Brice Chaponneau

Vue.js

Applications web complexes et réactives

Éditions EYROLLES

Le framework JavaScript concurrent de React et Angular

Vue.js

Un ouvrage de référence pour le développeur web

Vue.js est un framework JavaScript orienté front-end qui mérite considération à plusieurs égards. Il est réactif, performant, versatile, facilement testable, maintenable et sa courbe d'apprentissage est réellement rapide.

L'écriture globale est idéalement structurée et son écosystème aide à créer, organiser et maintenir vos applications clientes.

Ce framework peut se suffire à lui-même pour développer des applications complexes en ayant recours à de simples composants, des mixins ou des plug-ins. De plus, il s'accompagne d'un univers où de multiples outils sont disponibles pour aider au développement : des extensions, des plug-ins et des librairies complètes pour vous faire gagner en temps de réalisation, en qualité de code et également en performance.

Compléments web

Tous les exemples des programmes du livre sont en téléchargement sur notre site Internet : www.editions-eyrolles.com/dl/0067783.

À qui s'adresse cet ouvrage ?

- Aux développeurs et chefs de projet web qui souhaitent réaliser des applications web performantes.
- À toutes les personnes qui souhaitent découvrir Vue.js et acquérir des connaissances certaines afin d'être autonomes dans le développement web autour de ce framework.

Au sommaire

Brice Chaponneau est titulaire d'un Master IT dans le développement des applications réparties. Il a travaillé dans différents domaines dont la banque (Caisse d'Épargne, Société Générale, Edmond de Rothschild, Natixis), l'assurance (Monceau Assurance), les transports (SNCF) et le secteur industriel (ArcelorMittal). Brice a donc été développeur, lead technique, chef de projet MOE, Scrum Master et consultant. Il a majoritairement développé en JavaScript, .Net et via des frameworks divers. Il maîtrise les bases de données NoSQL comme Mongo DB et SGBDR (SQL Server, Oracle, Sybase).

www.editions-eyrolles.com

Vue.js

DANS LA MÊME COLLECTION

S. Ringuedé. – **SAS.**
N° 67631, 2019, 688 pages.

M. Bidault. – **Programmation Excel avec VBA.**
N° 67786, 2019, 512 pages.

R. Goetter. – **CSS 3 Grid Layout.**
N° 67683, 2019, 131 pages.

C. Blaess. – **Solutions temps réel sous Linux.**
N° 67711, 3e édition, 2019, 318 pages.

C. Pierre De Geyer, J. Pauli, P. Martin, E. Daspet. – **PHP 7 avancé.**
N° 67720, 2e édition, 2018, 736 pages.

H. Wickham, G. Grolemund. – **R pour les data sciences.**
N° 67571, 2018, 496 pages.

F. Provost, T. Fawcett. – **Data science pour l'entreprise.**
N° 67570, 2018, 370 pages.

J. Chokogoue. – **Maîtrisez l'utilisation des technologies Hadoop.**
N° 67478, 2018, 432 pages.

H. Ben Rebah, B. Mariat. – **API HTML 5 : maîtrisez le Web moderne !**
N° 67554, 2018, 294 pages.

W. McKinney. – **Analyse de données en Python.**
N° 14109, 2015, 488 pages.

E. Biernat, M. Lutz. – **Data science : fondamentaux et études de cas.**
N° 14243, 2015, 312 pages.

SUR LE MÊME THÈME

É. Sarrion. – **React.js.**
N° 67756, 2019, 350 pages.

T. Parisot. – **Node.js.**
N° 13993, 2018, 472 pages.

C. Herby. – **Apprenez à programmer en Java.**
N° 67521, 2018, 788 pages.

Brice Chaponneau

Vue.js

Applications web complexes et réactives

Éditions
EYROLLES

ÉDITIONS EYROLLES
61, bd Saint-Germain
75240 Paris Cedex 05
www.editions-eyrolles.com

Avant-propos

Comment lire cet ouvrage ?

L'ouvrage est construit de manière incrémentale sur la compréhension et l'utilisation de Vue.js : de l'installation du cœur et des packages principaux de ce framework à la création d'un projet complet, en passant par l'ensemble de ses composantes, la préconisation d'outils et leur configuration mais également des astuces d'écriture de code ou d'optimisation d'exécution.

