(Leçon 03)
Introduction aux applications Web Serveur

LOTFI DERBALI, PH.D
DERBALI.IT@GMAIL.COM
HTTPS://GITHUB.COM/DERBALI-IT

Contenu

- Fonctionnement d'un site web : sites statiques, sites dynamiques
- Les requêtes : GET et POST
- Autres technologies utilisées en programmation Web serveur
- Introduction à l'architecture MVC
  - Les éléments d'une application ASP.NET MVC
  - Présentation et description du modèle, de la vue et du contrôleurs

Qu'est-ce qu'une application web (Web App)?

- « Une application Web (aussi appelée Web App) est un logiciel applicatif manipulable grâce à un navigateur Web » (Thierry Pires, décembre 2011)

- Avantages
  - L'accès est universel
  - Les coût de support sont réduits
  - L'utilisateur n'a pas des installations à faire
  - L'utilisation est multi-support
  - Vous faites les mises à jour pour vos clients
  - Vous pouvez travailler en collaboratif
  - Et puis… vous faites des économies massives

Différences entre applications Web et sites Web

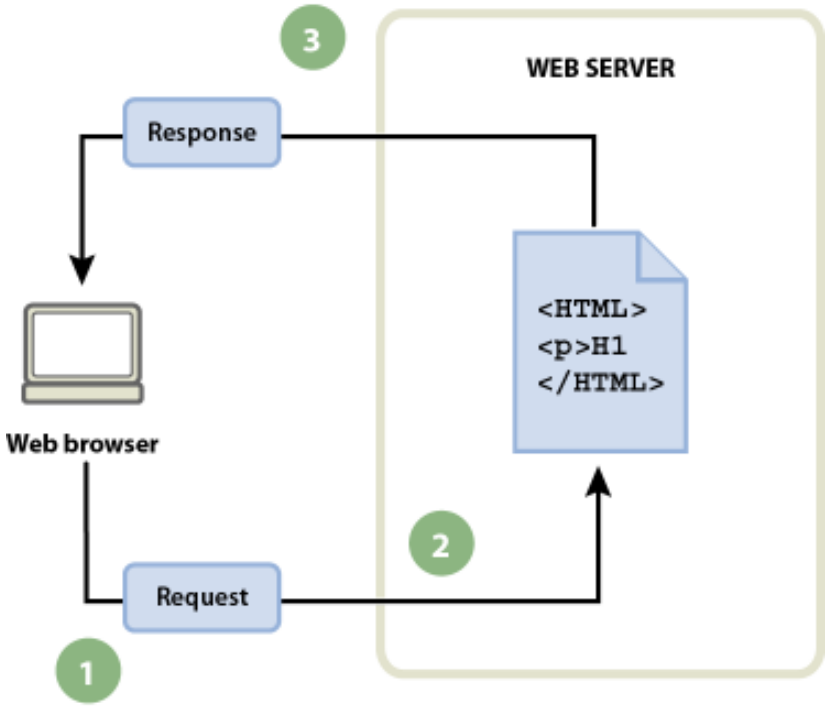
Sites Web

- sont basés sur HTML, JavaScript ou CSS.
- Les sites Web au contraire ont pour la plupart un caractère informatif.

