

L'AVENIR DU WEB S'ÉCRIT-IL EN JAVASCRIPT ?

Auteur : Jean-Guillaume Dujardin / Pierre Vaidie
Novembre 2013

JavaScript était déjà bien installé sur nos navigateurs. Il est depuis passé côté serveur (avec Node.js). Et maintenant, il se structure côté client. Des frameworks JavaScript émergent...



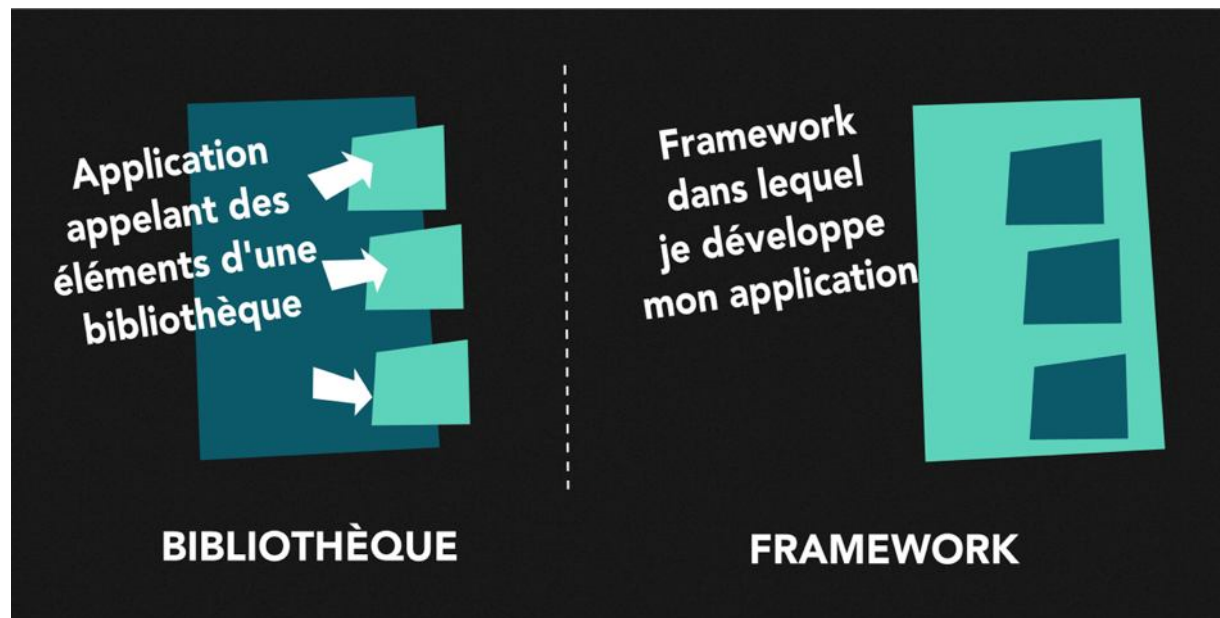
JavaScript

BIBLIOTHEQUE OU BIEN FRAMEWORK, QUELLE DIFFERENCE ?

Vous-vous dites : « Pour le JavaScript, pas besoin de me casser la tête, on a déjà JQuery, c'est bien suffisant ». C'est vrai, JQuery est devenu presque universel. C'est la bibliothèque la plus utilisée. Mais cela reste une bibliothèque.

Les Frameworks sont différents des bibliothèques car ils guident l'architecture logicielle.

Schématiquement, on a le principe suivant :

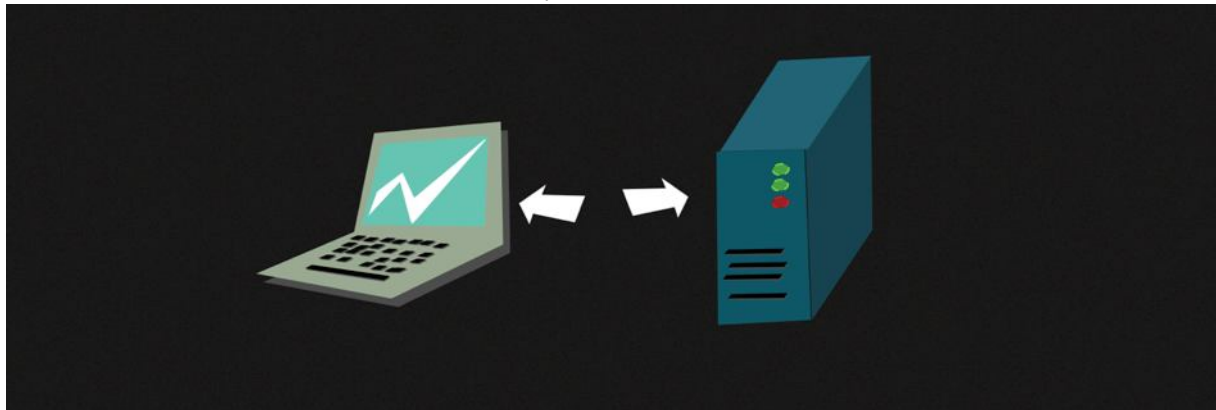


En fait, ces frameworks sont nécessaires à de nouvelles architectures...

