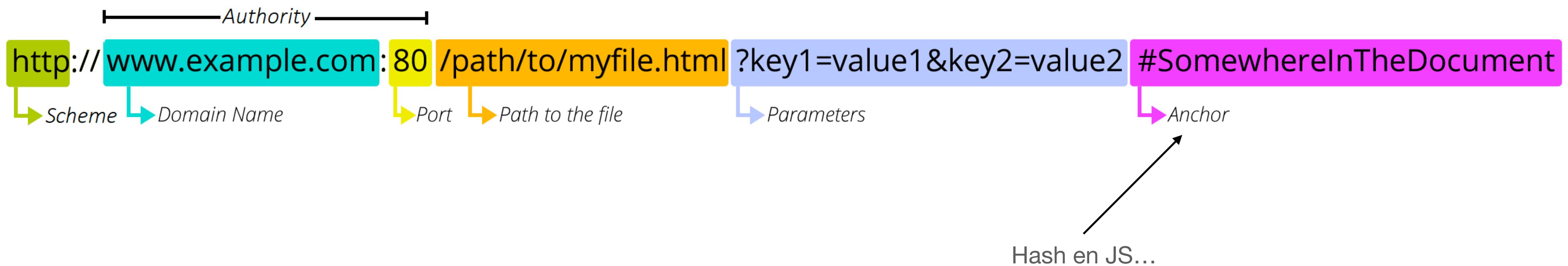
JS



Single Page Application - Routing JS

Comment géré l'état
de la page en cours?

Single Page Application - Think RESTful... JS



https://developer.mozilla.org/en-US/docs/Learn/Common_questions/What_is_a_URL

Single Page Application - Think RESTful... JS

http://localhost:8080/hello?something=bonjour#quelquechose

> window.location

=>

```
hash => "#quelquechose"
host => "localhost:8080"
hostname => "localhost"
href => "http://localhost:8080/hello?something=bonjour#quelquechose"
origin => "http://localhost:8080"
pathname => "/hello"
port => "8080"
protocol => "http:"
search => "?something=bonjour"
```

Single Page Application - Routing JS

Deux options principales :

- Utiliser les Anchors ou Hash
- Utiliser l'API history et la réécriture d'URL sur le path

Single Page Application - Anchors/Hash JS

- Historiquement, les anchors (<a />) sont des ancrés dans la page
- Une manière de mettre des liens internes à la page pour avancer dans le contenu
- Single page by design -> ne refresh pas la page lorsqu'on clique dessus
-> parfait pour notre app !

Single Page Application - Anchors/Hash JS

- Possible par exemple de prendre la structure d'URL suivante :
 - /#home
 - /#player
 - /#artists
 - /#playlists-12



Avantages ? Inconvénients ?