Web & Mobile UI – Cours 1

Conception d’applications web

Loris Gavillet, HEIG-VD, 02.2025, v1.13

Enseignants

Cours

Phase 1

Loris Gavillet

loris.gavillet@heig-vd.ch

Semaine 1 -> Semaine 6

Phase 2

Nicolas Chabloz

nicolas.chabloz@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
- Stack fonctionnelle

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 8 demi-journées de cours
- **Examen sur 4 périodes le 25.03.2026**

Technologies utilisées

Cours

Phase 1

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

Technologies utilisées

Cours

Repository et informations générales

Cours

Repository principal : https://github.com/lgavillet/webmobui

Repository live : https://github.com/lgavillet/webmobui-correction

Projet

Concept

Projet

Spotlified

Simplified spotify

Prototype

Projet

https://www.figma.com/proto/8RRFnlJnbTzH1csC2a1Ys4/Spotlified

Concept

Projet

Spotlified version simplifiée

3 modèles :

- Chansons
- Artistes
- Favoris

Architecture

Projet

Features

Projet

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

Composants ?

Projet

- Projet Vite vide
- Squelette HTML
- Styles CSS structurels
- Icônes
- Composants rendering
- Routeur pour les pages web
- Client pour l'API JSON
- Lecteur audio
- Local storage pour les favoris
- Détection online/offline
- Manifest PWA
- Caching
- Service worker
- Notifications Push

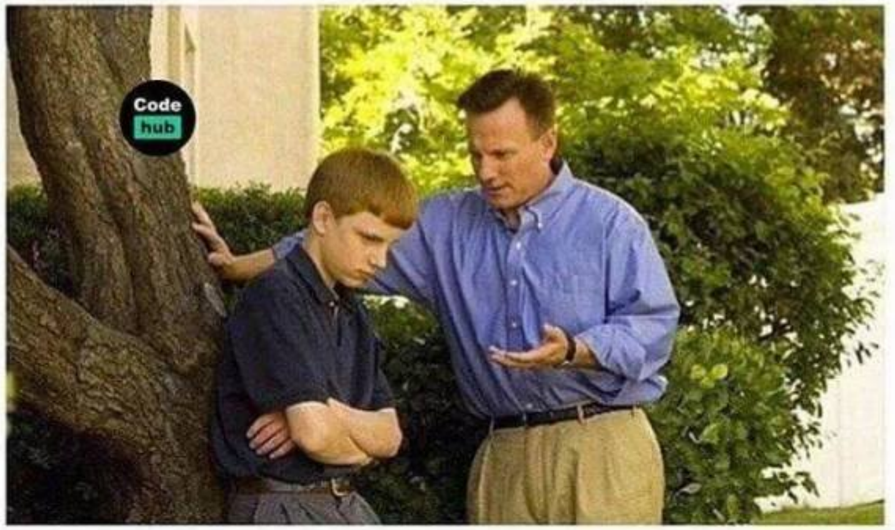
Truth be told

Projet

Dad: Why are your eyes red son?

Son: I smoke weed

Dad: Don't lie, you're crying because you have been coding in JavaScript



Packages

What is it?

Packages

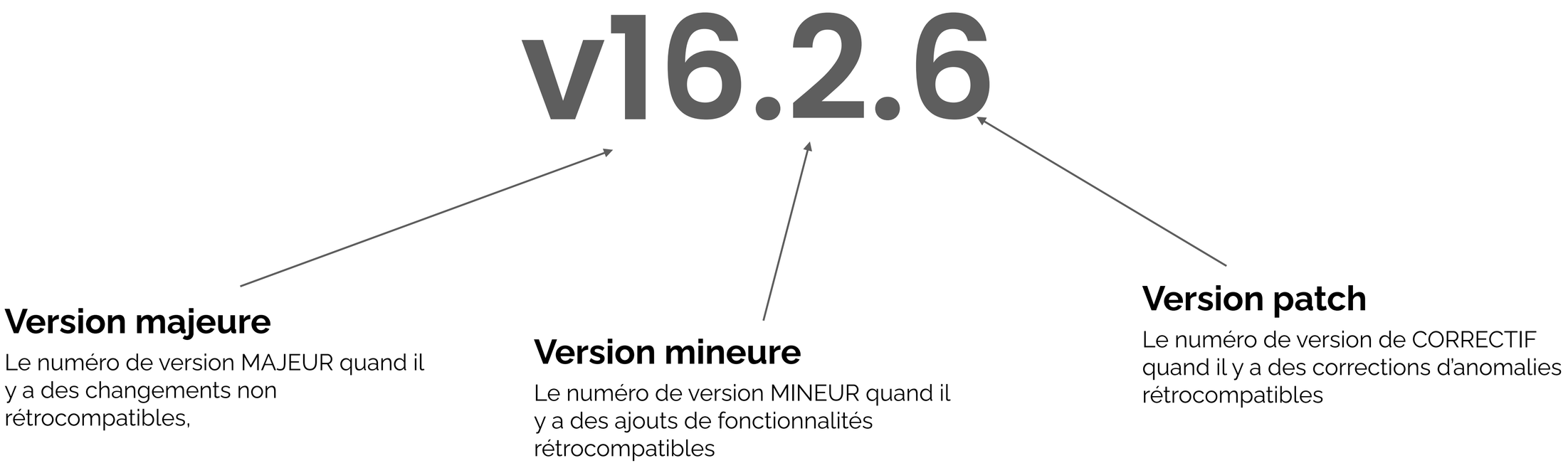
Définition

Package

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

Versioning – Semantic versioning

Package



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

https://semver.org/lang/fr/

Choix d'un package

Package

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

Intégration

Package

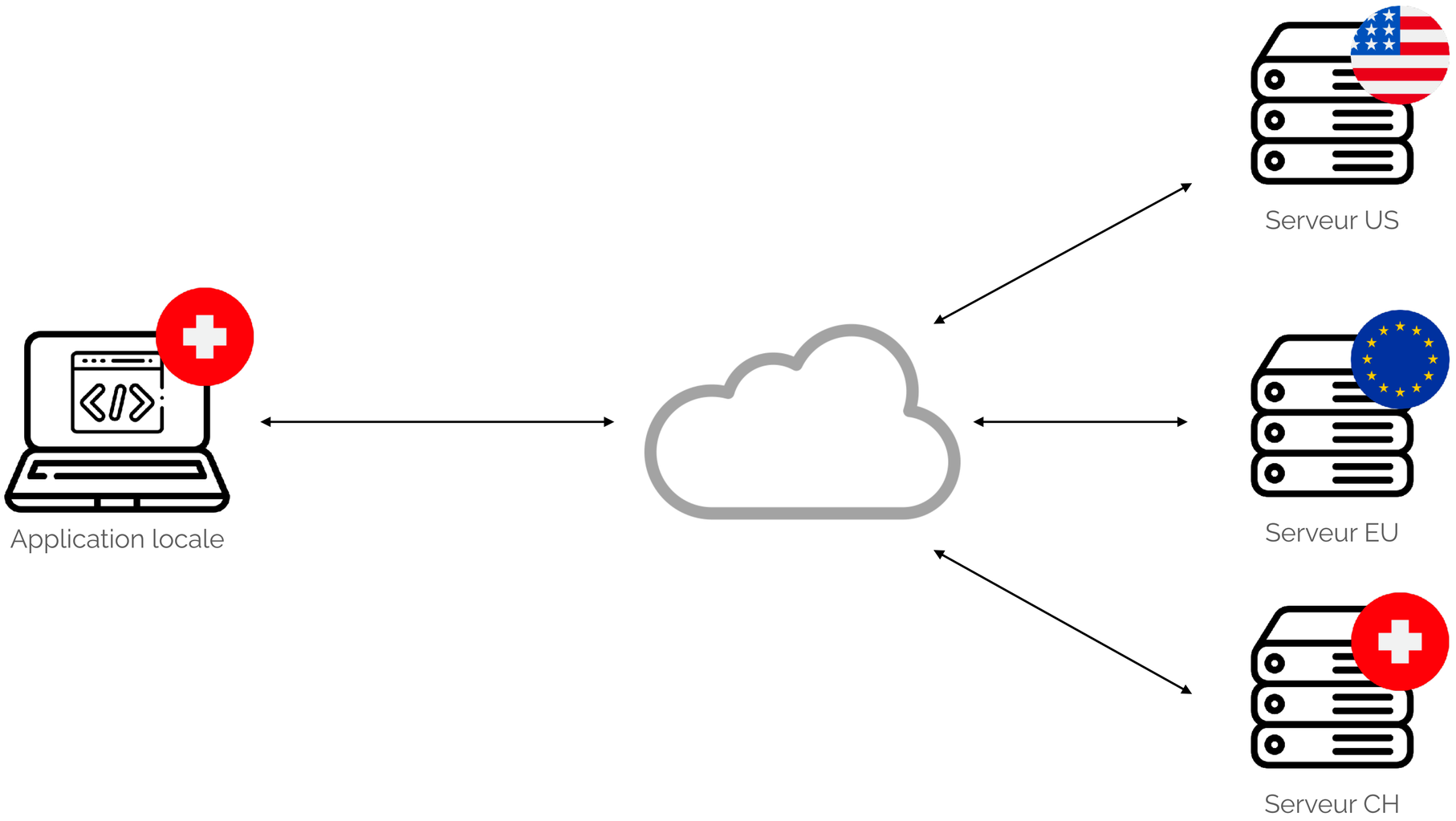
1. Copy-paste old school / Téléchargement