Que contient ce livre ?

Pour appuyer les chapitres décrivant une composante majeure de Vue, des exercices seront proposés afin de bien en comprendre les rouages. Ils mèneront à un projet final. Il est donc conseillé de lire ce livre chapitre après chapitre.

Le premier chapitre est la mise en lumière de Vue. Le deuxième chapitre expose la multitude de possibilités offertes pour installer et/ou utiliser Vue et propose un panel d'outils, ainsi que leur configuration. Les chapitres suivants décortiquent les paradigmes du framework.

À qui s'adresse-t-il ?

Vue est écrit en JavaScript (JS) et ce livre n'a pas vocation à reprendre les bases de ce langage, ni les conventions ECMAScript – même si nous y trouverons quelques rappels et comparaisons – mais à disséquer ce framework. Il est donc nécessaire que le lecteur connaisse les bases de JS, du langage HTML et des styles CSS. Notons cependant que les exemples vont à l'essentiel afin d'exposer un sujet précis et ne veulent en aucun cas perdre le lecteur dans des propositions surchargées.

Code source en téléchargement

Tous les codes sources sont également disponibles sur l'espace de téléchargement dédié sur le site des éditions Eyrolles (editions-eyrolles.com/dl/0067783) afin de permettre au lecteur de se focaliser sur la compréhension et les tests plutôt que sur la réécriture.

Un fichier nommé `Readme.html` à la racine de ce dossier de téléchargement apporte une explication simple et détaillée pour pouvoir lancer les serveurs et apprécier le code ainsi que le rendu.

Table des matières

CHAPITRE 11

CHAPITRE 12

CHAPITRE 13

CHAPITRE 14

CHAPITRE 15

CHAPITRE 16

Introduction

Vue (prononcé « view ») est un framework évolutif JavaScript front-end et open source qui permet de construire des interfaces web utilisant des liaisons de données MVVM (Modèle-Vue-Vue-Modèle) très simplement, le tout architecturé autour du composant et surtout de la réutilisabilité.

Il est possible de réaliser des composants unitaires, des applications SPA *(Single Page Application)*, SSR *(Server Side Rendering)*, Mobile… De plus, un projet Vue peut être couplé avec d'autres outils ou bibliothèques tierces.

Vue est disponible sur le site officiel du framework à l'adresse suivante : https://vuejs.org/.

Historique de Vue.js

Après avoir travaillé chez Google sur divers projets avec leur framework AngularJS, Evan You a implémenté un framework plus léger, organisé et plus modulable. La première version de Vue a été déposée sur GitHub en février 2014 et son code couvert par des tests unitaires sous Karma (bibliothèque JavaScript de tests unitaires).

Aujourd'hui, ce projet est maintenu par divers auteurs à l'international tant pour le noyau que pour les outils et modules complémentaires.

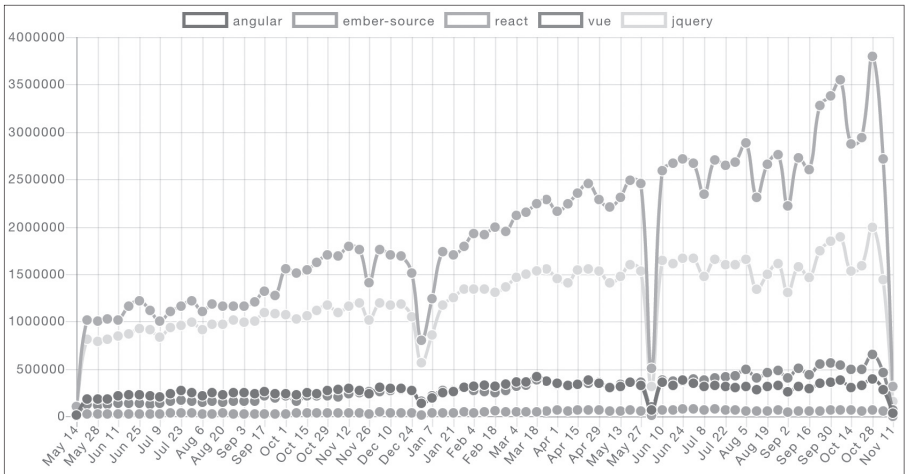
Comparatif des frameworks JavaScript actuels

À ce jour, il existe une multitude de bibliothèques JavaScript telles que JQuery, Angular, Ember et React pour ne citer que les plus populaires. Voici quelques comparatifs que nous pouvons apprécier concernant l'évolution de l'utilisation et de la cote de popularité de Vue.js et ce, malgré sa jeunesse somme toute relative.

