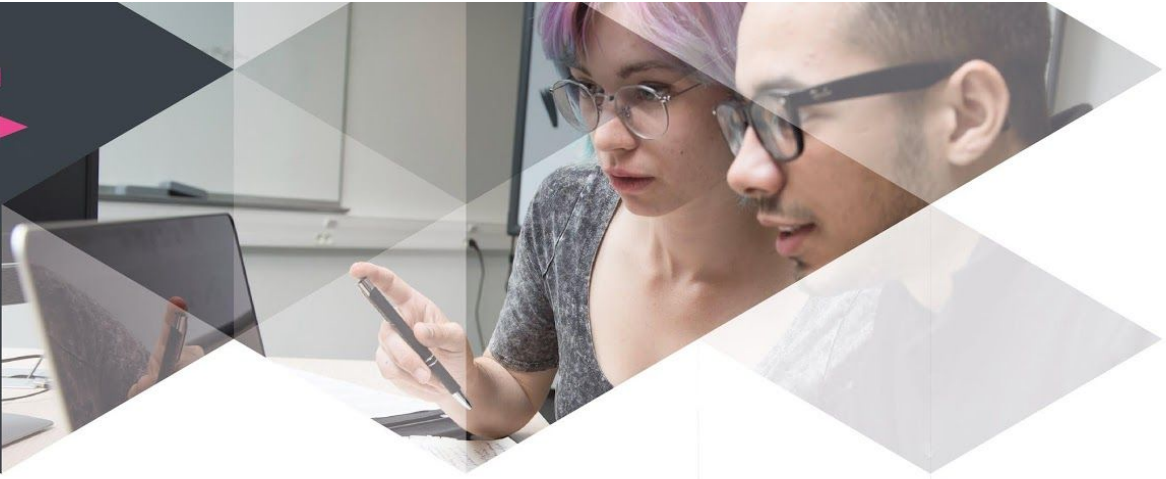




le
campus
numérique
in the ALPS



Découverte de Vue.js

Vous avez vu qu'il est possible d'intégrer Vue.js à différents environnements.

Dans un premier temps, nous découvrirons les bases de Vue.js, sans aucun outil de build.

Ensuite nous mettrons en place un projet buildé, qui sera beaucoup plus efficace que de développer dans l'émulateur android avec Cordova :



Mise en place outils de développement VueJS.

Modalités

- Travail en îlot, en autonomie
- Production individuelle

Objectif (compétences)

- Mettre en place un environnement de dev pour VueJS

Consignes

- Configurez votre éditeur de code, si besoin
- Configurez votre navigateur web pour pouvoir développer avec VueJS : ajoutez l'extension chrome Vue DevTools.

Ressources

- Extension navigateur : <https://github.com/vuejs/vue-devtools>

Livrable

- ☐ Votre PC Configuré

Découverte de Vue.js

Pour commencer, nous allons prendre en main Vue.js en le chargeant, simplement dans notre page, comme une librairie externe, comme on le fait pour jQuery

Modalités

- Travail en îlot, en autonomie
- Production individuelle

Objectif (compétences)

- Découvrir Vue.js avec des fonctionnalités simples
- Constater la différence de logique entre jQuery et Vue.js

Consignes

- Découvrez les différences entre jQuery et Vue.js en comparant ces exemples simples :

- Capturer un texte entré dans un champ de saisie : [jQuery](#) | [Vue.js](#)
 - Capturer un texte entré lors d'un évènement : [jQuery](#) | [Vue.js](#)
 - Ajouter / Enlever une classe sur un élément : [jQuery](#) | [Vue.js](#)
 - Afficher / Cacher un élément : [jQuery](#) | [Vue.js](#)
-
- Chargez la librairie Vue depuis un fichier externe, comme vous le feriez pour jQuery (utilisation d'un CDN)
 - Créez directement dans un fichier html votre première application Vue.
 - Rendez votre application **réactive** : Affichez du texte qui sera renseigné en javascript, dans l'objet data de Vue.
 - Modifiez la valeur de votre data en Javascript dans la console de votre devtools.
 - Modifiez la valeur de votre data dans l'extension Vue Devtools.