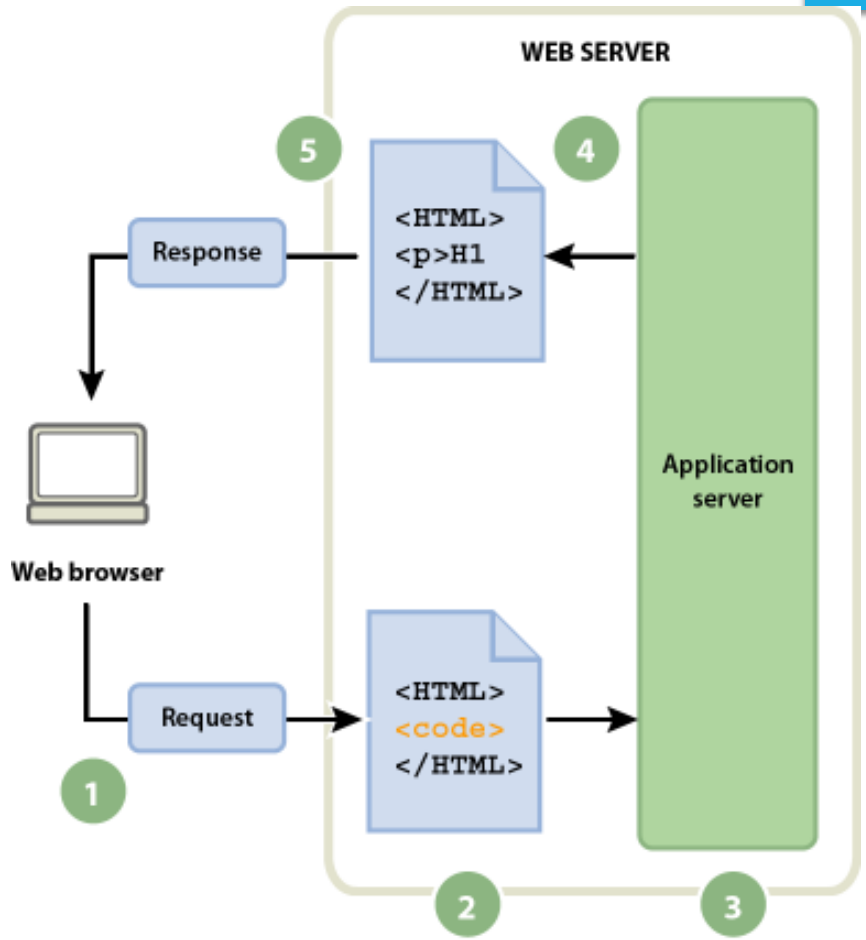
Applications Web

- Les applications Web contiennent toujours des éléments interactifs.
- Les application Web vous permet de travailler avec des ressources côté serveur telles que les bases de données.
- Certains services Google comme Google Maps, Gmail ou le moteur de recherche sont des applications Web, certaines offres Internet comme Amazon et eBay en font partie.