2. CDN - Content Delivery Network

3. Package manager

CDN – Content Delivery Network

Package



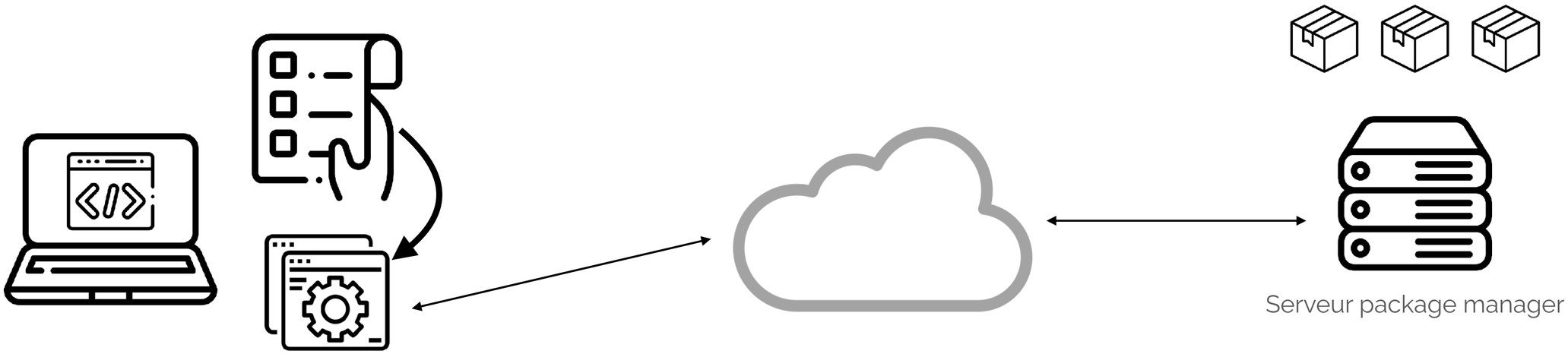
CDN – Content Delivery Network

Package

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

Package manager

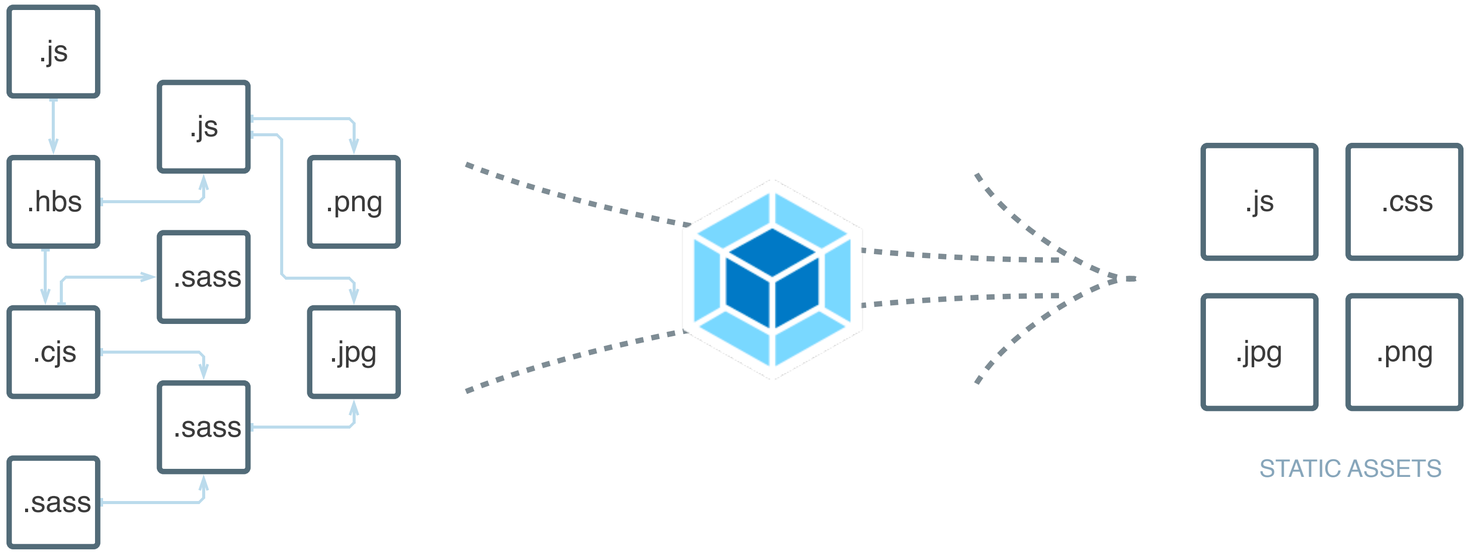
Package



Vite

Concept

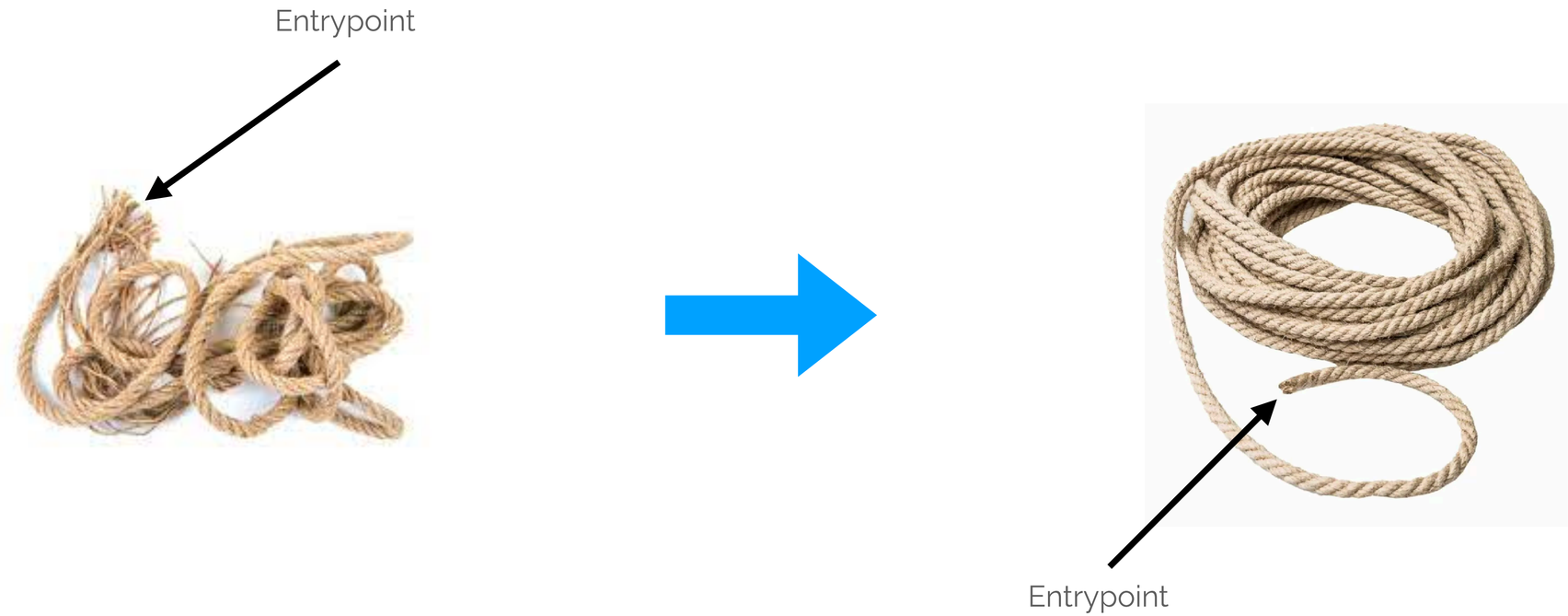
Vite



https://vite.dev/

Concept

Vite



Concept

Vite

“Vite 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.”

(Ok… this was taken from https://webpack.js.org/, but check https://vite.dev/guide/)

Concept

Vite

Basé sur

- NodeJS est un langage de programmation backend, écrit en Javascript
- A ne pas confondre avec du Javascript frontend
- Node est livré avec un gestionnaire de package : NPM (**N**ode **P**ackage **M**anager)

https://nodejs.org/en

Concept

Vite

- Vite est un package de NodeJS

- S'installe comme tout autre package avec

```
> npm install --save-dev vite
```

ou

```
> yarn add --dev vite
```

https://vite.dev/guide/

Pour le cours

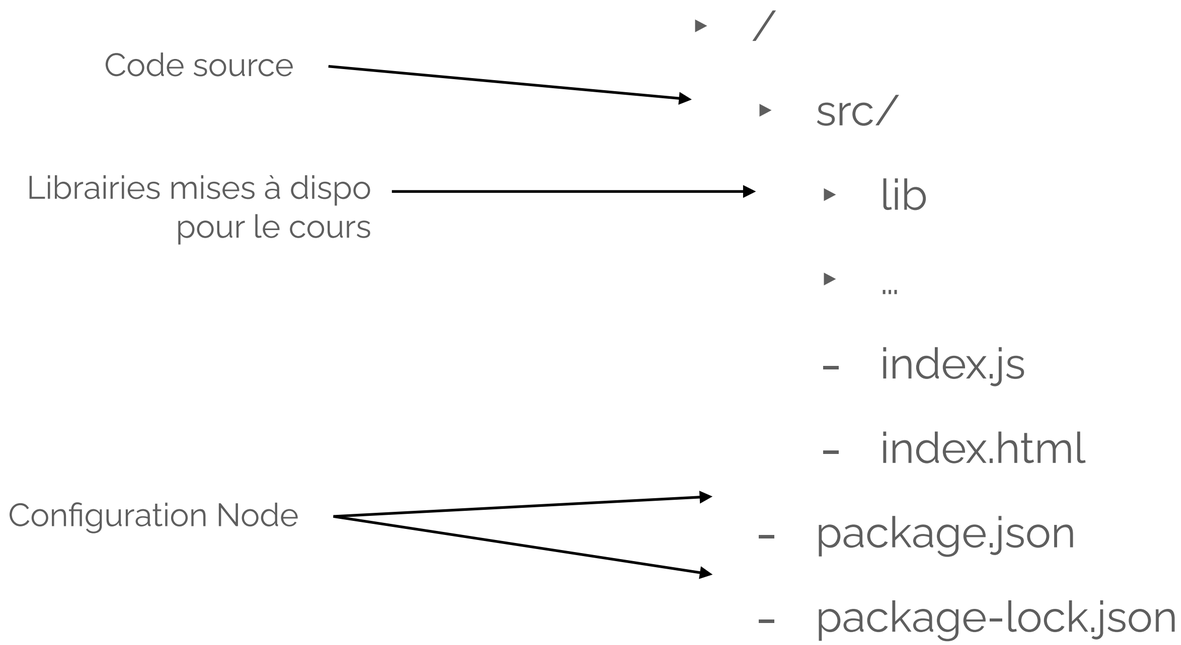
Vite

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

https://vite.dev/guide/

Structure de base utilisée

Vite



https://vite.dev/guide/

Node installé ?

Vite

```
> node --version
```

Installation

Vite

Installation

- Installer Node (si pas déjà fait) https://nodejs.org/en/download/ Version LTS
- Télécharger le projet vide sous: https://github.com/lgavillet/webmobui/raw/main/Ressources/projet-vide.zip
- Ouvrir l'invité de commande dans votre dossier projet

```
> npm install
```

https://vite.dev/guide/

Utilisation

Vite

Démarrer le serveur de dev

```
> npm start
```

ou

```
> npm run start
```

https://vite.dev/guide/

HTML / CSS / JS

W3C vs WHATWG

HTML / CSS / JS

- W3C (**W**orld **W**ide **W**eb **C**onsortium)

  Acteur historique…

- WHATWG (**W**eb **H**ypertext **A**pplication **T**echnology **W**orking **G**roup)

