Table des matières

[Objectif Pédagogique 3](#_Toc63177042)

[Présentation HTML 4](#_Toc63177043)

# Objectif Pédagogique

# Présentation Angular

Angular est un framework JS permettant de faciliter le développement de (SPAs) réactive.

* 1. De manière schématique cela signifie que notre application sera construite autour d’une seule page HTML. On donne l’illusion de naviguer à travers plusieurs pages HTML mais au final notre page ne change JAMAIS.
  2. Au final nous utilisons 1 fichier HTML + du code JS que l’on appelle depuis un serveur
  3. Chaque changement est rendu dans le navigateur. Cela garanti à l’utilisateur une meilleure UX très réactive.
  4. JS est plus rapide que d’avoir à appeler un serveur pour chaque changement dans nos pages ou données que l’on veut afficher
  5. Cette approche permet de créer des applications Web qui ressemble et réagissent comme des applications mobiles

Si on a besoin de données depuis un serveur on va les charger en arrière-plan.

Dans notre application chaque action va modifier notre Single Page

Comment ?

JS va modifier le DOM à chaque changement en modifiant le code HTML de notre SPA dynamiquement. On ne recharge pas la page on ne fait que travailler sur la page chargée en cours.  
Si on inspecte le code source d’une application angular on peut constater qu’il n’y a que très peu d’HTML.

Le nombre de versions d’Angular peut paraître confus   
Angular 1 (AngularJS) : première version d’Angular (Petite révolution en son temps) 2012 

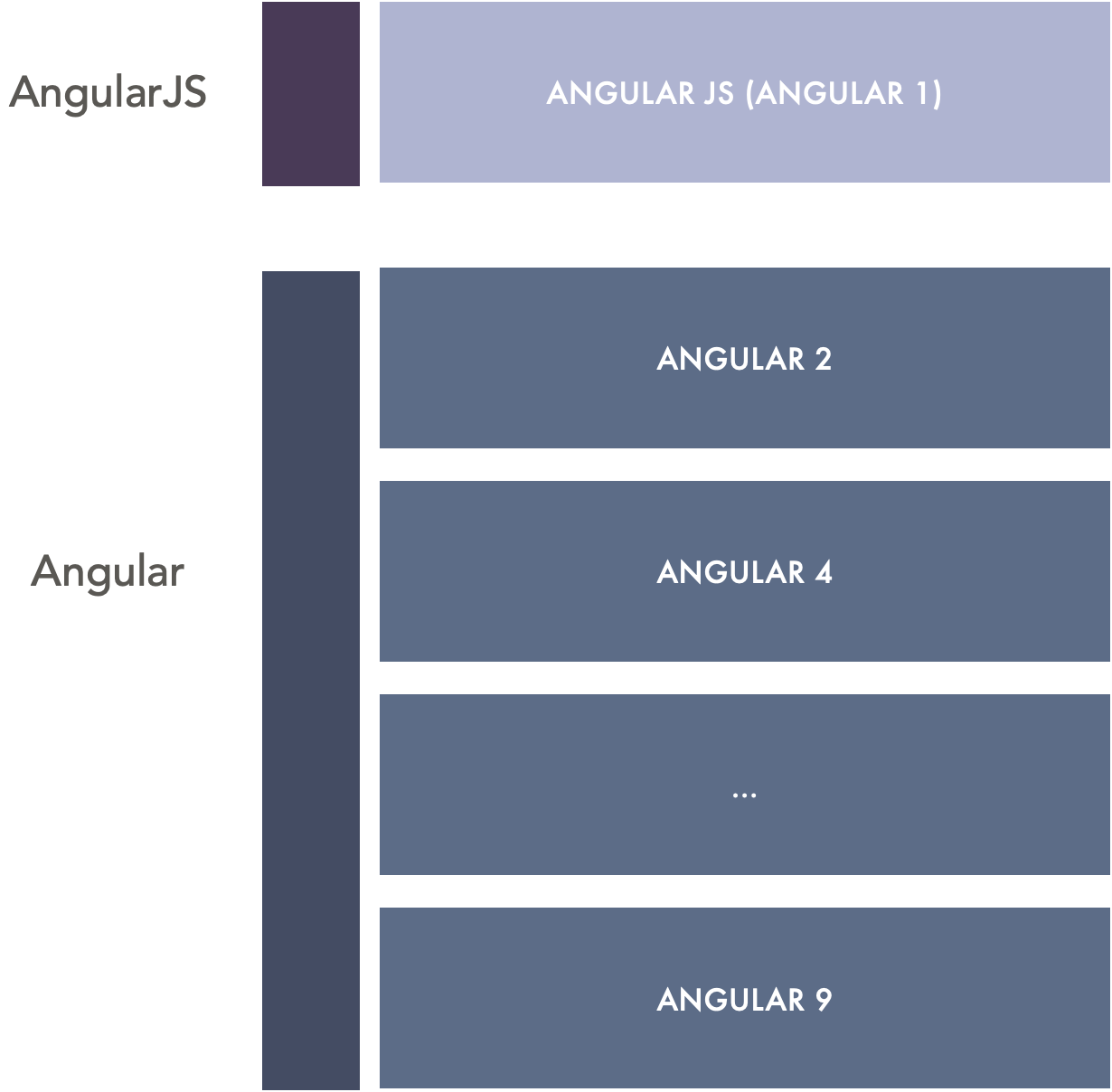
Néanmoins cette première version « beta » comportait tout de même certains problèmes de performances (essentiellement lié aux limites du langage Javascript).