Ressources

- La **documentation officielle** de Vue est entièrement traduite en français et très bien rédigée : <https://fr.vuejs.org/v2/guide/index.html>
- Hello world : <https://jsfiddle.net/romainpetit/36g8n1h7>
- Tutoriel vidéo par GrafikArt : <https://www.youtube.com/watch?v=XkgiXngcpWk>
- Les très bons [exemples officiels](#)

Livrable

Un projet en local utilisant Vue contenant :

- ☐ Un champs INPUT permettant de mettre à jours le contenu d'une balise <h1></h1>
- ☐ Un champs INPUT permettant de mettre à jours le contenu d'une balise <h2></h2> au clic sur un bouton
- ☐ Une checkbox permettant d'ajouter la classe "red" au <h1> et <h2> quand elle est check
- ☐ Une checkbox pour afficher / masquer une image

Mise en place d'un projet buildé avec vue-cli

La structure du projet va changer un peu : on renseignera maintenant notre data dans l'instance racine de Vue, dans le fichier main.js.

Modalités

- Travail en îlot, en autonomie
- Production individuelle

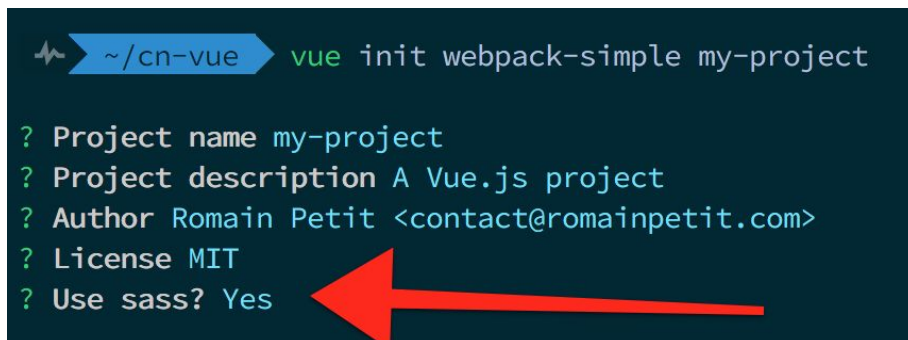
Objectif (compétences)

- Séparer le code JS du code HTML en créant des composants
- Stocker la donnée de sa page dans l'objet data de Vue

Consignes

- Assurez vous que Node soit en version > 8 et Npm sont bien installés
- Installez le Vue CLI
- Initialisez un projet avec Vue CLI **avec Sass** avec :

`vue init webpack-simple <nom_projet>`



```
~/cn-vue ➤ vue init webpack-simple my-project

? Project name my-project
? Project description A Vue.js project
? Author Romain Petit <contact@romainpetit.com>
? License MIT
? Use sass? Yes
```

- Lancez le serveur local pour votre nouveau projet (et découvrez que vous n'avez plus besoin de recharger votre page web pour voir vos modifs !)
- Affichez deux machines à café, une inactive, et une active.
- Poussez votre projet avec git dans un nouveau repo github

Ressources

- **Documentation officielle** : Vue CLI <https://vuejs.org/v2/guide/installation.html#CLI>
- <https://github.com/vuejs/vue-cli/blob/dev/docs/cli.md>
- Notre template utilisé : <https://github.com/vuejs-templates/webpack-simple>

- Tuto GrafikArt : Vue CLI <https://www.grafikart.fr/formations/vuejs/vue-cli>
- Tuto GrafikArt : Webpack https://www.youtube.com/watch?v=_KXGVca8uXw

Livable

- ❑ Votre projet Vue, poussé sur github

Utiliser des composants

On a maintenant un projet en place, prêt pour gérer notre nouveau parc de machines à café.

Nous allons mettre en place un système simple pour lister des machines et les allumer ou les éteindre.

Modalités

- Travail en îlot, en autonomie
- Production individuelle

Objectif (compétences)

- Créer un composant et en comprendre l'utilité
- Passer de la donnée aux composants
- Boucler sur de la data avec Vue
- Ajouter une dépendance Vue avec NPM

Consignes

- Renseignez dans votre data votre liste de machines sous forme d'une collection **machines**
- Affichez dans la vue principale la liste des machines depuis votre data
- Donnez à chaque machine un nom, et un statut.
- Ajoutez un composant externe avec NPM : un toggle <https://github.com/euwl/vue-js-toggle-button>
- Donnez vie à ce toggle : il doit pouvoir afficher le statut de chaque machine, mais aussi changer ce statut au clic.
- Créez un composant Machine qui va afficher le statut **d'une** machine à café, et le toggle pour changer ce statut.

Ressources

- Style Guide : comment écrire du Vue.js | Priorité 1
<https://vuejs.org/v2/style-guide/#Priority-A-Rules-Essential-Error-Prevention>
- <https://github.com/euwl/vue-js-toggle-button>

Livable

- ☐ Commit 1 : Lister les machines depuis la data
- ☐ Commit 2 : Lister les machines en tant que composants

Rendu final :

Liste des machines

Machine 1 ☒

Machine 2 ☐

Machine 3 ☒