  Collaboration non officielle des différents navigateurs web

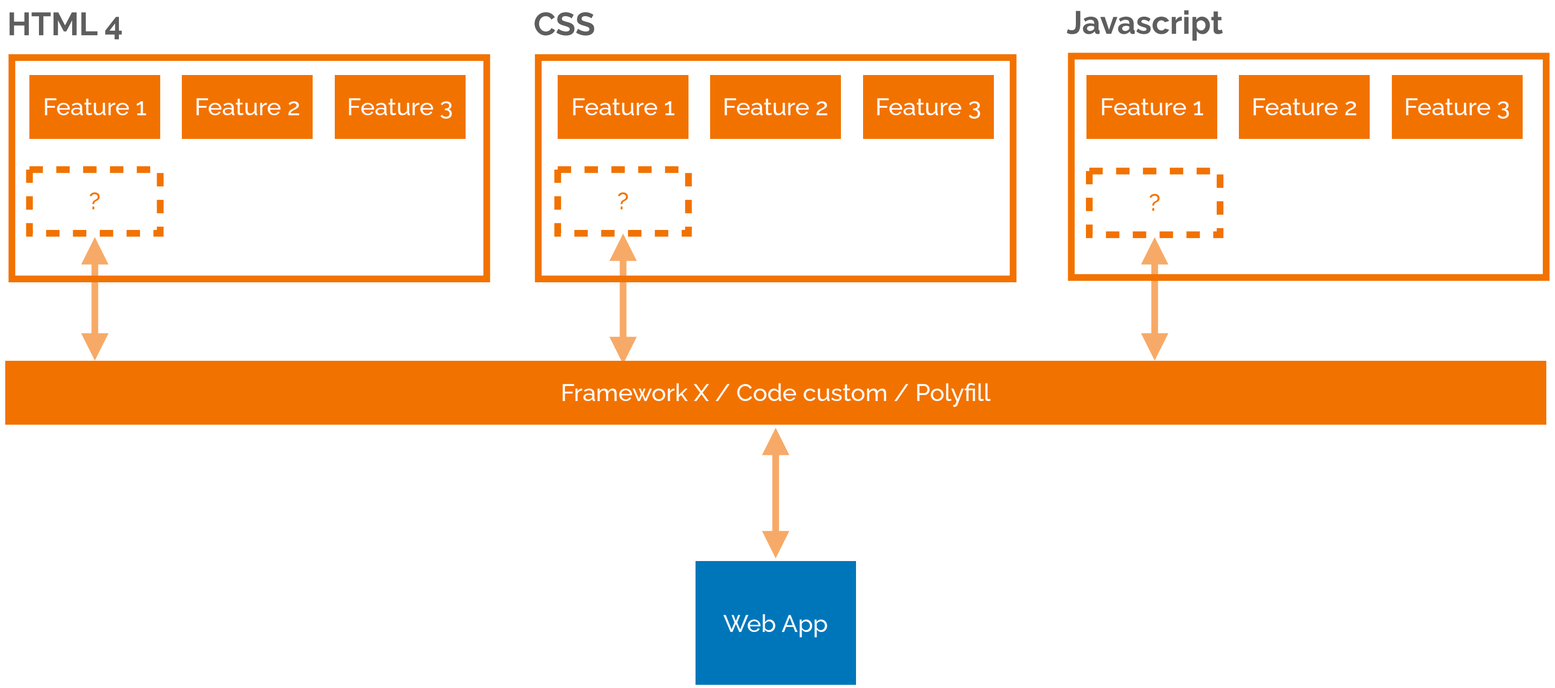
Nouveau standard HTML

HTML / CSS / JS

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

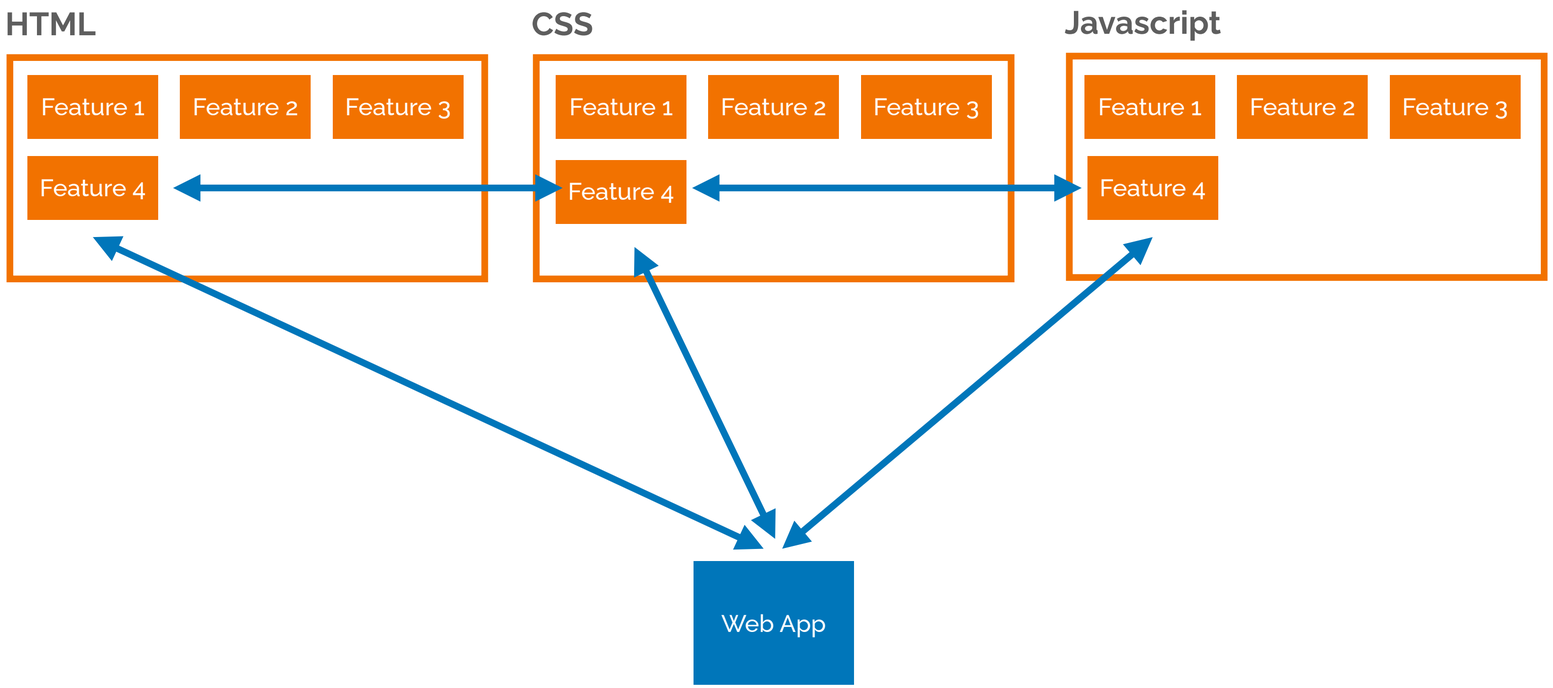
Standardisations des use cases courants

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

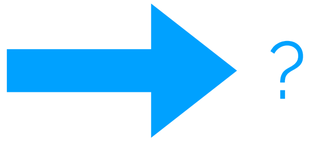
- Amélioration de HTML 4
- Fortement lié à Javascript
- Nouveaux éléments sémantiques (header, main, section, article, …)
- Form inputs améliorés (types d'inputs, claviers mobiles, validations, transformations)
- APIs (géolocalisation, local storage, history, push, service workers, …)
- Distribution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

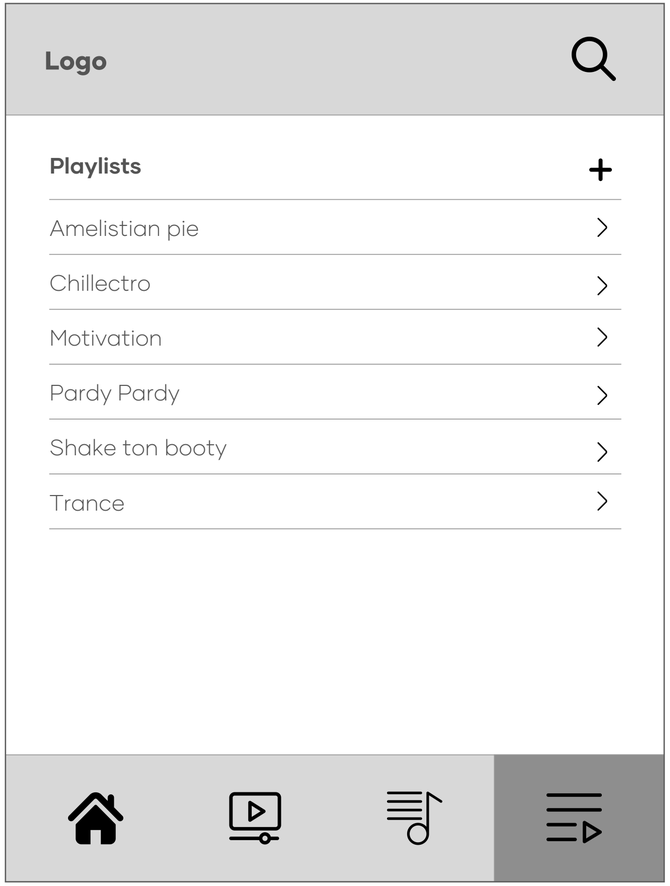
HTML

Eléments sémantiques

HTML

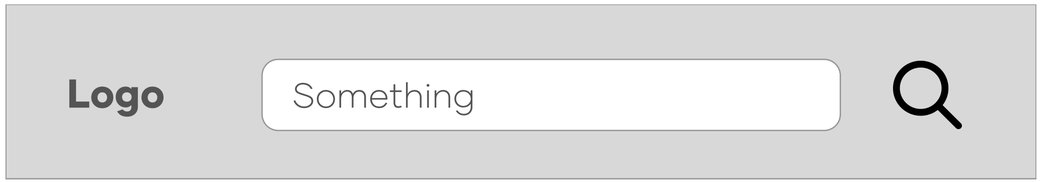
```
<header />
<nav />
<main />
<section />
<footer />
<article />
```





Inputs on steroids

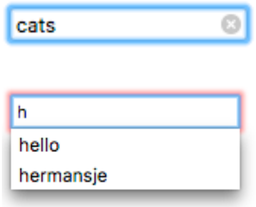
HTML

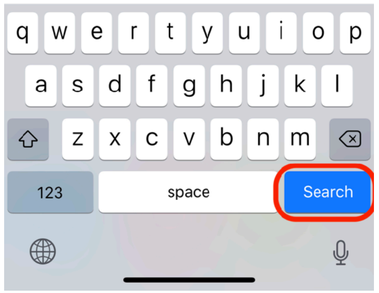


? `<input type="search" />`

Inputs on steroids

HTML






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

autofocus et autofocus=“true" sont sémantiquement équivalents

https://developer.mozilla.org/fr/docs/Web/HTML/Element/Input/search

Audio element

HTML

?

```
<audio src="https://…" />
```

0:00 / 0:01

https://developer.mozilla.org/fr/docs/Web/HTML/Element/audio

Limitations...

HTML and a bit more...

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

https://developer.mozilla.org/fr/docs/Web/HTML/Element/Input/search

Progress v1

HTML

?

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

https://developer.mozilla.org/fr/docs/Web/HTML/Element/Progress

Progress v2

HTML

?

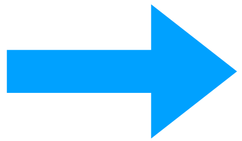
```
<input
  type="range"
  value="20"
  min="0"
  max="100"
/>
```

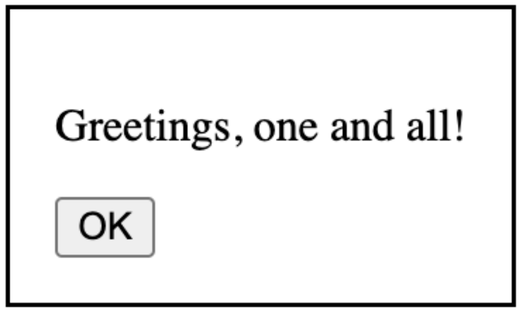
https://developer.mozilla.org/fr/docs/Web/HTML/Element/Input/range

Modal

HTML

```
<dialog>
  <p>Greetings, one and all!</p>
  <form method="dialog">
    <button>OK</button>
  </form>
</dialog>
```





Customer l'arrière plan avec le pseudo-element ::backdrop

https://developer.mozilla.org/en-US/docs/Web/HTML/Element/dialog

Image responsives

HTML

```
<img
  srcset="cat-480w.webp 480w, cat-800w.webp 800w”
  sizes="(max-width: 600px) 480px, 800px"
  src=“cat-800w.webp"
  alt=“Majestic cat"
/>
```

```
<img
  srcset=“cat-2x.webp 2x”
  src="cat.webp"
  alt=“Majestic cat"
/>
```

https://developer.mozilla.org/fr/docs/Learn/HTML/Multimedia_and_embedding/Responsive_images

CSS

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 : `<span class="material-icons">face</span>`
- Libre à vous !

En parlant de CDN...

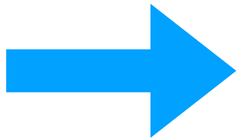
Tips & tricks

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

placekitten.com

Usage: http://placekitten.com/[width]/[height]

```
<img src="http://placekitten.com/200/139" />
```



Unités CSS

CSS

Deux types d'unités :

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

https://developer.mozilla.org/fr/docs/Learn/CSS/Building_blocks/Values_and_units

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

https://developer.mozilla.org/fr/docs/Learn/CSS/Building_blocks/Values_and_units

Unités CSS – em

CSS

```
<body>

  <div>

    <p>Hello</p>

  </div>

</body>
```

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

```
body {
  font-size: 1em;
}

div {
  font-size: 0.75em;
}

p {
  font-size: 0.75em;
}
```

3/4 du parent -> 12 pixels

3/4 du parent -> 9 pixels

https://developer.mozilla.org/fr/docs/Learn/CSS/Building_blocks/Values_and_units

Unités CSS – rem

CSS

- 1 rem = Taille de police à 100% de l'élément root (**R**oot **EM**)

- Si pas d'élément parent, 1 rem est égal à la hauteur d'une lettre en taille standard, selon la résolution d'écran

https://developer.mozilla.org/fr/docs/Learn/CSS/Building_blocks/Values_and_units

Unités CSS – rem

CSS

```
<body>

  <div>

    <p>Hello</p>

  </div>

</body>
```

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

```
body {
  font-size: 1rem;
}

div {
  font-size: 0.75rem;
}

p {
  font-size: 0.75rem;
}
```

3/4 du root -> 12 pixels

3/4 du root -> 12 pixels

https://developer.mozilla.org/fr/docs/Learn/CSS/Building_blocks/Values_and_units

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

https://developer.mozilla.org/fr/docs/Learn/CSS/Building_blocks/Values_and_units

Unités CSS - rem

CSS

```css
body {
  font-size: 16px;
}

h1 {
  font-size: 2.5rem;
  margin-bottom: 1rem;
}

h2 {
  font-size: 2rem;
  margin-bottom: 0.75rem;
}

…
```

https://developer.mozilla.org/fr/docs/Learn/CSS/Building_blocks/Values_and_units

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

https://developer.mozilla.org/fr/docs/Learn/CSS/Building_blocks/Values_and_units

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

https://developer.mozilla.org/fr/docs/Web/CSS/Using_CSS_custom_properties

Variables CSS

CSS

Déclaration

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

Utilisation

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

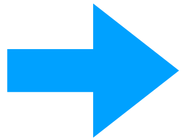
https://developer.mozilla.org/fr/docs/Web/CSS/Using_CSS_custom_properties

Variables CSS - Portée

CSS

Exemple

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

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



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

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

https://developer.mozilla.org/fr/docs/Web/CSS/Using_CSS_custom_properties

Variables CSS - Déclaration idéale

CSS

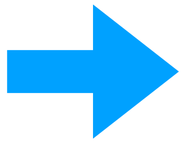
```css
/* Utiliser le pseudo element :root */

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

https://developer.mozilla.org/fr/docs/Web/CSS/Using_CSS_custom_properties

Variables CSS - Variable d'une variable

CSS