La figure I-1 présente les statistiques NPM (graphique) et GitHub (tableau) des frameworks JS sur 2 ans, avec comme métriques la popularité et la tendance d'utilisation avec les curseurs suivants :

- *stars* : popularité des utilisateurs ;
- *forks* : nombre de copies des référentiels faites par les utilisateurs ;
- *issues* : nombre d'anomalies remontées par les utilisateurs.

On observe donc qu'à ce jour, React.js est en première position et Vue.js en deuxième position. Il faut garder en mémoire que React a eu la promotion de Facebook et que ce framework est né 1 an avant Vue. Evan You a su promouvoir son projet, le rendre fiable et populaire.



GitHub Stats

	stars	forks	issues	updated	created
ember.js	20169	4060	336	Nov 13, 2018	May 26, 2011
angular.js	59243	28997	494	Nov 12, 2018	Jan 6, 2010
vue	118793	16895	279	Nov 13, 2018	Jul 29, 2013
jquery	50324	17249	86	Nov 12, 2018	Apr 3, 2009
react	115292	20800	451	Nov 13, 2018	May 24, 2013

Figure I-1 – Comparatif des frameworks JS
(Source : https://www.npmtrends.com)

Cet autre site donne d'abord les statistiques de popularité (onglet ***Popular***) et dans un second temps la tendance d'utilisation (onglet ***Trending***) des frameworks actuels (figures I-2 et I-3).

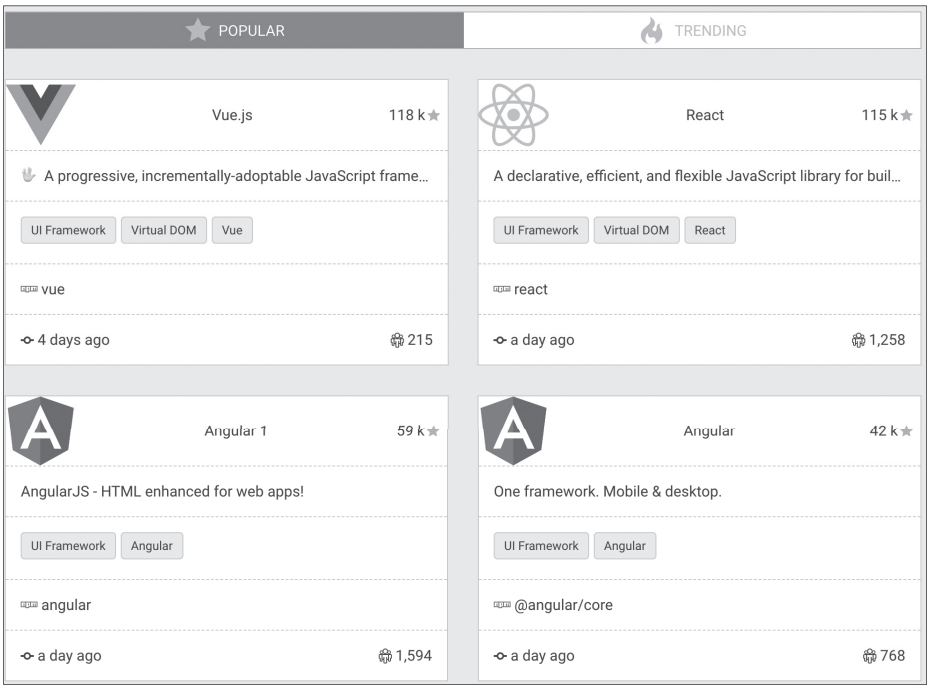


Figure I-2 – Popularité des frameworks JS sur 1 an
(Source : https://bestof.js.org)