Angular 2 propose une ré-écriture complète du framework, avec le langage TypeScript (un superSet de Javascript orienté sur le typage des données) afin de corriger les problèmes de la première version.

Depuis angular 2 les équipes de développement du framework ne font qu’ajouter des fonctionnalités

1 nouvelle version tous les 6 mois, les MAJ ne font que modifier des éléments liés au fonctionnement interne d’Angular ,les nouvelles fonctionnalités ne viennent pas « remplacer » les anciennes.

La philosophie du framework et sa syntaxe ne changent pas. Si on apprend Angular 2, on ne sera pas perdus sur les prochaines versions. Il s’agit en fait de petites améliorations incrémentales et rétro-compatible.



MPA (Multiple Page Application)

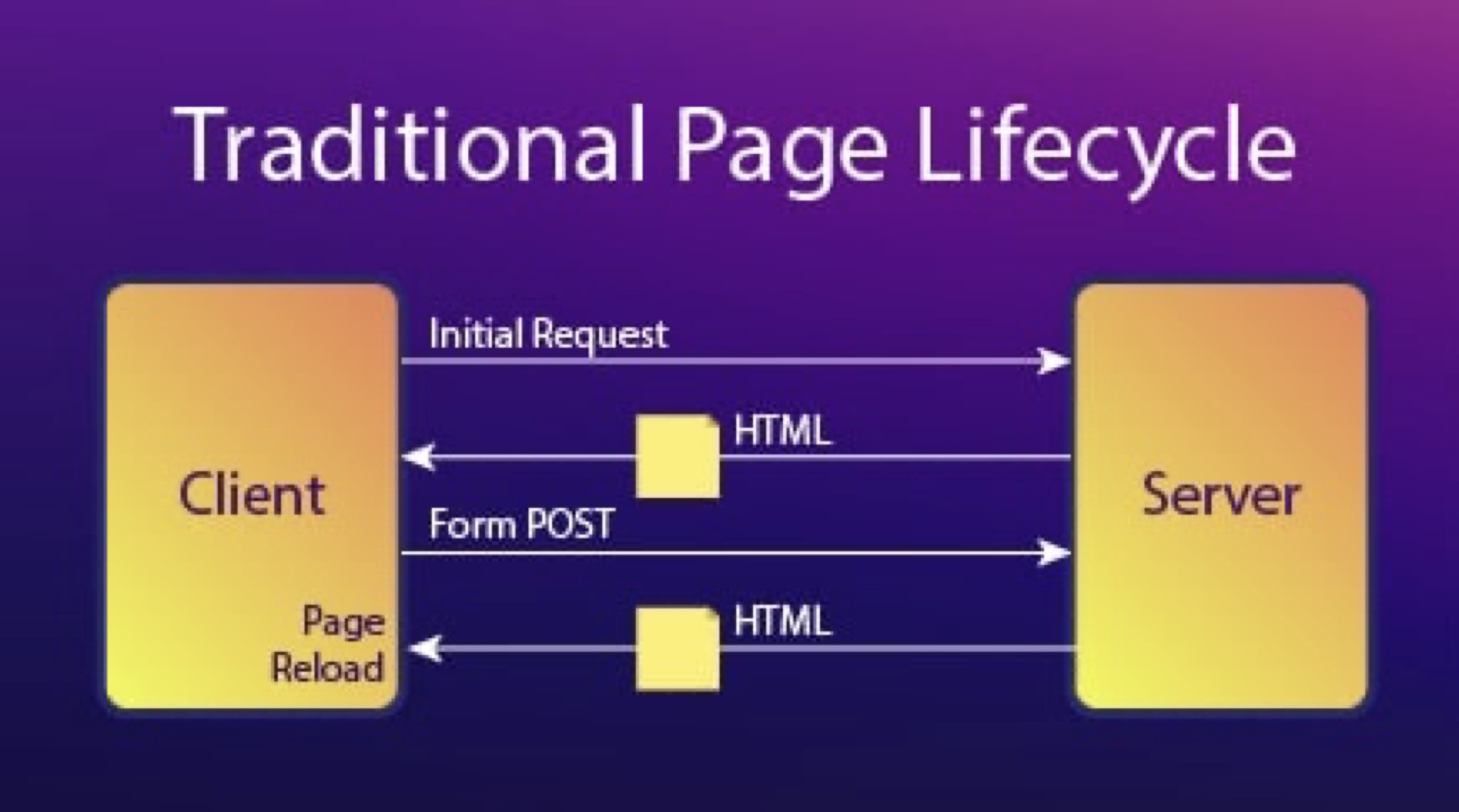
Schéma d’application web traditionnelles,