Différences entre applications Web et sites Web

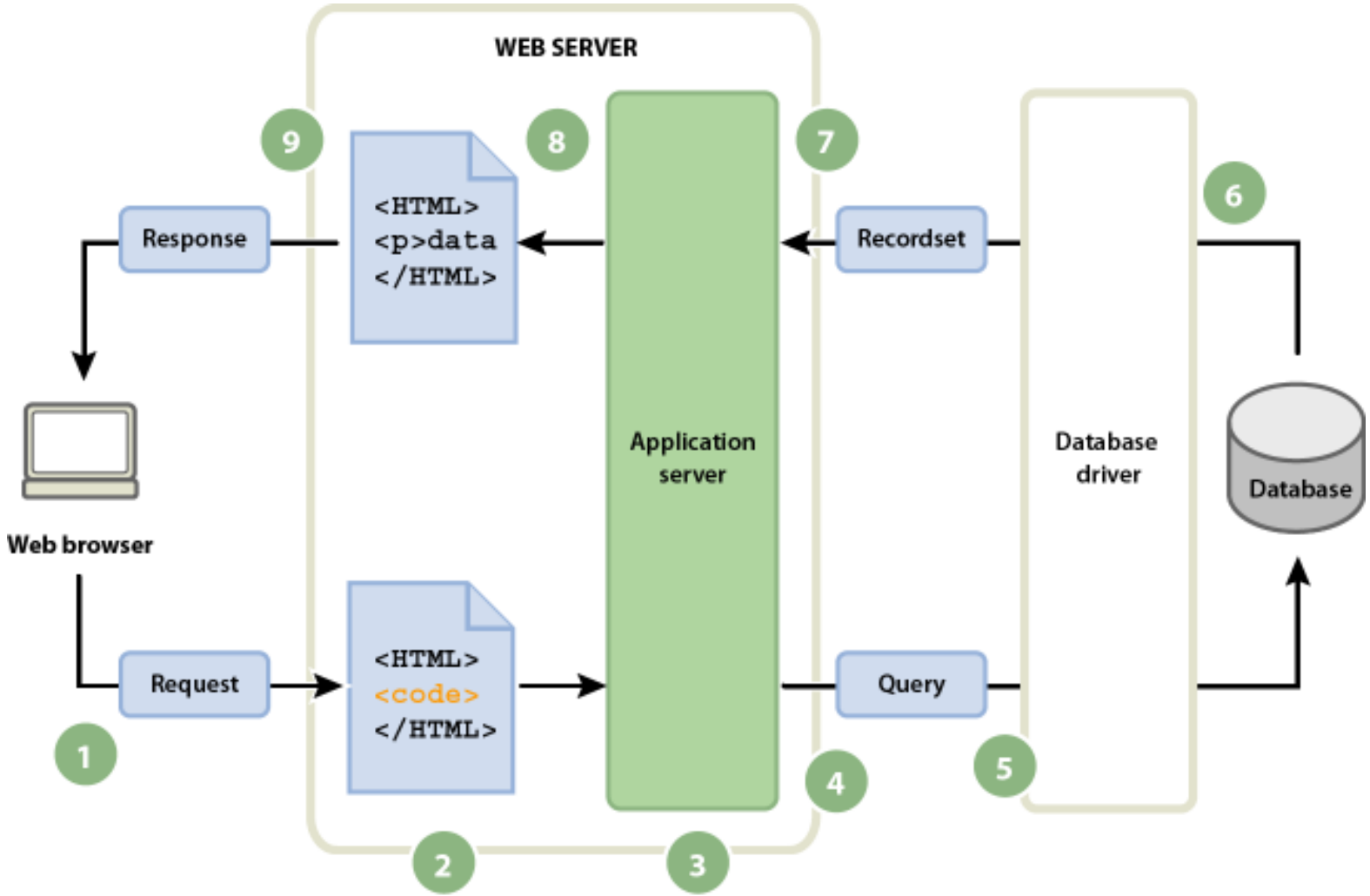


Traitement des pages web statiques



Traitement des pages dynamiques

Accès à une base de données



Accès à une base de données

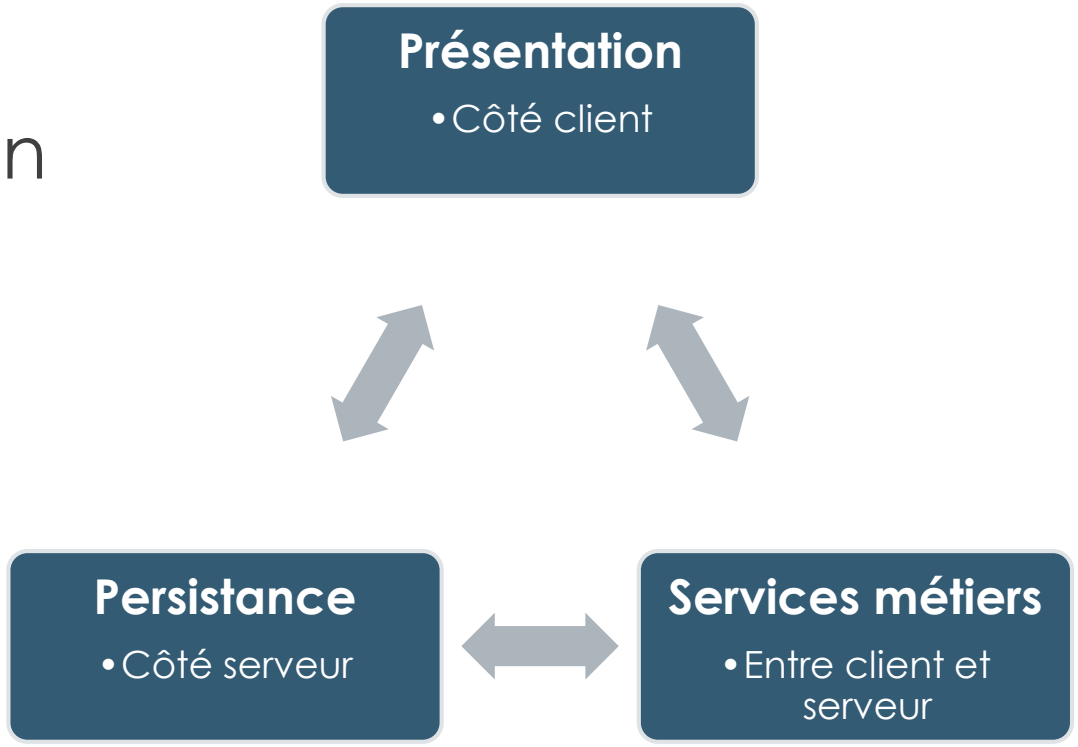
Rôles des applications Web serveur

- Le Web, ce n'est pas qu'un ensemble de documents HTML statiques !

- Les programmes côté serveur permettent :
  - de traiter des soumissions de formulaire ;
  - d'afficher de manière uniforme l'ensemble des pages d'un site ;
  - de proposer des applications interactives ;
  - de permettre à l'utilisateur d'ajouter ou modifier du contenu ;
  - etc.

Application Web serveur

- La méthode des couches est très employée en entreprise pour permettre une meilleure organisation du code
  - Il est ainsi plus facile de faire évoluer le code, de le maintenir et de le corriger