```css
:root {
  --primary-color: #00ff00;

  --link-color: var(--primary-color)
}
```



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

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

https://developer.mozilla.org/fr/docs/Web/CSS/Using_CSS_custom_properties

Pseudo-classes

CSS

- Beaucoup de pseudo-classes…
- Les historiques `:hover, :active, :visited`
- Les game-changes `:has, :not, :last-child, :valid, :user-valid, :checked, …`

https://developer.mozilla.org/fr/docs/Web/CSS/Pseudo-classes

Pseudo-classes

CSS

```css
a {
  color: #000000;
}

a:hover {
  color: #00ff00;
}
```

Lien

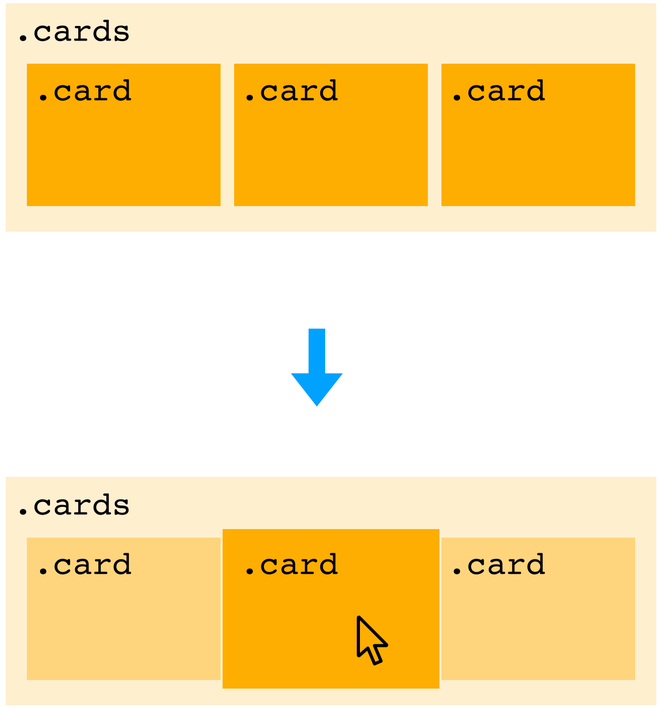
Lien

https://developer.mozilla.org/fr/docs/Web/CSS/Pseudo-classes

Pseudo-classes

CSS

```css
.cards .card:hover {
  transform: scale(1.1);
}

.cards:has(.card:hover) .card:not(:hover) {
  opacity: 0.4;
}
```



https://developer.mozilla.org/fr/docs/Web/CSS/Pseudo-classes

Transitions CSS

CSS

- Tiens sur une ligne ! Animations du flemmard !

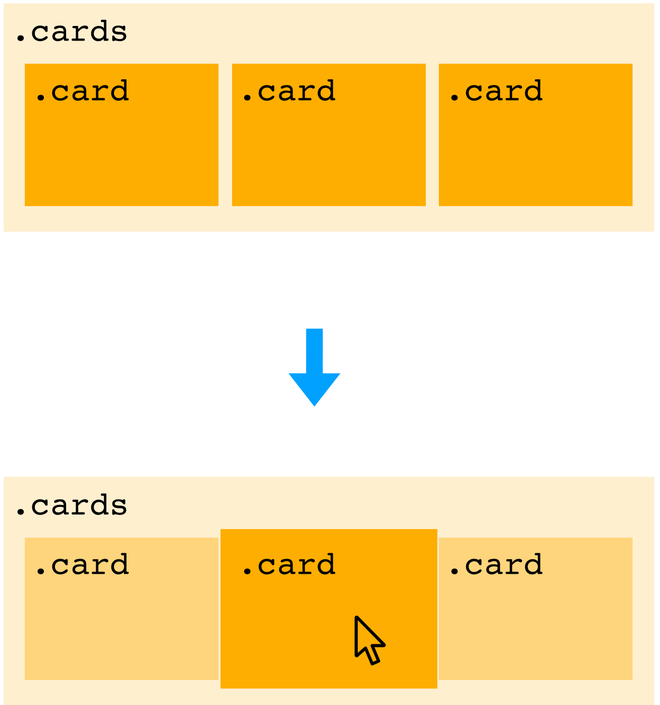
- Il suffit de dire :
  - Quoi
  - Combien de temps
  - Comment

https://developer.mozilla.org/fr/docs/Web/CSS/Pseudo-classes

Transitions CSS

CSS

```css
.cards .card {
  transition: all 500ms ease-in-out;
   // ou
  transition: transform 500ms ease-in-out, opacity 500ms ease-in-out;
}
.cards .card:hover {
  transform: scale(1.1);
}
.cards:has(.card:hover) .card:not(:hover) {
  opacity: 0.4;
}
```



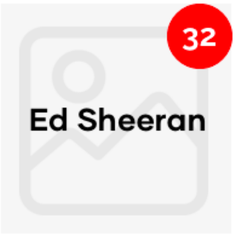
https://developer.mozilla.org/fr/docs/Web/CSS/Pseudo-classes

Data attributes

CSS

```
<div
  data-id=“1”
  data-name="Ed Sheeran"
  data-count="32"
  data-image=“https://…"
>
</div>
```

```
div {
  /*___/!\ Bientôt…___*/
  background-image: attr(data-image);


  /*___Possible! Content only ___*/
  content: attr(data-name);
}

div:before {
  /*___Possible! Content only ___*/
  content: attr(data-count);


  position: absolute;
  top: 0;
  right: 0;
  border-radius: 50%;
  background-color: #ff0000;
  color: #fff;
  text-align: center;
}
```



Viewport

CSS

- Le viewport est la 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

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

Viewport

CSS

Viewport

View

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.

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

Viewport – La balise meta viewport

CSS

Zoom par défaut

Pas zoomable par l'utilisateur

```
<meta
  name="viewport"
  content="width=device-width, initial-scale=1, maximum-scale=1, user-scalable=0"
/>
```

La largeur par défaut est
celle de l'écran à 100%
-> pas de zoom

Le zoom maximum est de 1

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

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

https://developer.mozilla.org/fr/docs/Web/CSS/Media_Queries/Using_media_queries

Media queries - Blocs CSS

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
```

https://developer.mozilla.org/fr/docs/Web/CSS/Media_Queries/Using_media_queries

Media queries - Blocs CSS

CSS

Exemple 1

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

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

Exemple 2

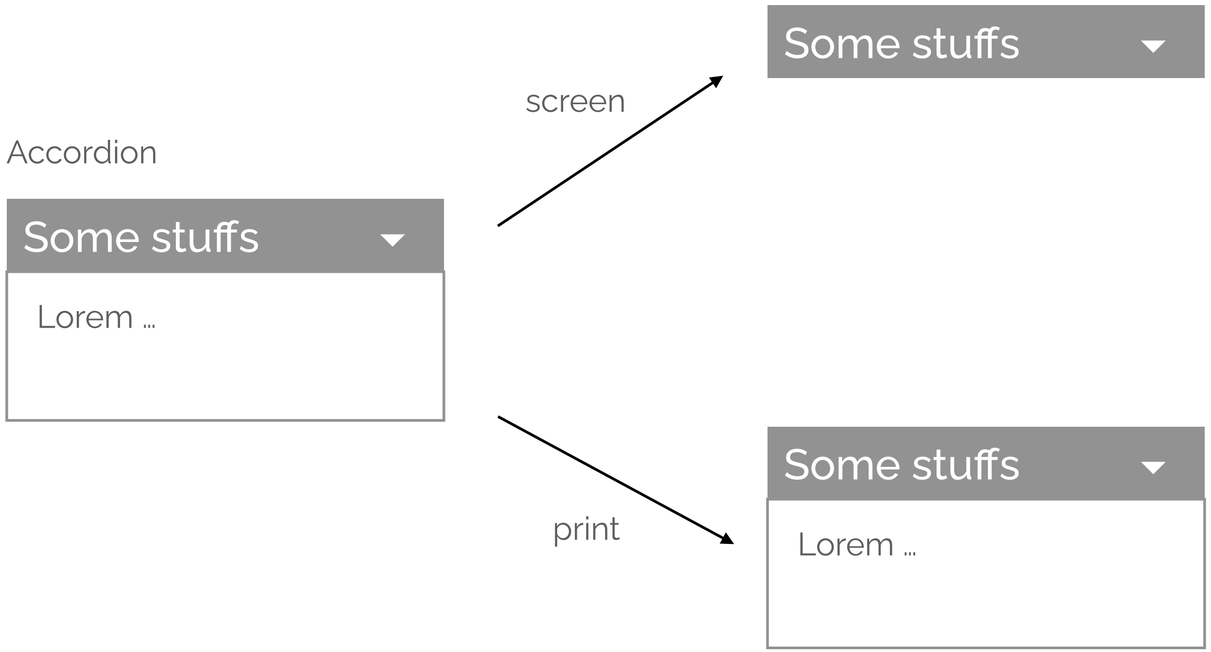
```
body {
  background-color: white;
  …
}

@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;
  }
}
```



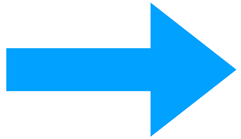
Media queries - Blocs CSS

CSS

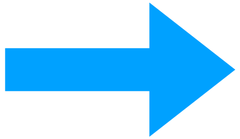
Cumul des règles par qualification des sélecteurs

```css
.container {
  display: flex;
  flex-direction: row;
  …
}

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

Sélecteur interprété comme



```css
.container {
  display: flex;
  flex-direction: row;
  …
}

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



Media queries - Blocs CSS

CSS

Cumul des règles par qualification des sélecteurs

```css
.container {
  display: flex;
  flex-direction: row;
  …
}

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

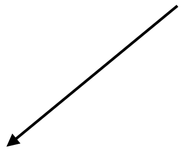
Comparable à une sur-qualification du type

```css
.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 ?

https://developer.mozilla.org/fr/docs/Web/CSS/Media_Queries/Using_media_queries

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

https://developer.mozilla.org/fr/docs/Web/CSS/Media_Queries/Using_media_queries

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 */
}
```

https://developer.mozilla.org/fr/docs/Web/CSS/Media_Queries/Using_media_queries

Media queries - Javascript

CSS

Ajouter un listener sur une media query

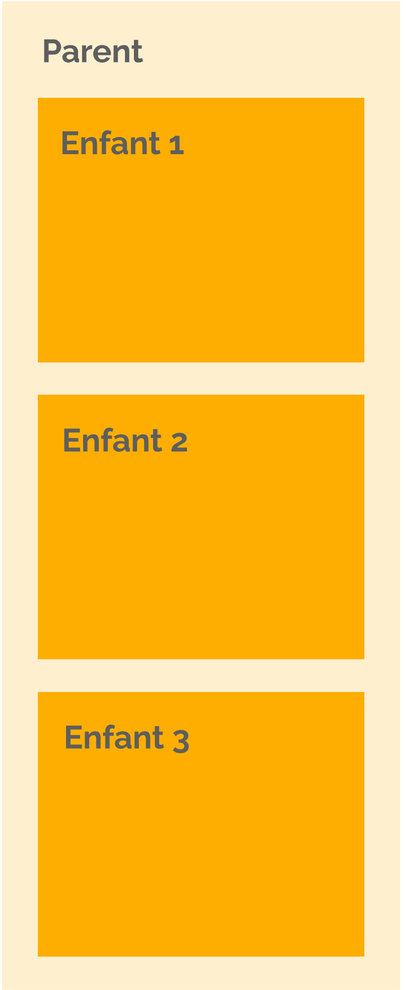
```
const mql = window.matchMedia("(orientation: portrait)”);
mql.addListener(handleOrientationChange);