1 interaction sur 1 page = rechargement de toute la page pour en afficher une nouvelle.

A chaque échange de données, on fait une requête serveur pour afficher une nouvelle page dans notre navigateur.

Ce processus prend du temps pour générer les pages côté serveur, les envoyer au client pour ensuite l’afficher dans le navigateur.

Cependant, AJAX avait rendu possible le rendu d'un composant particulier de l'application, mais cela rend le processus de développement encore plus difficile et complexe.



Cycle de vie d’une requête client – serveur (MPA)

1 requête initiale de client au serveur (on tape une URL)

Le serveur renvoi une page Html à afficher

La page se recharge

Le client effectue une requête (envoyer un formulaire) au serveur

Le serveur renvoi une page Html à afficher

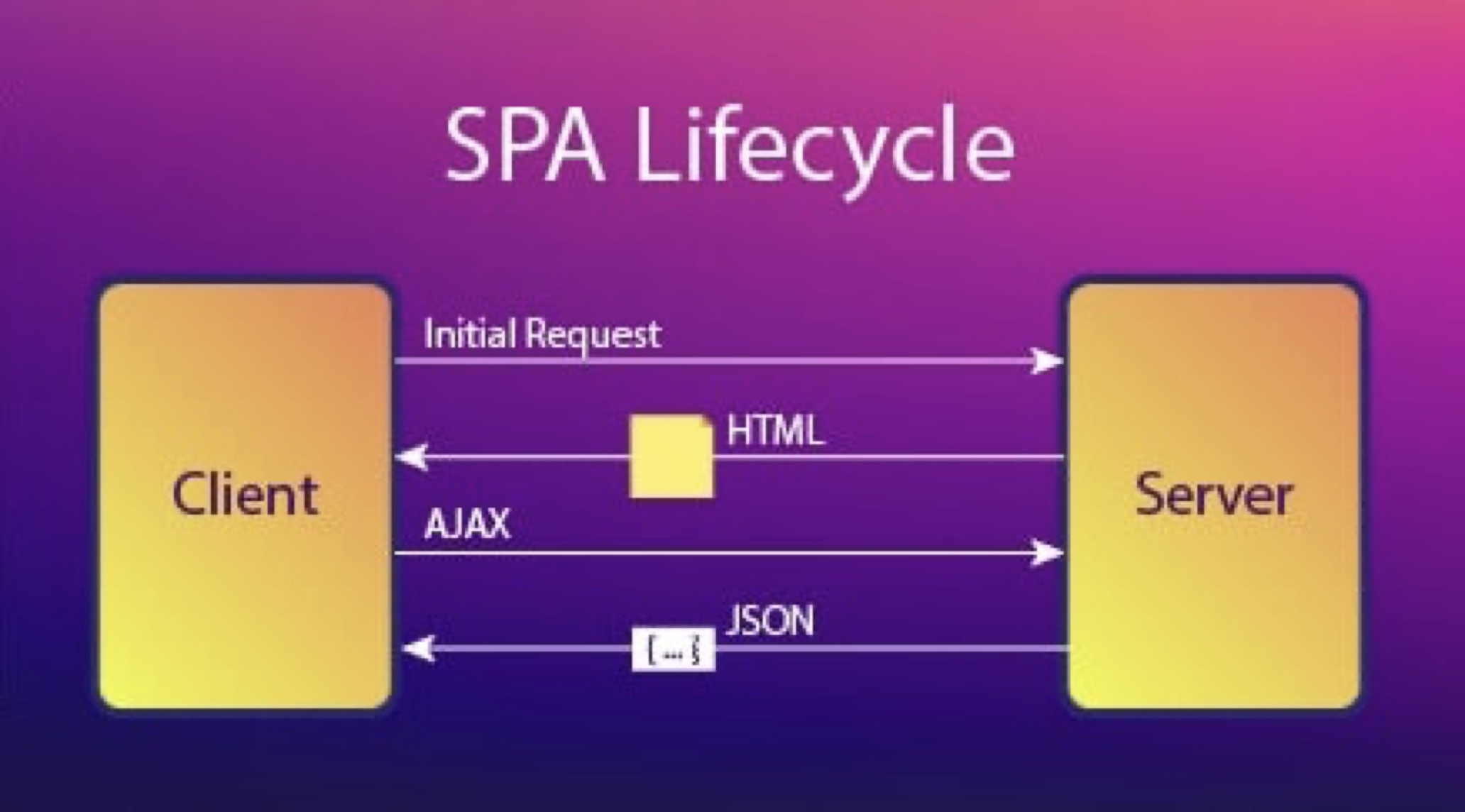
La page se recharge

MPA a une architecture classique, ce qui signifie que chaque page d'un site Web doit être complètement rechargée à la demande du serveur lors de la mise à jour des données.

SPA (Single Page Application)

Schéma d’application dans lequel on ne mettre à jour qu'une partie d'un contenu, ce qui permet d'éviter le rechargement complet de la page.

SPA peut être défini comme une application dynamique d'une seule page, une alternative rapide et conviviale aux applications multipages.



Cycle de vie d’une requête client – serveur (SPA)

Le SPA est créé de manière à ne mettre à jour qu'une partie d'un contenu, ce qui permet d'éviter le rechargement complet de la page.

Ainsi, SPA peut être défini comme une application dynamique d'une seule page et une alternative rapide et conviviale aux applications multipages.

SPA les +

+ rapide : due à un minimum de données chargées

Meilleure navigation : + smooth, pas de lag, pas de chargement

Accès Offline : SPA met les données en cache de manière transparente, l’appli (ou une partie) fonctionne hors ligne, on se dédouane des problèmes de faible bande passante.

(Rappel : 90% des internautes s’en vont de votre site si il met + de 3secondes à charger)