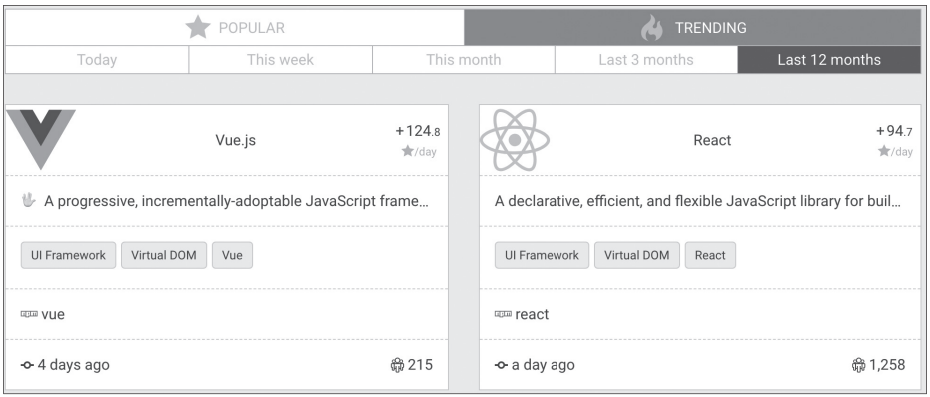


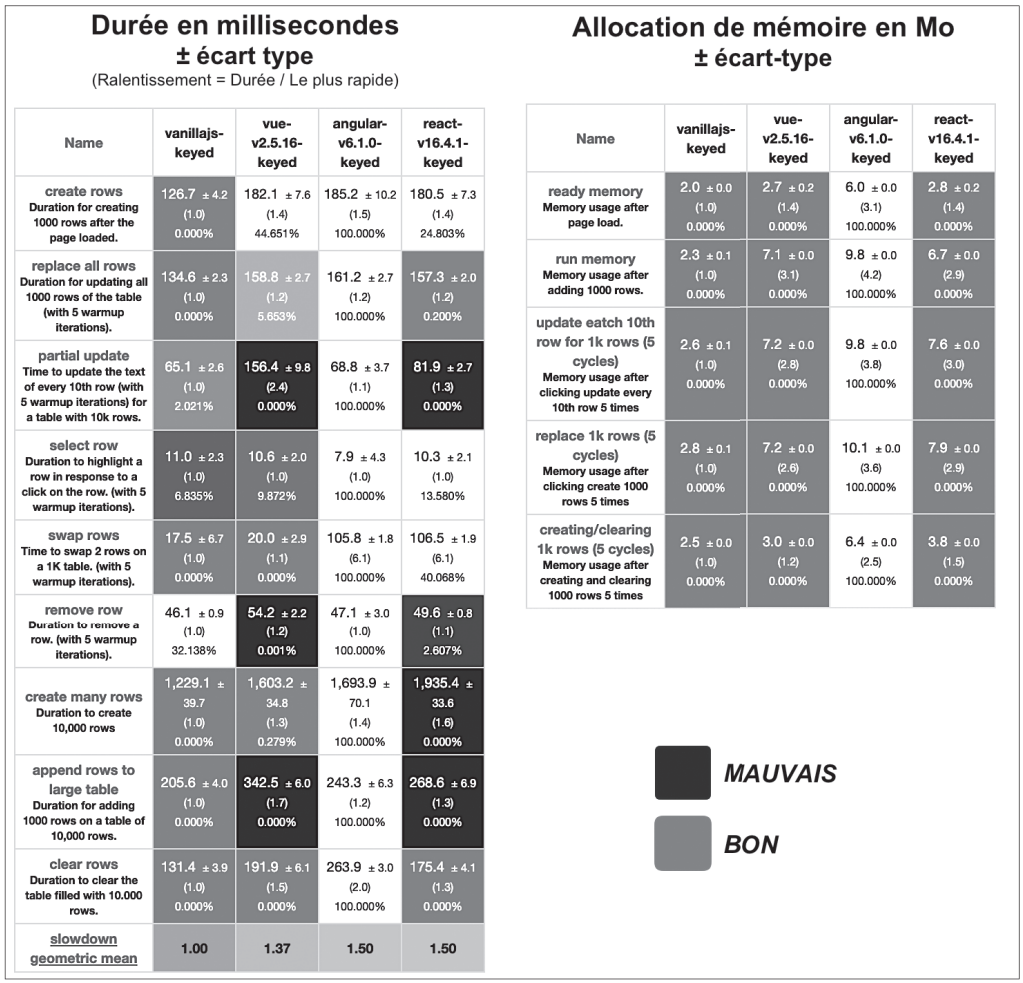
Figure I-3 – Tendance de l'utilisation des frameworks JS sur 1 an
(Source : https://bestof.js.org)