  - Exécution sur des machines différentes



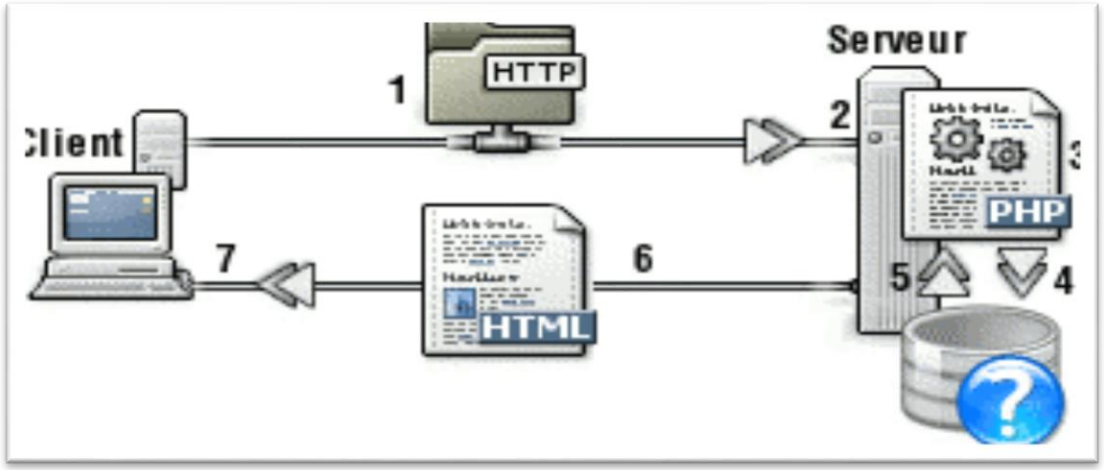
- Client:
  - présentation, interface utilisateur (formulaires, liens vers des URLs, …)
- Serveur de données:
  - Persistance, gestion physique de données, requêtes
  - Pour réaliser cela
    - soit c'est natif dans le langage utilisé (ex: PHP)
    - soit on passe par des Framework ou des API dédiés
- Serveur applicatif
  - Serveur HTTP exécutant des composants logiciels qui génèrent dynamiquement du contenu HTML via des requêtes à des bases de données

Microsoft (ce cours ☺)

- La technologie ASP.NET est proposée par Microsoft, permettant de développer des applications Web
  - Programmation Web du côté serveur.
  - Création de documents dynamiques grâce à la technologie des « Active Server Pages » (ASP.NET).
  - Accès aux bases de données.