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

https://developer.mozilla.org/fr/docs/Web/CSS/Media_Queries/Using_media_queries

Flexboxes

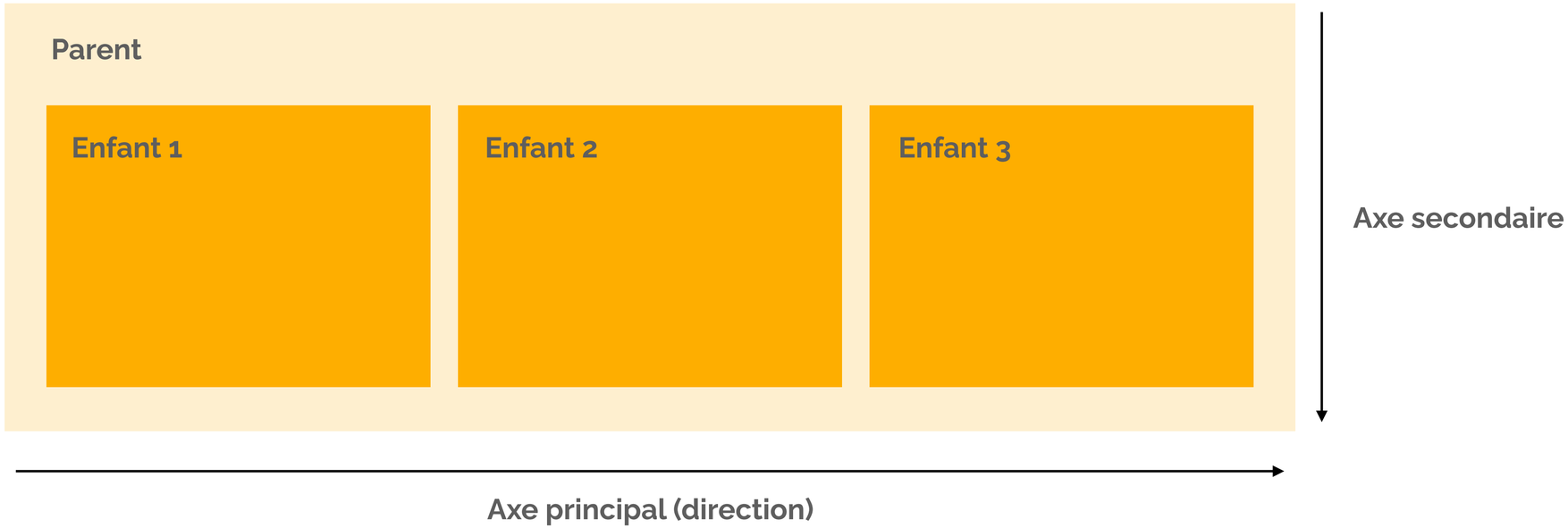
CSS

- 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"



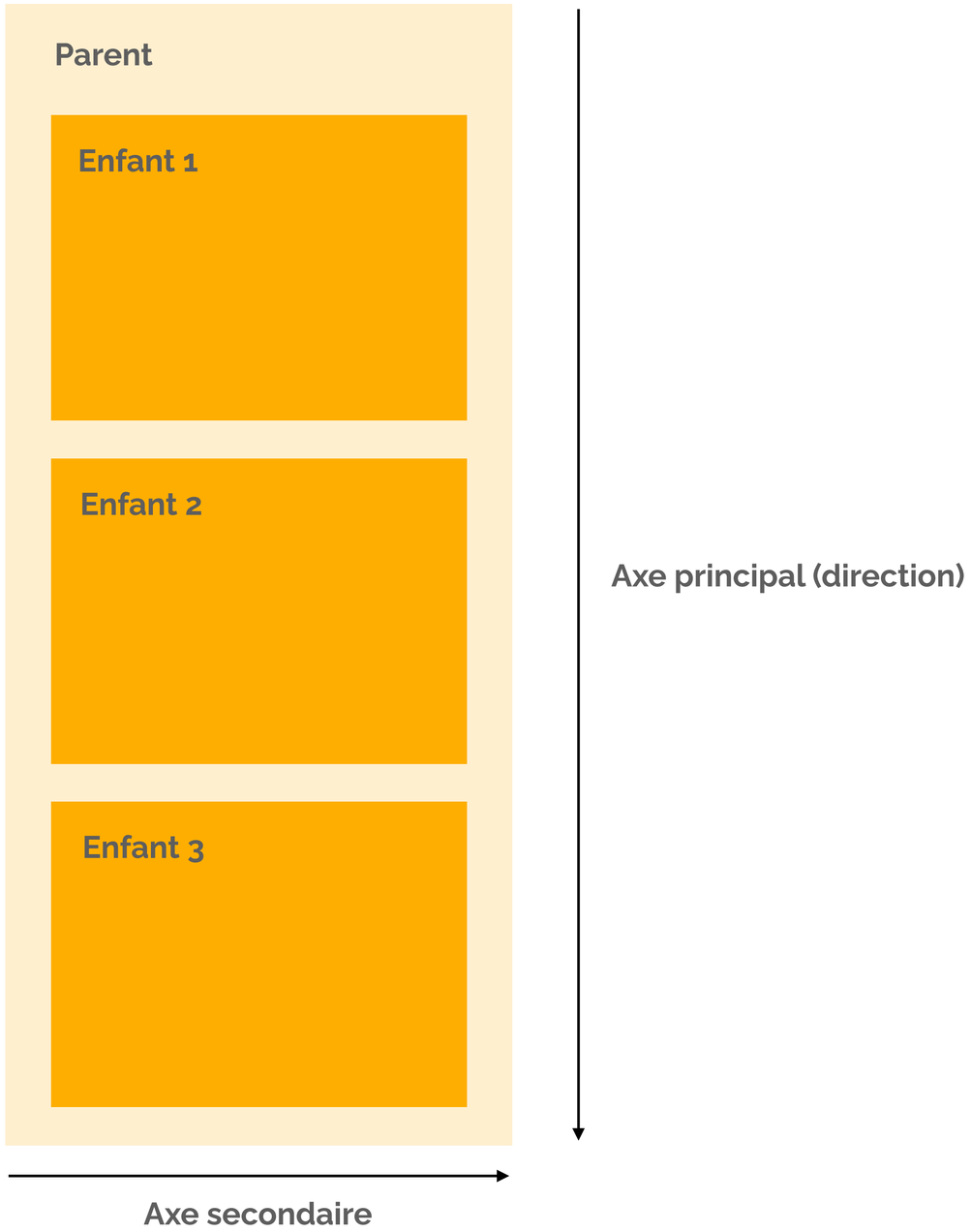
Flexboxes

CSS



Flexboxes

CSS



Flexboxes – Popriétés

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`)

https://developer.mozilla.org/fr/docs/Web/CSS/CSS_Flexible_Box_Layout/Basic_Concepts_of_Flexbox
https://css-tricks.com/snippets/css/a-guide-to-flexbox/

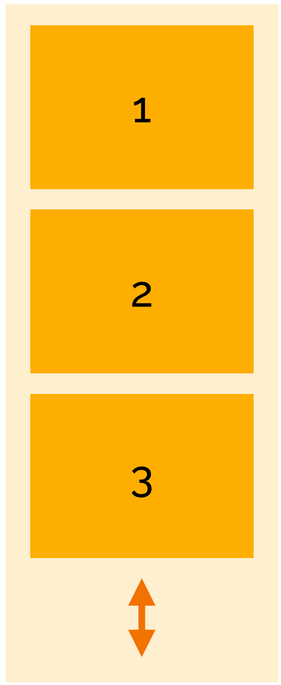
Flexboxes - Direction (flex-direction)

CSS

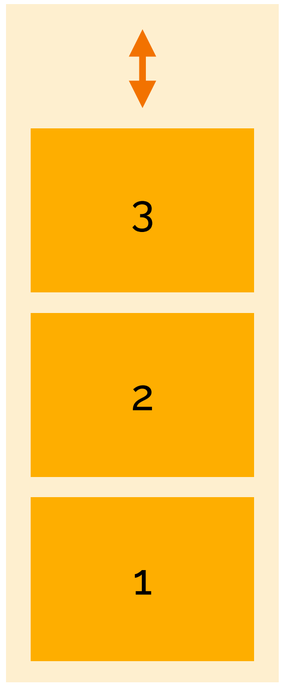
flex-direction: row; (default)

flex-direction: row-reverse;

flex-direction: column;



flex-direction: column-reverse;



Flexboxes – Direction rtl

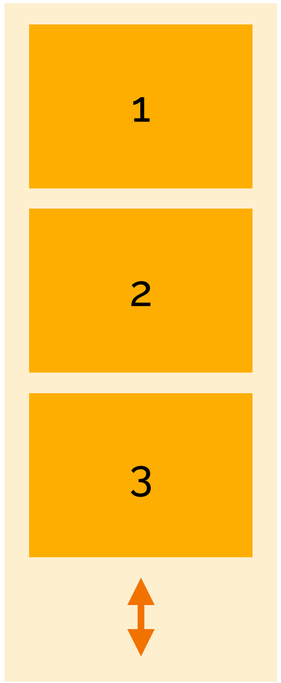
CSS

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

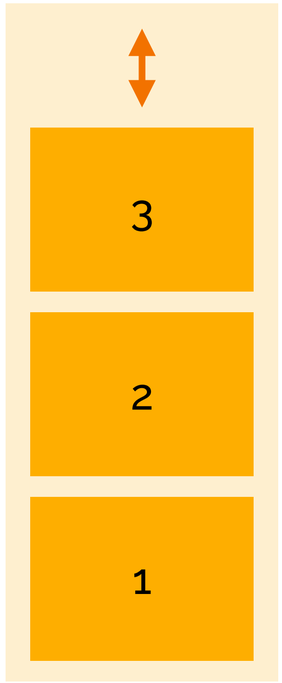
`flex-direction``: row;`

`flex-direction``: row-reverse;`

`flex-direction``: column;`



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

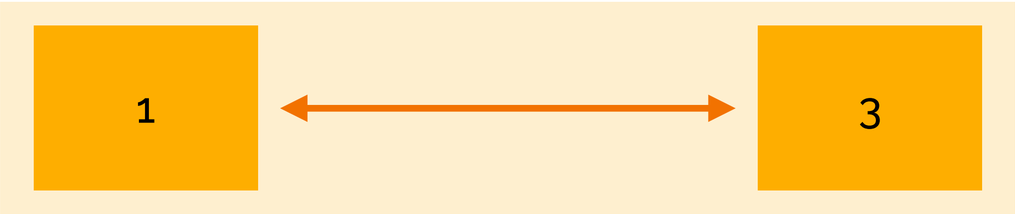


Flexboxes - Alignement axe principal (justify-content)

CSS

justify-content: flex-start; (default)

justify-content: space-between;



justify-content: flex-end;

justify-content: space-around;

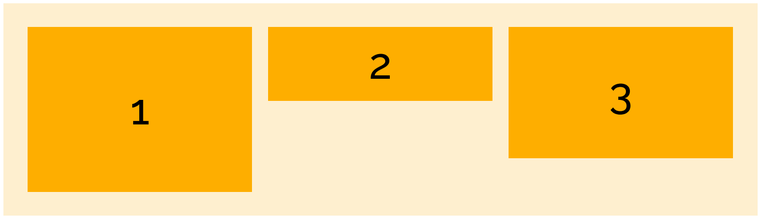
justify-content: center;

justify-content: space-evenly;

Flexboxes - Alignement axe secondaire (align-items)

CSS

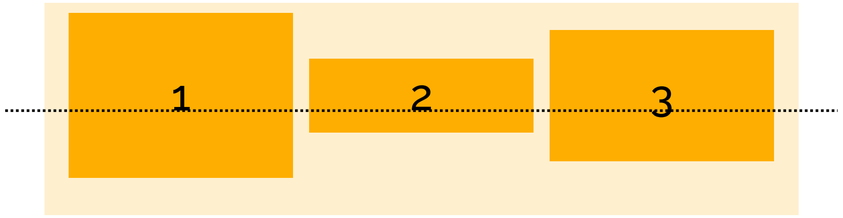
align-items: flex-start;



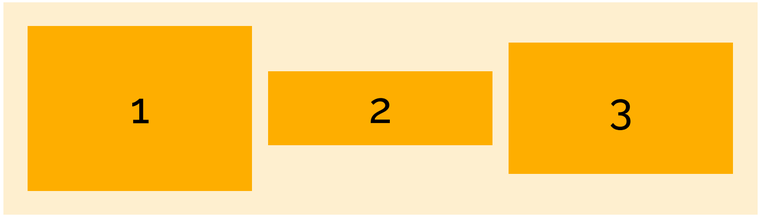
align-items: stretch; (default)

align-items: flex-end;

align-items: baseline;



align-items: center;