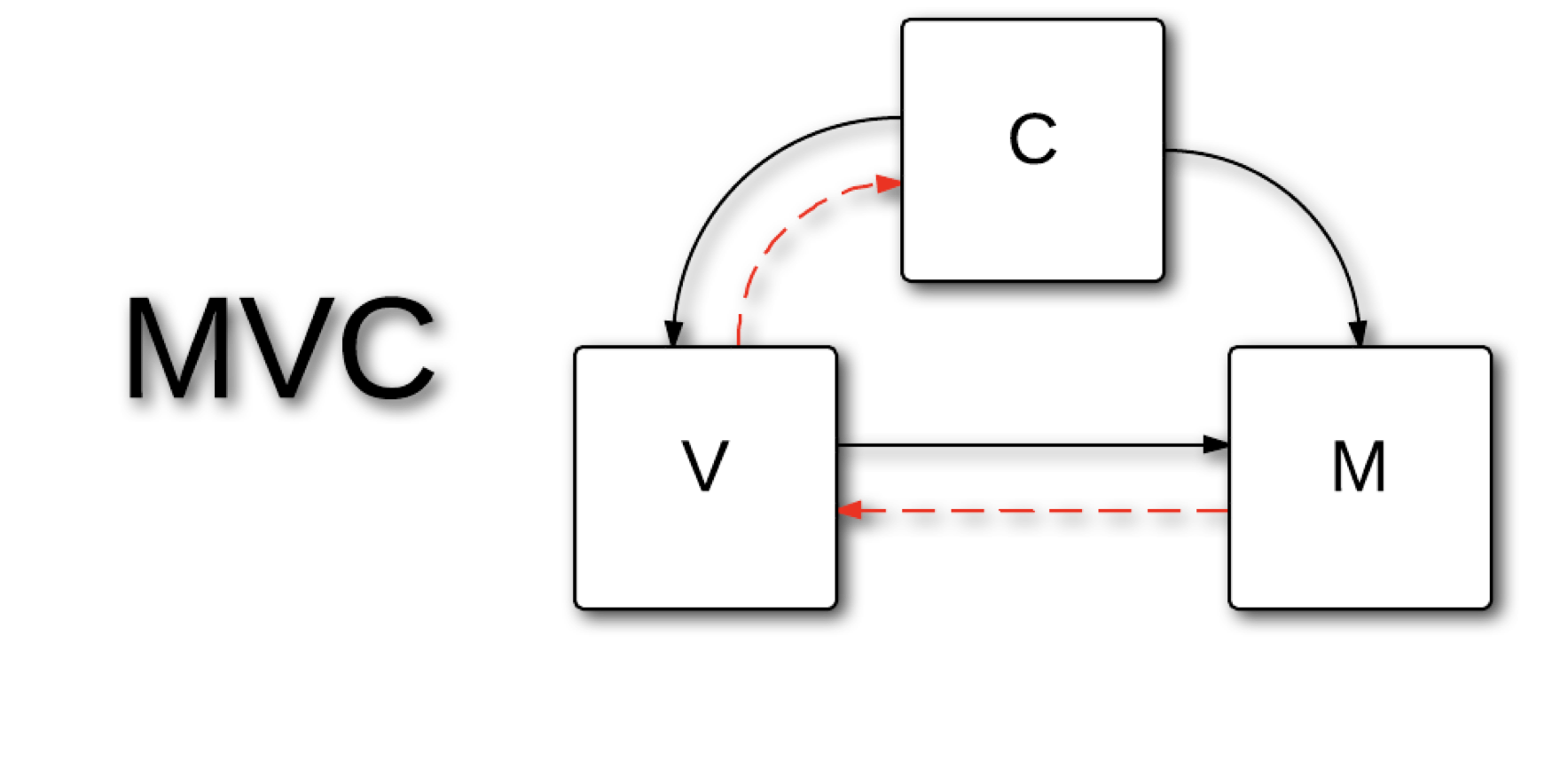
SPA les +

* 1) Les applications SPA répondent comme des applications de bureau : cela est réalisé en déplaçant les données et le traitement du serveur vers le client, et en déplaçant la logique métier du serveur vers le client : de cette façon, seules la validation des données et le stockage permanent sont situés côté serveur.
* 2) La bande passante du SPA est minimale : la communication avec le serveur se fait en petite quantité via Ajax, en utilisant la sintax économique JSON.
* 3) SPA se présente comme des applications de bureau : puisqu'il n'y a pas d'aller-retour pour récupérer du HTML.
* 4) Les SPA sont des applications accessibles : les utilisateurs peuvent utiliser des SPA sur tablettes, ordinateurs portables, smartphones, etc.
* 5) Les SPA continuent de fonctionner hors ligne : un SPA peut se déconnecter si la connexion au serveur tombe et synchroniser plus tard les données du navigateur avec le serveur.
* 6) Les SPA n'ont pas besoin de télécharger les mises à jour : tout ce que les utilisateurs doivent faire pour mettre à jour leurs applications, c'est de recharger le navigateur pour charger automatiquement la dernière version.
* 7) La navigation du SPA entre des pages "différentes" est immédiate : aucun rechargement de page n'est nécessaire, car les "nouvelles" pages sont générées par le SPA à la volée, sans charger HTML à partir du serveur
* 8) Les SPA sont multi-plateformes: bien sûr, un SPA fonctionne sur n'importe quel système d'exploitation.
* SPA les -
* Problèmes de référencement : au début des SPA les robots d'exploration Google ont souvent rencontré des problèmes lors de l'indexation des SPA. Cependant, ce problème a été résolu avec succès et il n'y a plus de tracas avec l'exploration.
* Framework Lourd : L’utilisation de framework lourd peut nécessiter que certaines parties doivent être préchargées côté client, pour éviter des problèmes de chargement.
* Fuite de mémoire en JS

Par rapport à une application multipage traditionnelle, le SPA est considéré comme moins sécurisé. En fait, les fuites de mémoire en JavaScript peuvent ralentir même un système puissant.  
Bien que la majorité se transforme lentement vers les SPA, certains projets ne peuvent toujours pas suivre cette tendance. Et pour eux, les MPA restent plus applicables.