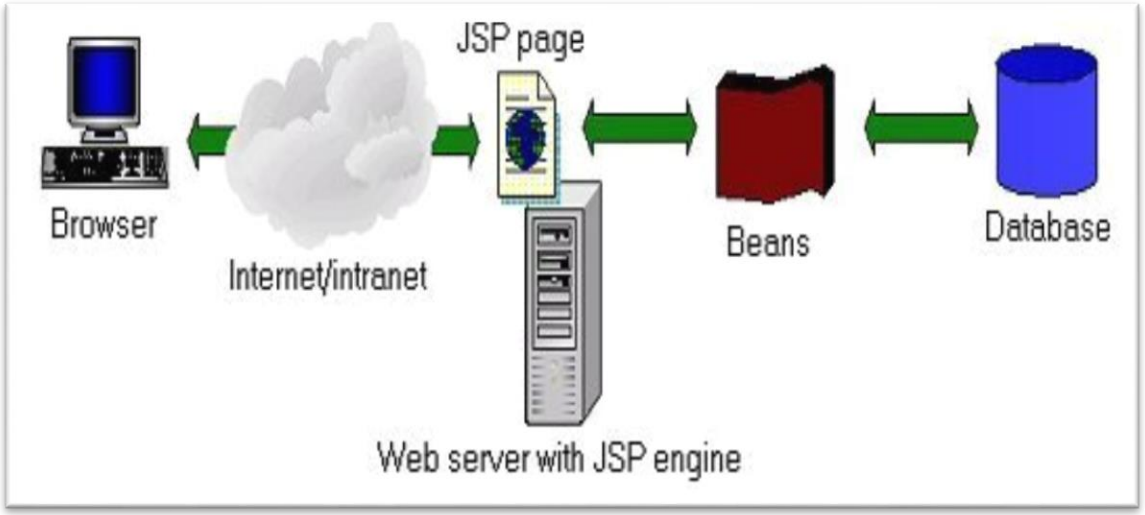
Autres possibilités

Fonctionnement du module PHP




```php
<html>
 <head>
  <title>Test PHP</title>
 </head>
 <body>
 <?php echo '<p>Bonjour le monde</p>'; ?>
 </body>
</html>
```

Fonctionnement du module JSP (Java)




```jsp
html>
<body>
<% String visitor = request.getParameter("name");
if (visitor == null) visitor = " World"; %>
Hello, <%= visitor %>!
</body>
</html>
```