Pour finir, observons un benchmark lancé sur un MacBook Pro 15 (2,5 GHz i7, 16 Go RAM, OS X 10.12.5, Chrome 58.0.3029.110, 64-bit) dans lequel nous avons pour le premier tableau le

temps d'exécution (exprimé en millisecondes) de plusieurs méthodes classiques, puis dans le second tableau l'allocation mémoire utilisée après le chargement de la page et après avoir ajouté 1 000 lignes (figure I-4).

Plus la cellule du tableau tend vers le foncé, pire est le traitement, et inversement lorsqu'elle tend vers le clair.

Nous constatons que le VanillaJS (nom moderne pour parler de JavaScript natif) est forcément en première position et que Vue est en deuxième position !



Durée en millisecondes
± écart type
(Ralentissement = Durée / Le plus rapide)

Name	vanillajs-keyed	vue-v2.5.16-keyed	angular-v6.1.0-keyed	react-v16.4.1-keyed
create rows Duration for creating 1000 rows after the page loaded.	126.7 ± 4.2 (1.0) 0.000%	182.1 ± 7.6 (1.4) 44.651%	185.2 ± 10.2 (1.5) 100.000%	180.5 ± 7.3 (1.4) 24.803%
replace all rows Duration for updating all 1000 rows of the table (with 5 warmup iterations).	134.6 ± 2.3 (1.0) 0.000%	158.8 ± 2.7 (1.2) 5.653%	161.2 ± 2.7 (1.2) 100.000%	157.3 ± 2.0 (1.2) 0.200%
partial update Time to update the text of every 10th row (with 5 warmup iterations) for a table with 10k rows.	65.1 ± 2.6 (1.0) 2.021%	156.4 ± 9.8 (2.4) 0.000%	68.8 ± 3.7 (1.1) 100.000%	81.9 ± 2.7 (1.3) 0.000%
select row Duration to highlight a row in response to a click on the row. (with 5 warmup iterations).	11.0 ± 2.3 (1.0) 6.835%	10.6 ± 2.0 (1.0) 9.872%	7.9 ± 4.3 (1.0) 100.000%	10.3 ± 2.1 (1.0) 13.580%
swap rows Time to swap 2 rows on a 1K table. (with 5 warmup iterations).	17.5 ± 6.7 (1.0) 0.000%	20.0 ± 2.9 (1.1) 0.000%	105.8 ± 1.8 (6.1) 100.000%	106.5 ± 1.9 (6.1) 40.068%
remove row Duration to remove a row. (with 5 warmup iterations).	46.1 ± 0.9 (1.0) 32.138%	54.2 ± 2.2 (1.2) 0.001%	47.1 ± 3.0 (1.0) 100.000%	49.6 ± 0.8 (1.1) 2.607%
create many rows Duration to create 10,000 rows	1,229.1 ± 39.7 (1.0) 0.000%	1,603.2 ± 34.8 (1.3) 0.279%	1,693.9 ± 70.1 (1.4) 100.000%	1,935.4 ± 33.6 (1.6) 0.000%
append rows to large table Duration for adding 1000 rows on a table of 10,000 rows.	205.6 ± 4.0 (1.0) 0.000%	342.5 ± 6.0 (1.7) 0.000%	243.3 ± 6.3 (1.2) 100.000%	268.6 ± 6.9 (1.3) 0.000%
clear rows Duration to clear the table filled with 10.000 rows.	131.4 ± 3.9 (1.0) 0.000%	191.9 ± 6.1 (1.5) 0.000%	263.9 ± 3.0 (2.0) 100.000%	175.4 ± 4.1 (1.3) 0.000%
slowdown geometric mean	1.00	1.37	1.50	1.50