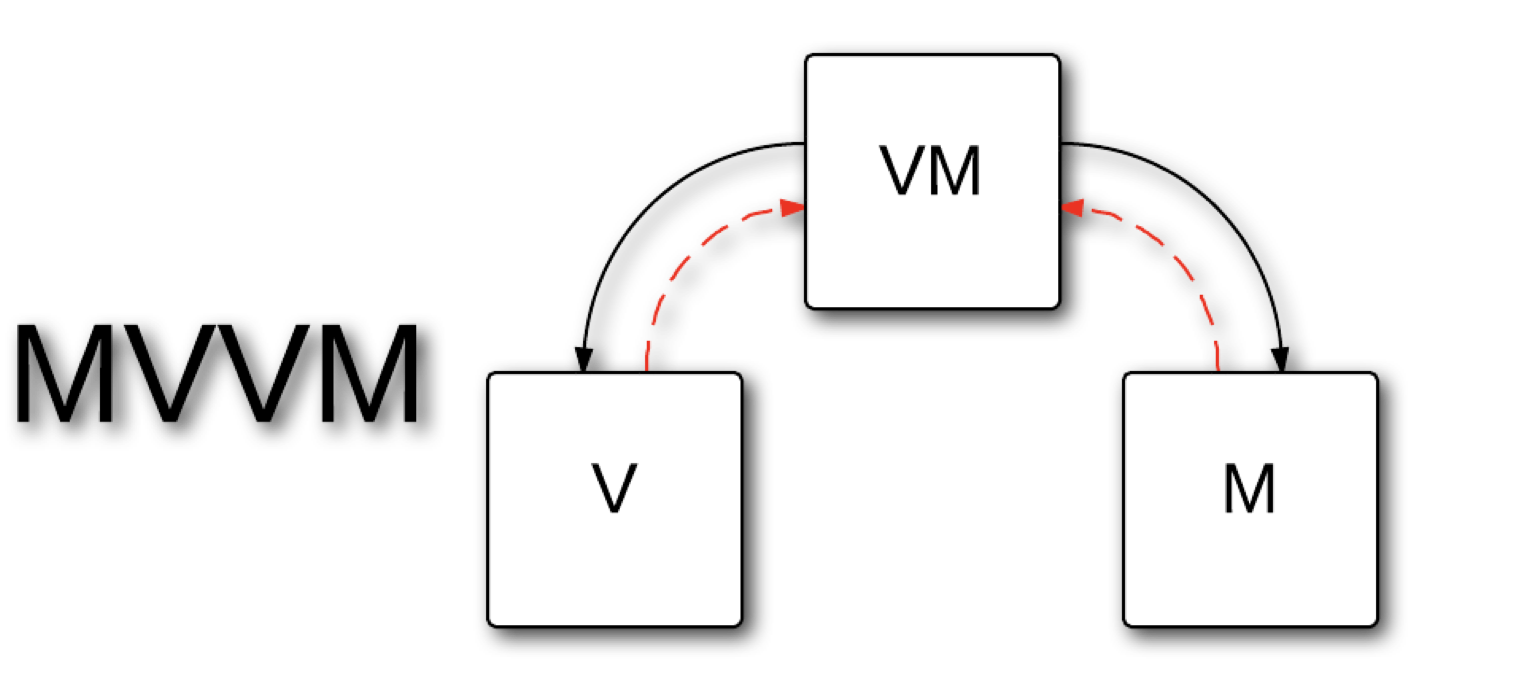
MVC

MVC = modèle, contrôleur, vue = communication essentiellement à sens unique



MVVM

MVVM = modèle, contrôleur, cache, vue = communication bidirectionnelle



Pour construire des applications, Angular utilise TypeScript, TypeScript est un SuperSet de Javascript

* Au final TypeScript est compilé en JS pour être interprété par les navigateurs
* Pourquoi ne pas utiliser JS tout simplement ?
  1. TS ajoute des fonctionnalités
  2. Code plus accessible, moins de possibilités d’erreurs
  3. Principale fonctionnalité : Le typage des données
  4. La création de classes avec le keyword class
  5. Utilisation des interfaces pour définir le contrat d’une classe ou d’une fonction

Les modules.