QUOI, UN NOUVEAU TYPE D'ARCHITECTURE ?

On dirait que les technologies sont un éternel recommencement... On pensait avoir trouvé une architecture universelle : un navigateur, un serveur PHP, des bases de données MySQL. Patatra ! Parfois, ce type d'architecture ne suffit plus : l'ergonomie des sites devient de plus en plus complexe, la vitesse d'exécution de l'affichage et des traitements est de plus en plus importante (il n'y a qu'à voir les études faites par Google ou Amazon sur la corrélation entre le taux de transformation et les performances), les clients qui se connectent sont de plus en plus différents (mobiles, tablettes), le big data ou bien encore la montée en charge des applications nécessitent de gérer d'énormes quantités de données.

Ainsi, le client doit à nouveau prendre en charge une partie croissante des traitements, comme au bon vieux temps des architectures « client lourd ».



ARCHITECTURE CLASSIQUE

CLIENT

Requêtes lourdes.
Affichage des pages HTML complètes.

SERVEUR

Requêtes traitées au fur et à mesure.
Traitement des données.
Template des pages HTML.

ARCHITECTURE JAVASCRIPT

CLIENT

Requêtes légères.
Récupération et traitement des données.
Rafraîchissement partiel continu.
Template des pages HTML.

SERVEUR

Requêtes légères traitées en simultané (Nodejs).
Gestion des données.

OK, ALORS LEQUEL FAUT-IL CHOISIR ?

Si le framework Node.js est incontournable pour la partie serveur, il existe une multitude de frameworks qui a été développée pour la partie client. Il y a fort à parier que l'un d'entre eux s'imposera. Aussi le choix du bon framework s'avérera certainement déterminant pour le futur.

En dehors d'une éventuelle surprise, trois frameworks sortent du lot : Ember, Backbone, Angular. On pourrait chercher des raisons objectives de préférer une solution plutôt qu'une autre en étudiant les différences techniques. Malheureusement, le succès d'une solution ne repose pas uniquement sur ses qualités techniques. Ce succès repose plutôt sur un consensus parmi la communauté des développeurs.

Aujourd'hui, les tendances que nous avons détectées (je ne juge pas de la rationalité de ces avis parce que je risquerais d'avoir trop de détracteurs) sont les suivantes :



Ember.js est perçu comme très complexe et trop lourd. Par ailleurs, la solution a eu du mal à se stabiliser.

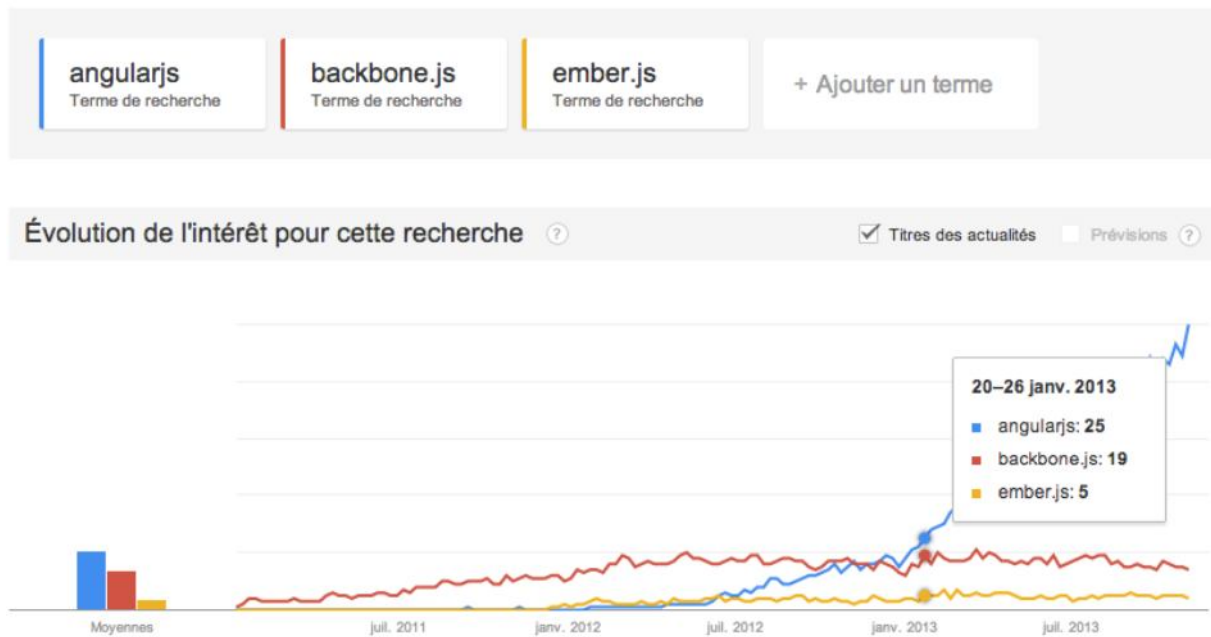


Backbone.js a plusieurs avantages : il est simple, léger et bien documenté. Il a les inconvénients de ses avantages : il est parfois trop limité et pas assez structurant.