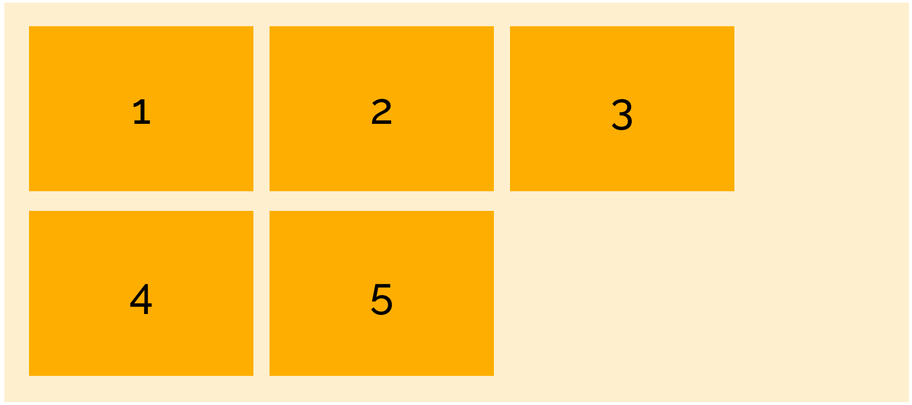


Flexboxes - Multi-lignes (wrap)

CSS

flex-wrap: nowrap; (default)

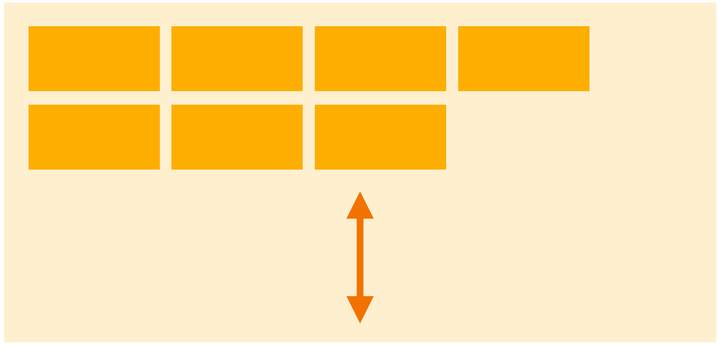
flex-wrap: wrap;



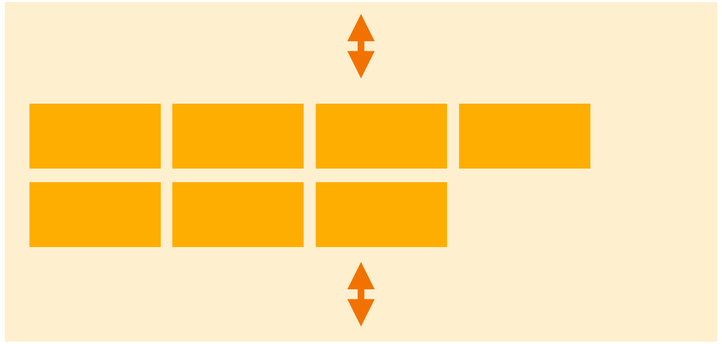
Flexboxes - Alignement multi-lignes (align-content)

CSS

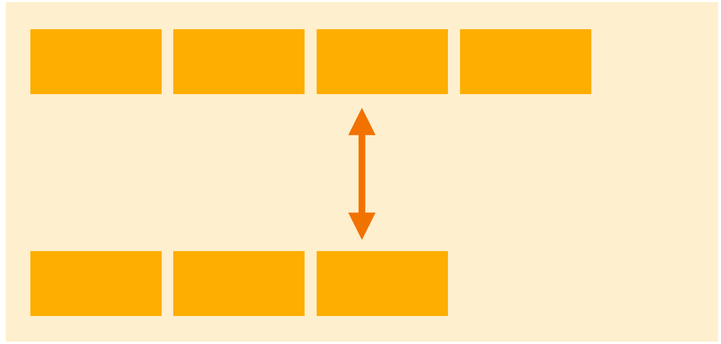
`align-content: flex-start;`



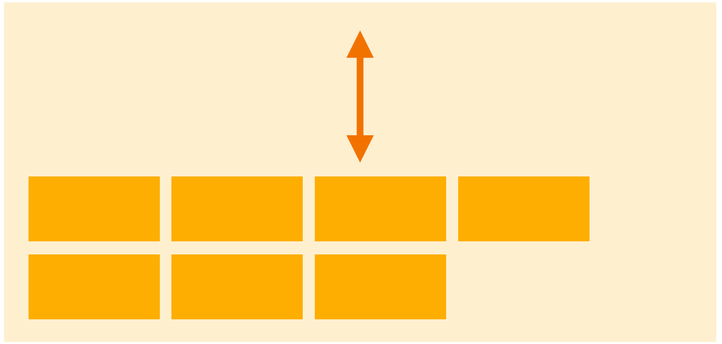
`align-content: center;`



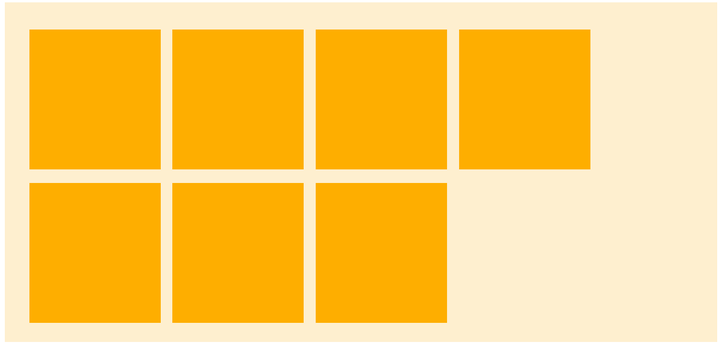
`align-content: space-between;`



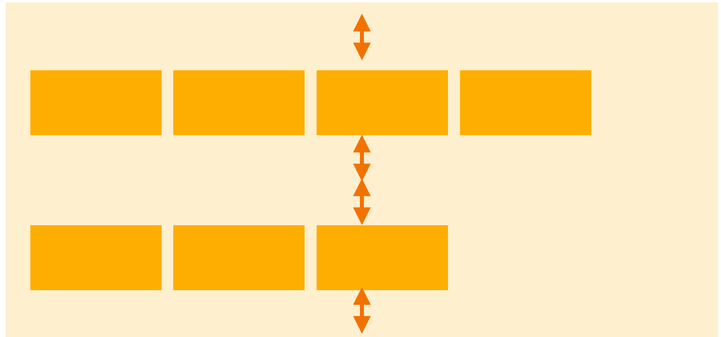
`align-content: flex-end;`



`align-content: stretch;`



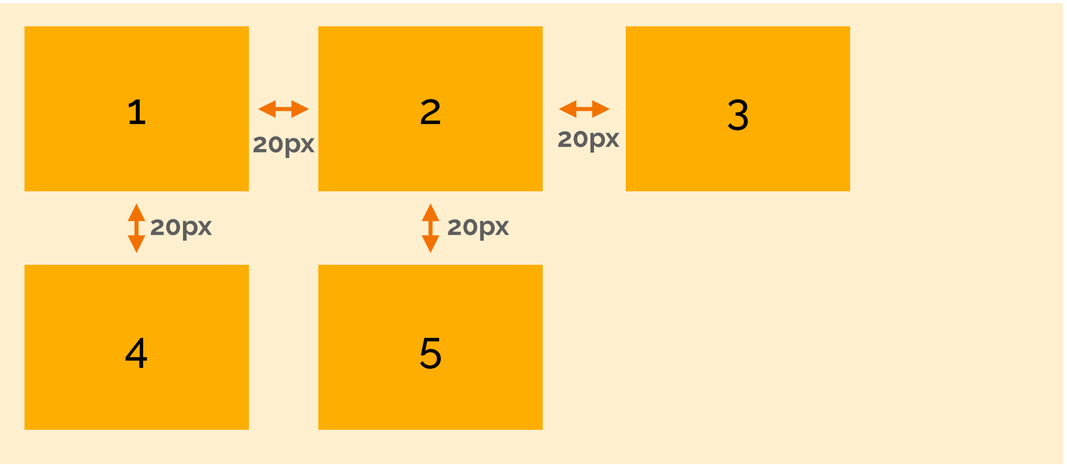
`align-content: space-around;`



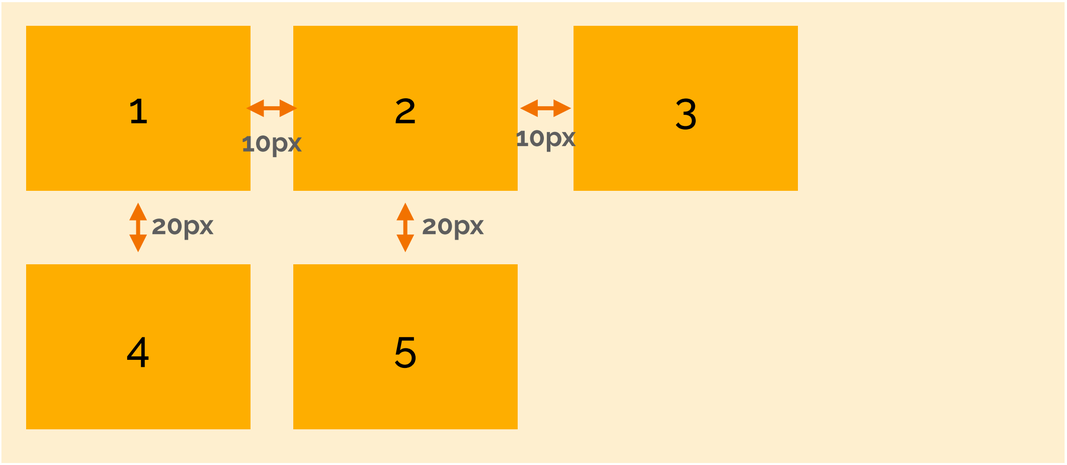
Flexboxes – Ecartement (gap)

CSS

`gap: 20px;`

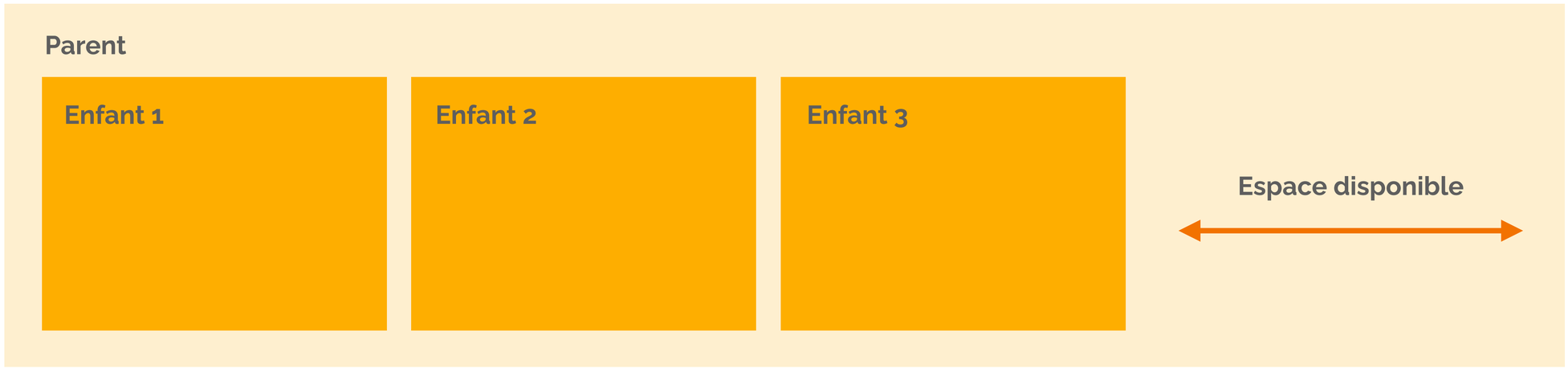


`gap: 20px 10px;`



Flexboxes – Sizing

CSS



Flexboxes – Sizing & responsiveness

CSS

flex-basis: [px|%|auto|…];