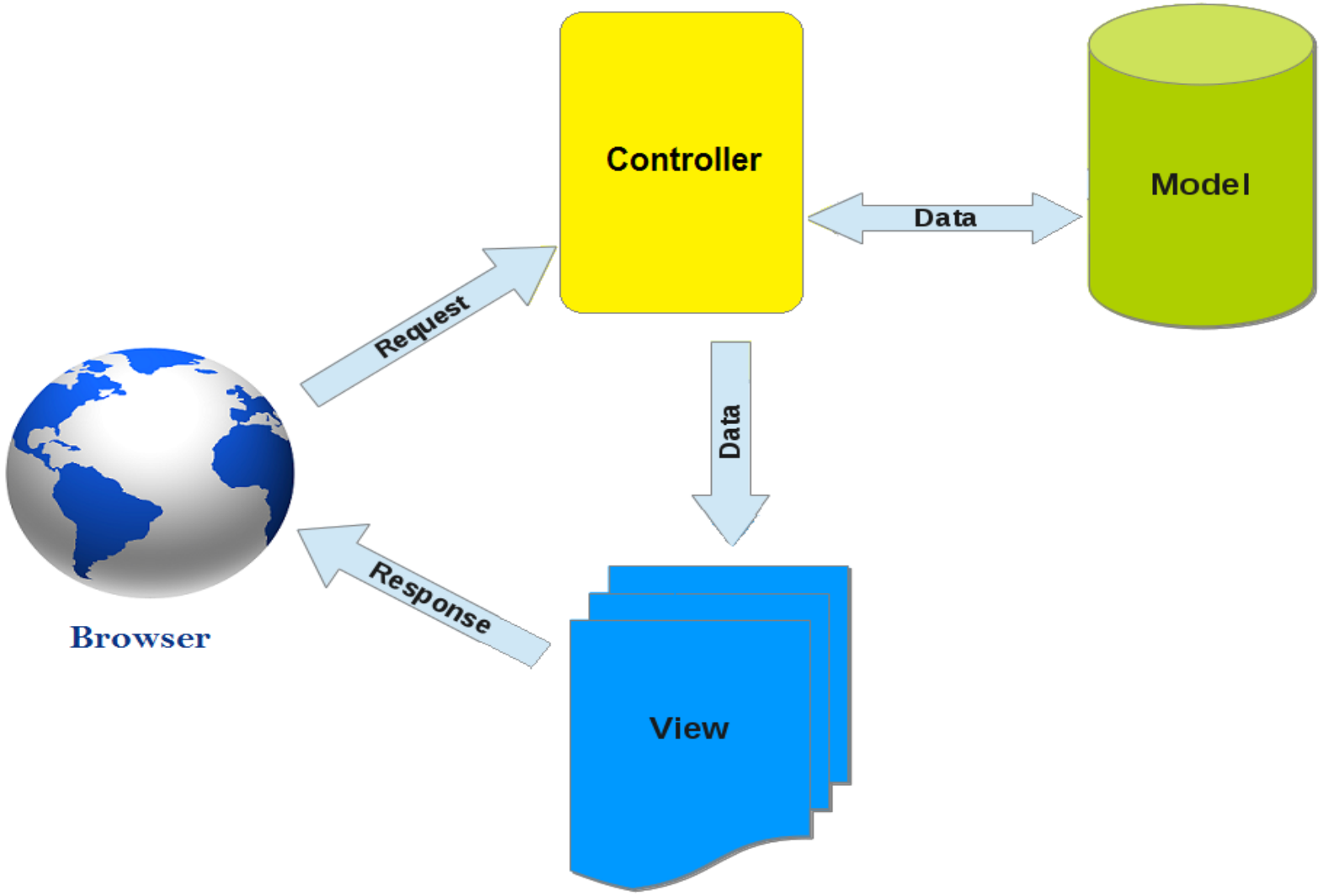
Architecture d'une application Web ASP.NET MVC

- ASP.NET MVC est un framework de programmation des applications web en MVC ajouté à ASP.NET en 2009
  - Dates:
    - Mars 2009: MVC 1.0
    - Mars 2010: MVC 2
    - Janvier 2011: MVC 3
    - Août 2012: MVC 4
    - Octobre 2013: MVC 5
    - Novembre 2015 : MVC 6
    - Juin 2016: ASP.NET Core 1.0
    - Aout 2017: ASP.NET Core 2.0

- Pourquoi utiliser ASP.NET MVC ?
  - ASP.NET MVC permet de
    - faciliter le développement de l'application
    - bien structurer l'application
    - concevoir une application qui prend en charge une infrastructure de routage riche (Front Controller)
    - faciliter les tests de l'application (TDD)