Vous l'avez certainement déjà compris, AngularJS est le framework qui récolte le plus de suffrage dans la communauté des développeurs. En plus, il est sponsorisé/développé par Google.

La courbe Google Trends illustre d'ailleurs bien cette tendance :

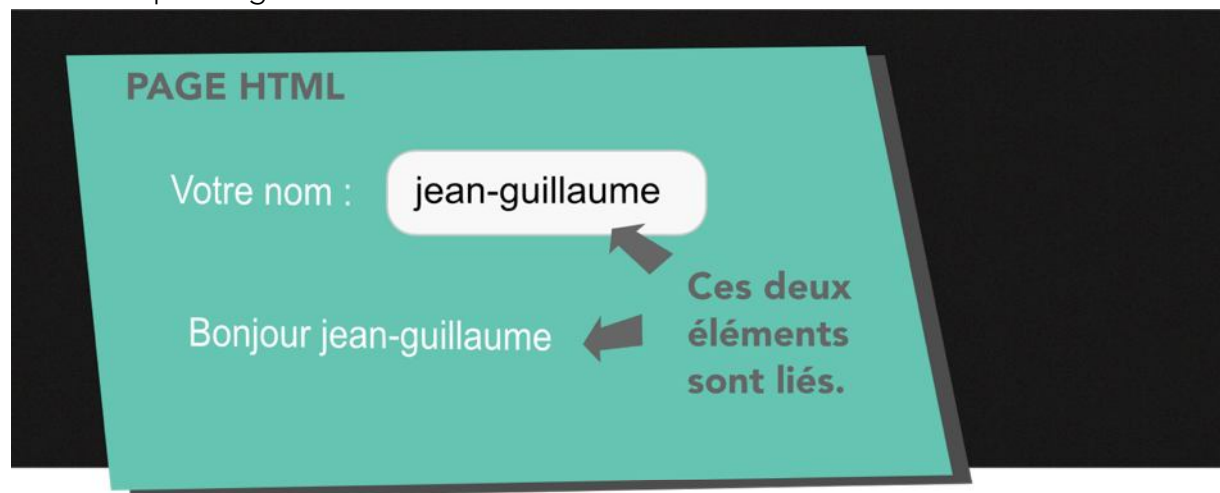


QU'EST-CE QUI EST BIEN AVEC ANGULAR ?

Angular, au-delà d'apporter un design pattern MVC côté client (on parle même de MVW, le W pour « whatever works for you »), présente de nombreuses fonctionnalités très intéressantes :

DATA-BIDING BIDIRECTIONNEL

Un input de la page peut s'afficher dans un autre endroit de la page. Cela n'a pas l'air dingue dit comme ça. En fait, cela simplifie considérablement la vie du développeur (moins de code, donc moins de bug). La manipulation du DOM est effectuée par Angular.



LES DIRECTIVES

Les directives permettent de développer des attributs et des éléments HTML spécifique de manière à manipuler proprement le DOM (structure HTML). Bien que ce soit une fonctionnalité un peu complexe à maîtriser surtout les scopes (il faut savoir dès le départ, dans votre projet, qu'elles sont les directives et les contrôleurs) cela rend votre application très modulaire. Il devient très simple d'ajouter ou de modifier des composants. Ces directives permettent de définir des attributs (une liste particulière avec une image par exemple) ou de piloter des comportements (un événement par exemple).

Ce qui nous fait penser qu'Angular va exploser : les directives peuvent se partager. Vous pouvez donc utiliser une directive en ajoutant le fichier à votre projet. Des bibliothèques comme Angular UI angularisent des librairies populaires comme JQuery UI et Bootstrap. Simplissime, non ?

Bien d'autres éléments sont à découvrir : la gestion des templates ou encore l'injection de dépendance et les services...

CONCLUSION

Ce type d'architecture n'est pas forcément adapté à tous les projets. Il faut encore que les besoins associés au projet soient en adéquation avec les avantages de ces solutions : vitesse d'exécution, composants HTML hautement réutilisables...

Le principal frein à l'utilisation de ces solutions est le référencement. Cette technologie ne permet pas aux moteurs de recherche d'indexer les sites reposant sur cette technologie. Ce sujet devrait bouger. Google fait (paraît-il) des gros efforts dans ce domaine et cela semble plutôt logique puisqu'il porte le projet.

L'avenir des architectures web va dans le sens des Web Components. C'est-à-dire des composants HTML modulaires et réutilisables. Angular, avec les Directives, est le premier framework diffusé largement qui va dans ce sens. Donc, à moins qu'un nouveau langage n'apparaisse dans le paysage comme Dart par exemple, le futur s'écrit en JavaScript. Cependant, à l'heure même où j'écris ces lignes, le 5 novembre 2013, Angular annonce la sortie d'AngularDart. Une drôle de coïncidence ?



Un projet, une idée,
TheCodingMachine peut
vous aider !

www.thecodingmachine.com

Tél : 01 71 18 39 73

contact@thecodingmachine.com

4, rue de la Michodière – 75002 PARIS