Allocation de mémoire en Mo
± écart-type

Name	vanillajs-keyed	vue-v2.5.16-keyed	angular-v6.1.0-keyed	react-v16.4.1-keyed
ready memory Memory usage after page load.	2.0 ± 0.0 (1.0) 0.000%	2.7 ± 0.2 (1.4) 0.000%	6.0 ± 0.0 (3.1) 100.000%	2.8 ± 0.2 (1.4) 0.000%
run memory Memory usage after adding 1000 rows.	2.3 ± 0.1 (1.0) 0.000%	7.1 ± 0.0 (3.1) 0.000%	9.8 ± 0.0 (4.2) 100.000%	6.7 ± 0.0 (2.9) 0.000%
update eatch 10th row for 1k rows (5 cycles) Memory usage after clicking update every 10th row 5 times	2.6 ± 0.1 (1.0) 0.000%	7.2 ± 0.0 (2.8) 0.000%	9.8 ± 0.0 (3.8) 100.000%	7.6 ± 0.0 (3.0) 0.000%
replace 1k rows (5 cycles) Memory usage after clicking create 1000 rows 5 times	2.8 ± 0.1 (1.0) 0.000%	7.2 ± 0.0 (2.6) 0.000%	10.1 ± 0.0 (3.6) 100.000%	7.9 ± 0.0 (2.9) 0.000%
creating/clearing 1k rows (5 cycles) Memory usage after creating and clearing 1000 rows 5 times	2.5 ± 0.0 (1.0) 0.000%	3.0 ± 0.0 (1.2) 0.000%	6.4 ± 0.0 (2.5) 100.000%	3.8 ± 0.0 (1.5) 0.000%

Figure I-4 – Benchmark de rapidité d'exécution et d'allocation mémoire des frameworks JS (Source : https://www.stefankrause.net)

Rappels de modélisation en génie logiciel

Architecture MVC

En génie logiciel, le modèle-vue-contrôleur *(Model-view-controller)* est une architecture destinée à découper une application en couches (figure I-5), surtout pour les interfaces web.

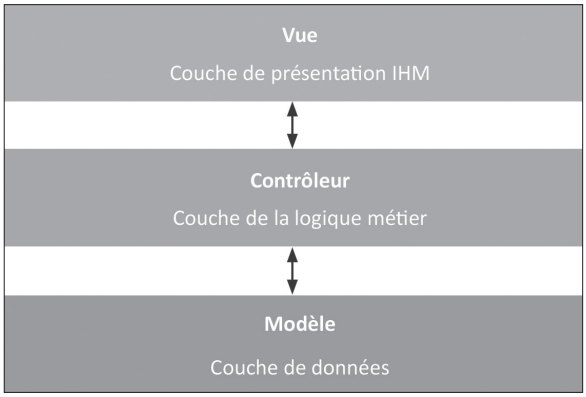


Figure I-5 – Schéma du modèle MVC

Architecture MVVM

En génie logiciel, le modèle-vue-vue modèle *(Model-view-viewmodel)* est une architecture et une méthode de conception qui est originaire de Microsoft, notamment pour WPF et Silverlight. Très similaire au modèle MVC avec une accentuation des principes de binding (liaison) et events (événements). C'est sur cette modélisation que s'appuie le framework Vue.js.

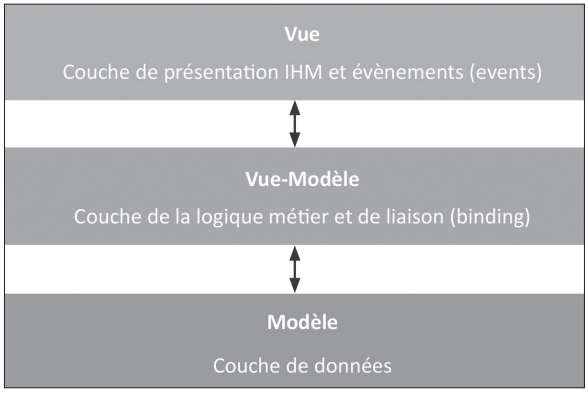


Figure I-6 – Schéma du modèle MVVM

Front-end et back-end

Ici, nous parlons d'organisation, de rôle, de métier…

Le back-end est la partie « immergée de l'iceberg », invisible pour les utilisateurs. Il est le cœur de l'application où l'on retrouve les données et le cœur métier.