Rétrécissement

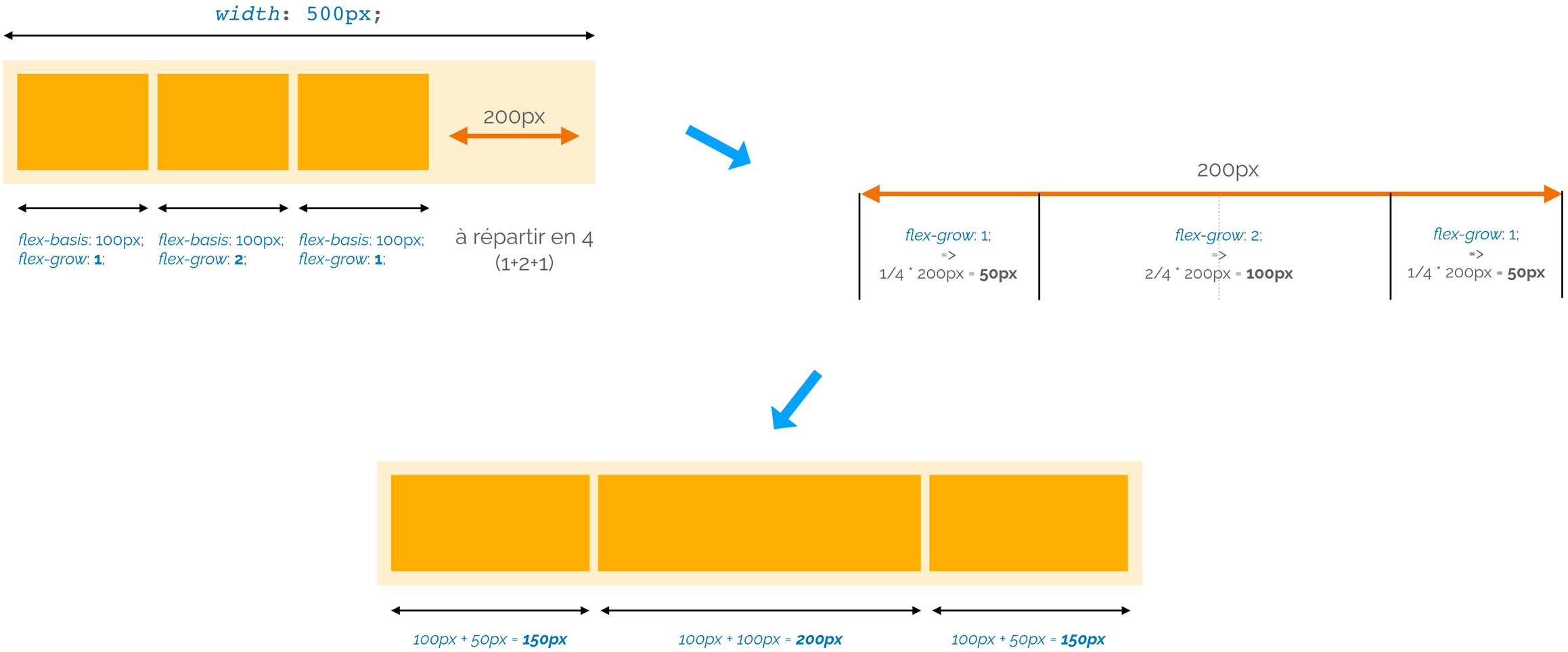
Agrandissement

flex-shrink: entier;

flex-grow: entier;

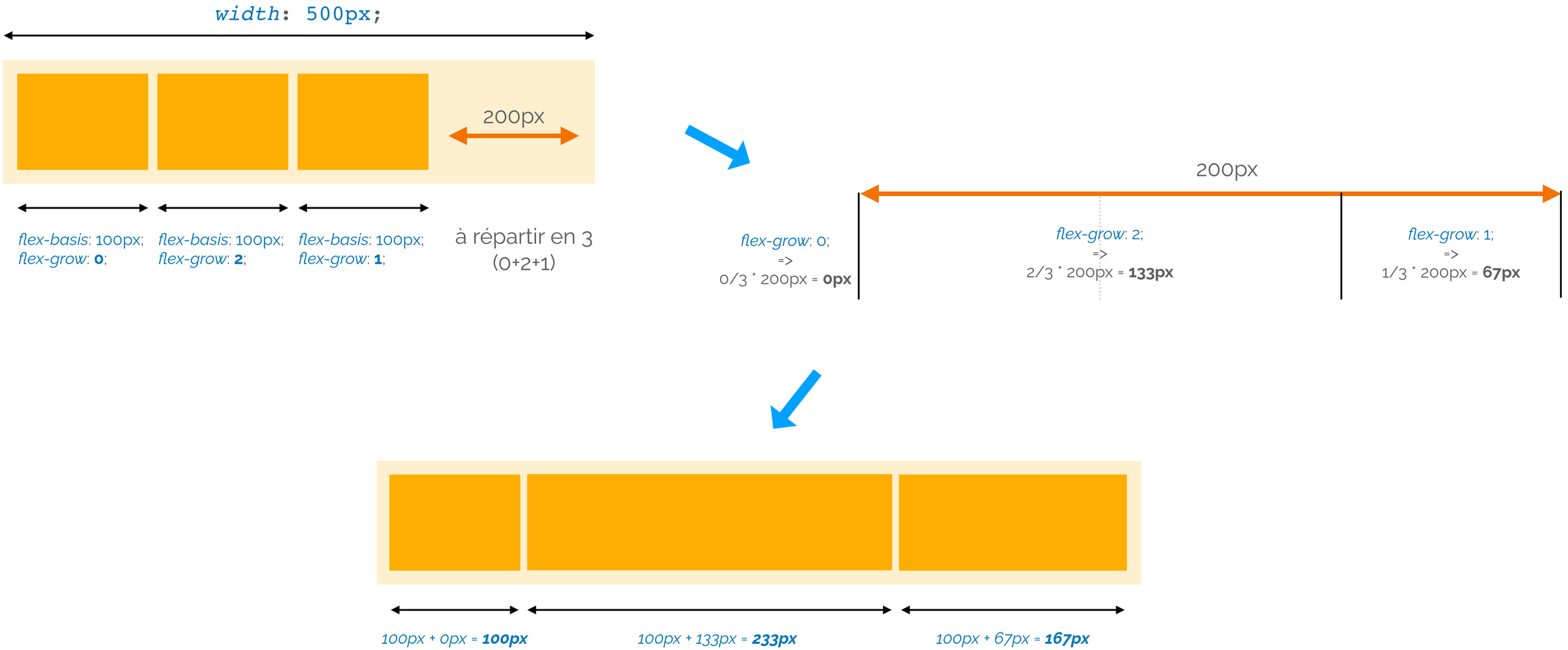
Flexboxes – Sizing flex-grow

CSS



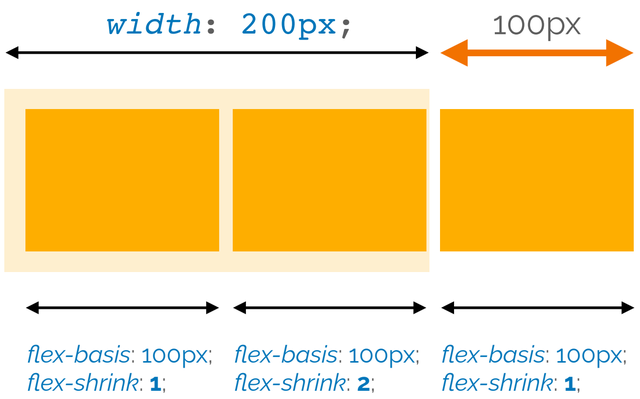
Flexboxes – Sizing flex-grow

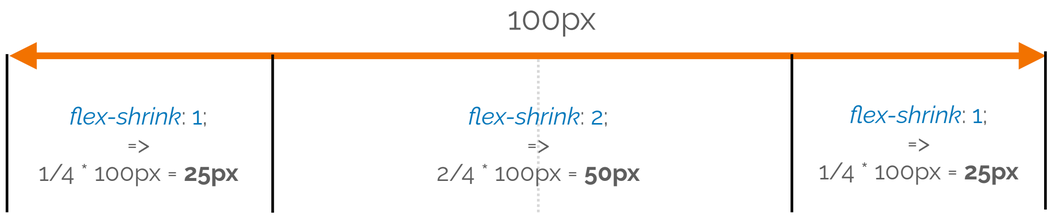
CSS

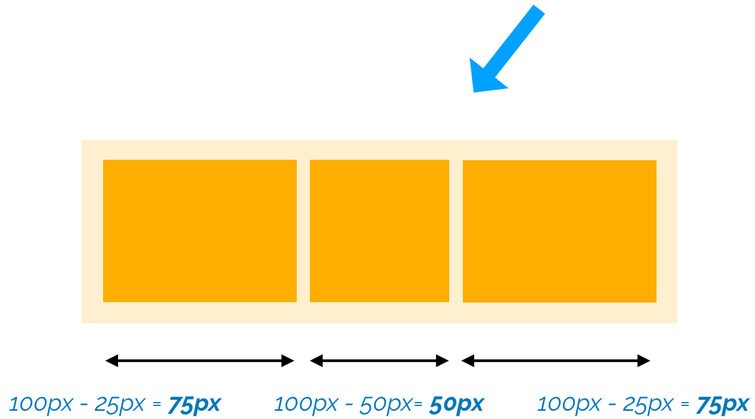


Flexboxes - Sizing flex-shrink

CSS







Flexboxes – Sizing flex-basis & sizing

CSS

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

Flex-grow vaut 0 par défaut

Flex-shrink vaut 0 par défaut

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

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

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`)

https://developer.mozilla.org/fr/docs/Web/CSS/CSS_Flexible_Box_Layout/Basic_Concepts_of_Flexbox
https://css-tricks.com/snippets/css/a-guide-to-flexbox/

Flexboxes – Order

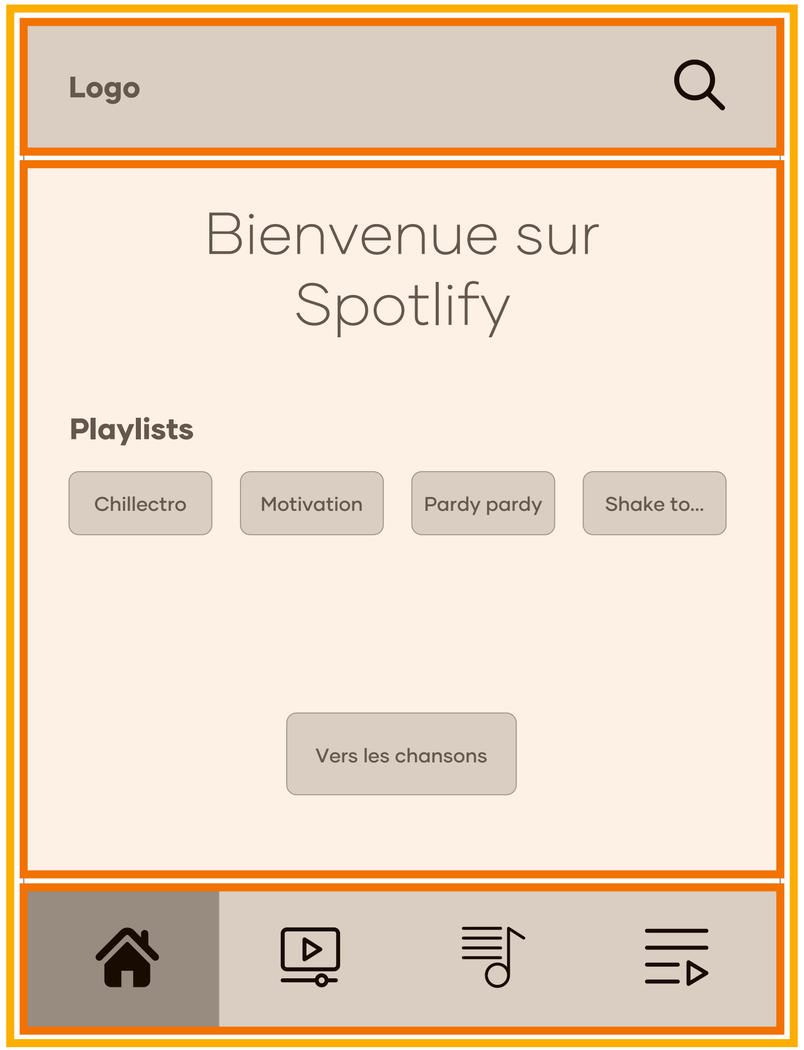
CSS

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

https://developer.mozilla.org/fr/docs/Web/CSS/CSS_Flexible_Box_Layout/Basic_Concepts_of_Flexbox
https://css-tricks.com/snippets/css/a-guide-to-flexbox/