Le front-end, quant à lui, est la « pointe de l'iceberg », visible par les utilisateurs. Il représente l'interface : les formulaires et le design de l'application.

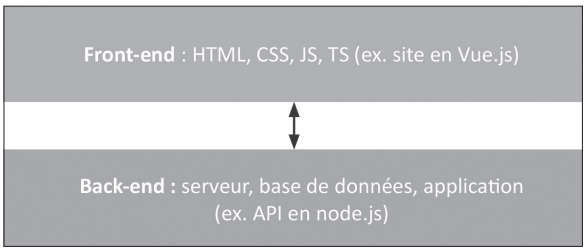


Figure I-7 – Schéma de communication front-end et back-end

1

Installer et utiliser Vue.js

Une version par environnement

Il est possible d'utiliser Vue de différentes manières, mais gardons en tête qu'il existe deux packages très distincts :

- la version de développement : débogage facilité et interaction avec divers outils (voir chapitre 2 sur les outils, page 13) ;
- la version de production : compressée au maximum (minifiée), ce qui aura pour désavantage de ne pas disposer de la richesse des messages d'informations du framework.

Important

Vue ne supporte pas les versions d'Internet Explorer (IE) 8 et inférieures car il utilise des fonctionnalités d'ECMAScript 5.

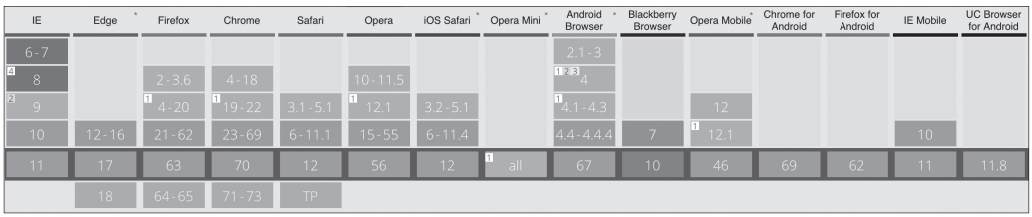


Figure 1-1 – Compatibilité des navigateurs pour ES5
(Source : https://caniuse.com/#feat=es5)

Sources du framework

Autonome – Source officielle

La version Autonome signifie que nous avons en notre possession les sources de Vue. Il suffit de télécharger le fichier désiré et de l'inclure avec un tag script dans son fichier HTML. Ensuite, Vue doit être enregistré comme une variable globale (voir chapitre 3 : Instance de Vue) :

- développement : https://fr.vuejs.org/js/vue.js
- production : https://fr.vuejs.org/js/vue.min.js

CDN – Serveur de distribution

Un CDN *(Content Delivery Network)* stocke sur ses serveurs les sources du framework et nous les propose via un lien directement exploitable dans nos sources. Le site officiel de Vue.js préconise d'utiliser le CDN de `unpkg` qui fournit la dernière version stable possible afin de refléter le package fournit par NPM. Mais il est possible de s'appuyer sur d'autres CDN tels que `jsdelivr` et `cdnjs`.

- https://unpkg.com/vue@2.5.17/dist/vue.min.js
- https://cdn.jsdelivr.net/vue/2.3.2/vue.min.js
- https://cdnjs.cloudflare.com/ajax/libs/vue/2.3.4/vue.min.js

Note

Par défaut, la version proposée est la version minifiée, mais il est possible de supprimer la mention .min dans l'URL afin d'obtenir la version de développement.

Nuxt – Le framework universel

Nuxt est un projet dédié à Vue.js qui embarque en son sein les packages de Vue, Webpack et Babel (*transpiler* qui convertit le code ES en JS). Sont intégrés dans le package vue-router, Vuex et vue-meta. Il suffit de se rendre sur le site https://fr.nuxtjs.org/ pour en savoir plus.

Vue CLI – Le générateur de projet officiel

Vue.js offre une CLI *(Command Line Interface)* officielle qui est une interface en ligne de commande pour générer un projet configuré avec les dépendances préconisées. Dans le terminal, il convient de saisir la commande pour l'installation de la CLI :

Installation de la CLI

```
npm install -g @vue/cli
```

Puis de créer un projet :

Création du projet

```
vue create nom-de-mon-project
```

Il faut ensuite se rendre sur la page https://cli.vuejs.org/ pour lire la documentation officielle détaillée.

CodeSandbox – Une solution clé en main

Le site CodeSandbox (https://codesandbox.io/s/vue) fournit un environnement complet et personnalisable pour développer rapidement et sans aucune installation requise (intégration de VS Code). Il peut être, et c'est d'ailleurs recommandé, couplé avec un compte Git.

CodeSandbox est aussi capable d'injecter des dépendances tierces. C'est un véritable IDE (*Integrated Development Environment* ou Environnement de développement) complet en ligne qui, par ailleurs, propose VS Code en guise d'outil d'écriture de code.

Bower – Un gestionnaire de dépendances

Bower est un outil, un gestionnaire de dépendances, à installer sur sa machine. C'est une solution que nous ne développerons pas ici mais qui mérite d'être mentionnée. Pour ceux qui l'utilisent, saisissez la ligne suivante dans l'invite de commande :

```
bower install vue
```

NPM (ou Yarn) – Le gestionnaire de référence

NPM est l'outil indispensable à avoir sur sa machine. Il est installé avec Node.js et est disponible à l'adresse suivante : https://www.npmjs.com/get-npm.

Utilisé avec le terminal, il permet de gérer les dépendances et s'intègre très facilement avec un empaqueteur de modules *(module bundler)* tel que Webpack, Parcel, Rollup…

Après l'avoir installé, il convient de saisir la ligne suivante dans l'invite de commande :

```
npm install vue
```

Empaqueteurs de modules

Voici les configurations pour les empaqueteurs les plus populaires et un benchmark comparatif.

Webpack

Disponible sur la page https://webpack.js.org/, Webpack est l'empaqueteur le plus populaire. Il demande néanmoins un temps assez important de compréhension et surtout de configuration avant de pouvoir monter un projet. Voici sa configuration pour Vue :

```
module.exports = {
  // ...
  resolve: {
    alias: {
      'vue$': 'vue/dist/vue.esm.js'
    }
  }
}
```

Rollup.js

Rollup est assez proche de Webpack en termes de configuration. Cependant, c'est celui qui met le moins de temps à compiler les sources, que ce soit pour le mode Développement comme pour le mode Production. Il est disponible à l'adresse https://rollupjs.org/guide/en et sa configuration (fichier `package.json`) pour Vue est la suivante :

```
const alias = require('rollup-plugin-alias')

rollup({
  // ...
  plugins: [
    alias({
      'vue': 'vue/dist/vue.esm.js'
    })
  ]
})
```

Parcel.js

Parcel est le plus léger des empaqueteurs, tant en poids qu'en configuration. Il a pour vocation de ne nécessiter aucune configuration, tout est intégré. Il est sûrement à privilégier pour de petits projets et des POC *(proof of concept)*. Pour plus d'information sur Parcel, consulter le site https://parceljs.org/. Pour ajouter la prise en charge des fichiers .vue, il suffit d'ajouter le code suivant dans le fichier `package.json` :

```
{
  // ...
  "alias": {
    "vue": "./node_modules/vue/dist/vue.common.js"
  }
}
```

Benchmark

Pour un même projet, relancé trois fois, voici les temps d'exécution en secondes pour chacun des empaqueteurs :

Tableau 1-1. Benchmark d'exécutions des empaqueteurs

Empaqueteur	1er lancement	2e lancement	3e lancement	Moyenne
Webpack	3,828	3,456	3,902	3,728
Rollup.js	0,650	0,498	0,495	0,547
Parcel.js	15,05	5,674	4,876	8,533

Notons que ce test est soit pour le mode Développement, soit pour le mode Débogage. Pour le mode Production, les temps sont sensiblement les mêmes sauf pour Parcel.js qui est complètement hors-jeu avec un temps de premier lancement multiplié par dix.