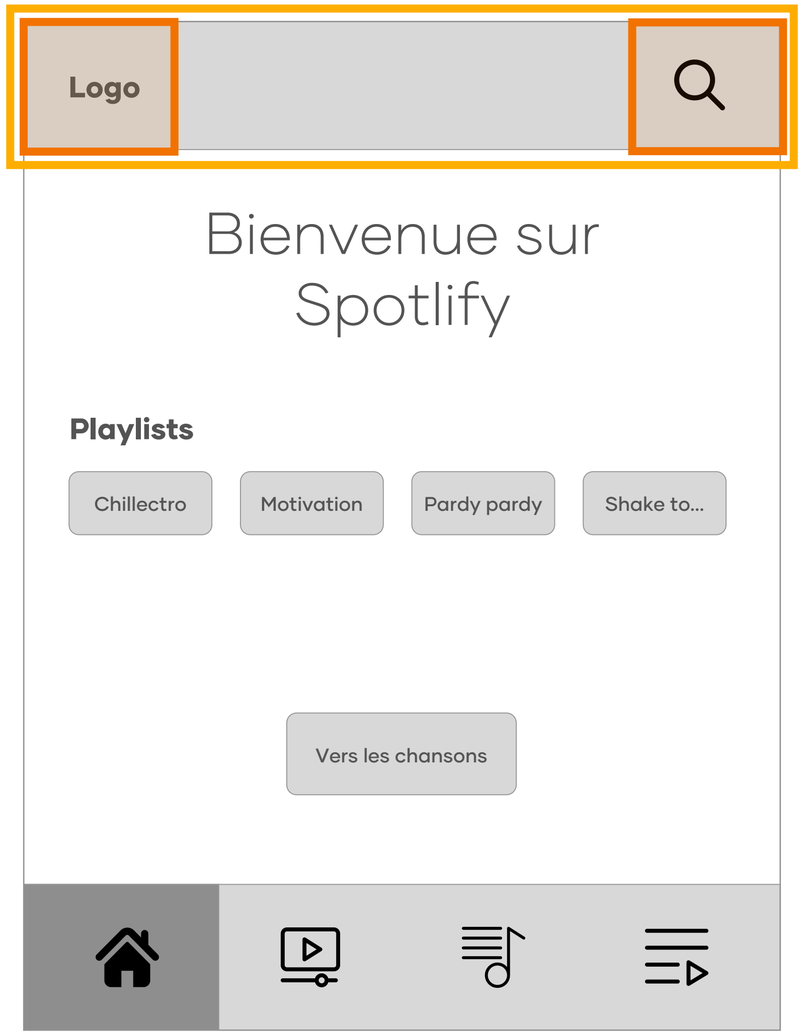
Flexboxes dans le projet

CSS



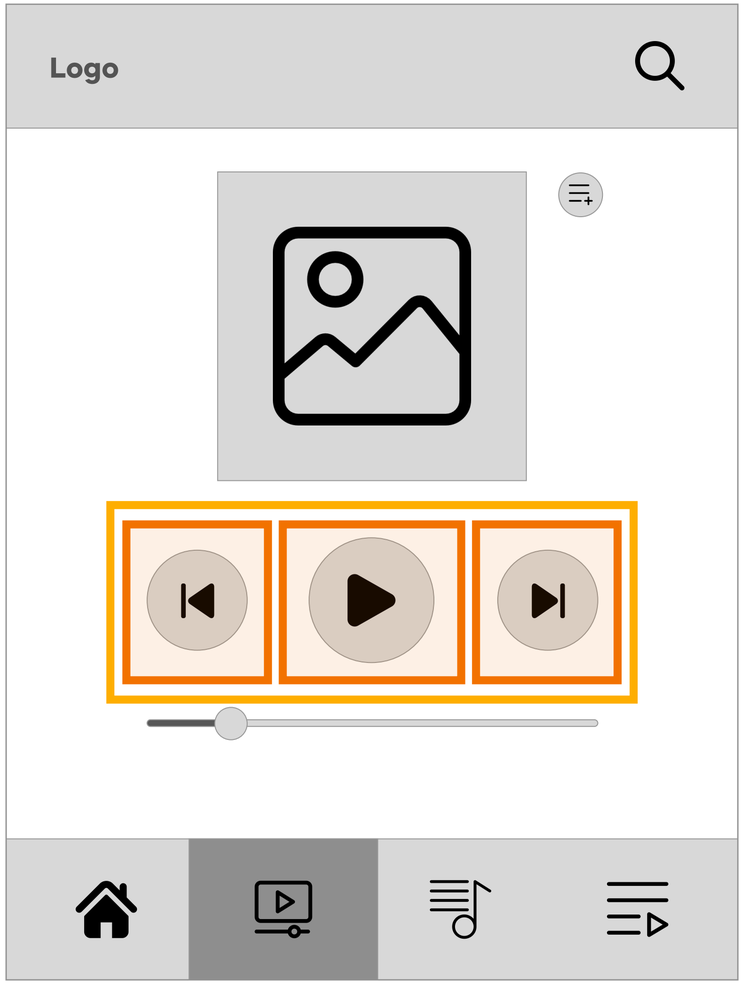
Header

CSS


Logo
Bienvenue sur Spotlify
Playlists
Chillectro
Motivation
Pardy pardy
Shake to...
Vers les chansons

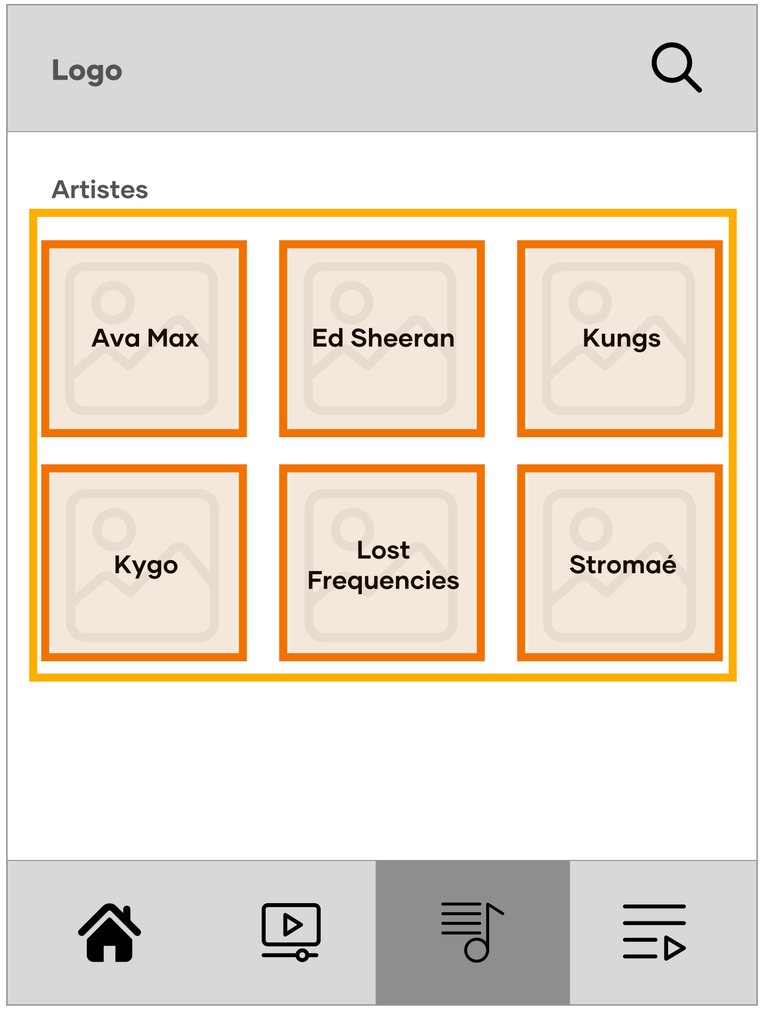
Flexboxes dans le projet

CSS



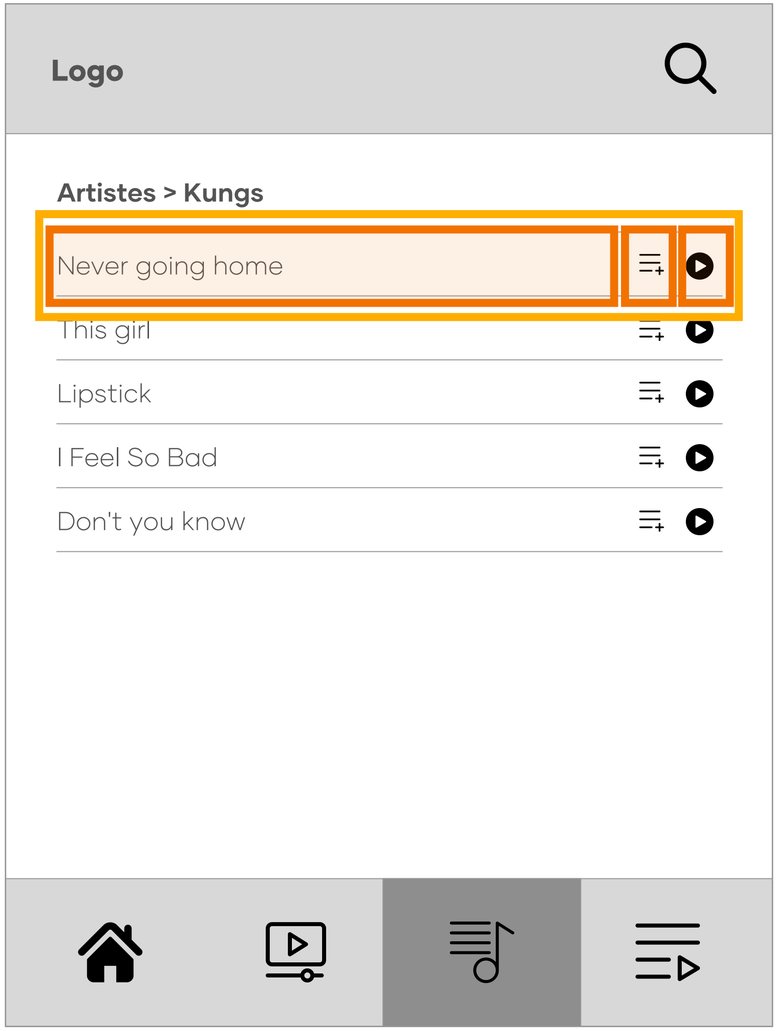
Flexboxes dans le projet

CSS



Flexboxes dans le projet

CSS



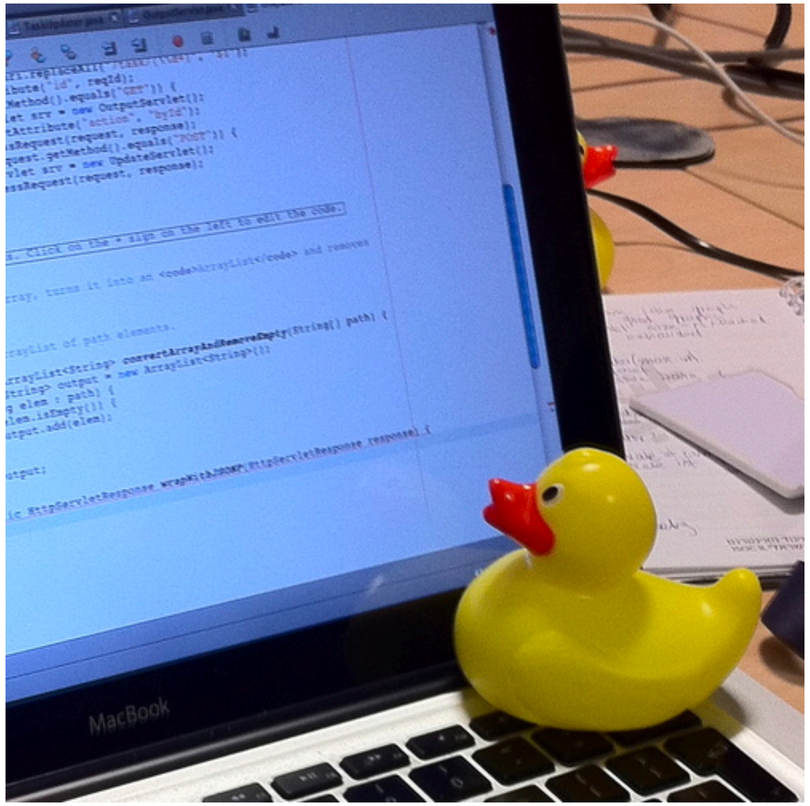
Goal

HTML/CSS

- Grâce aux flexboxs, réaliser le squelette vu ensemble
- La structure doit ressembler à la home page du prototype
- Démarrer le design global de l'app

Rubber duck debugging

Tips & tricks



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