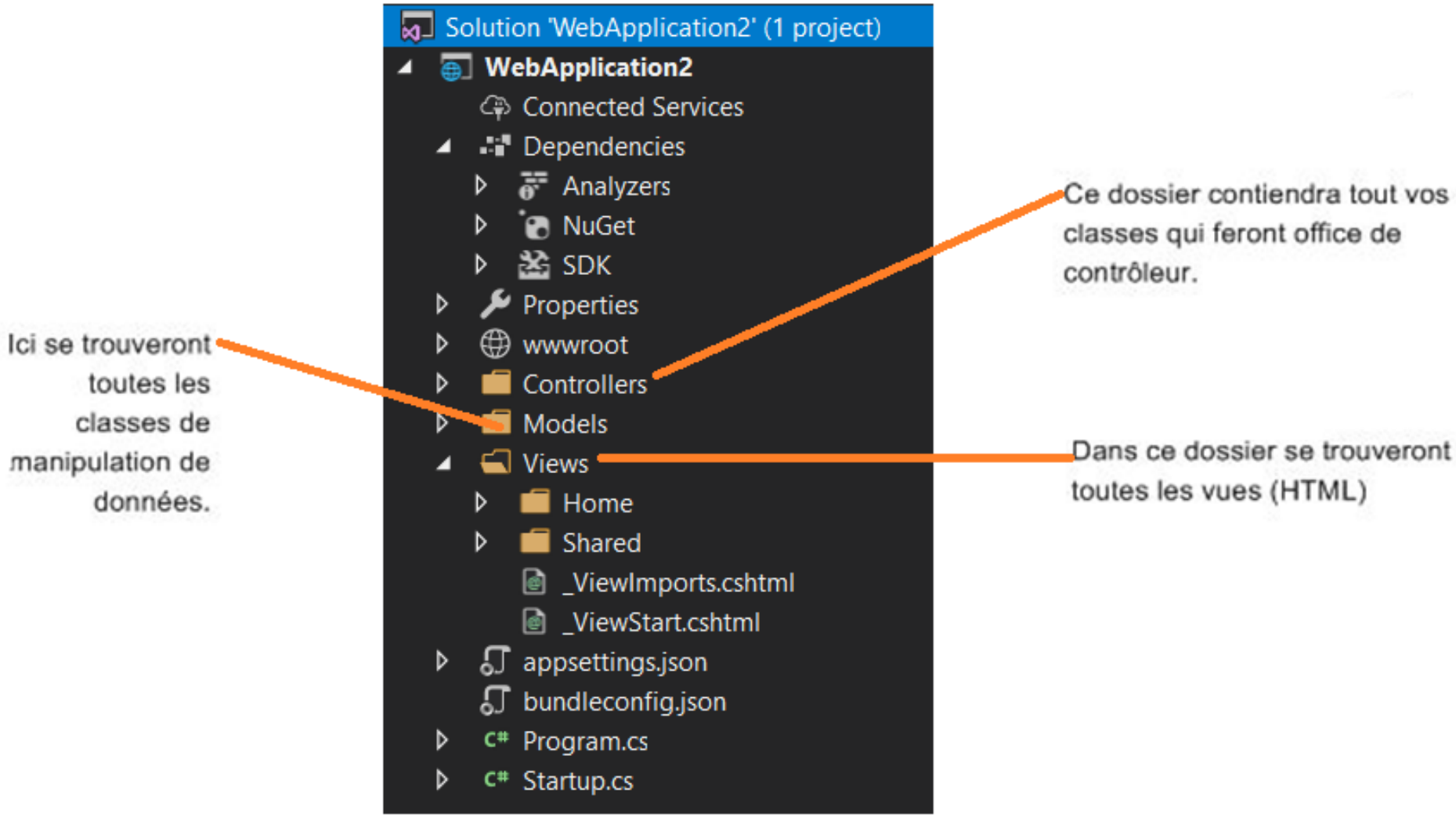
- MVC: Séparation entre
  - la présentation des données (V);
  - la définition du modèle de données (M);
  - la gestion des demandes de l'utilisateur (C).

Cycle d'exécution



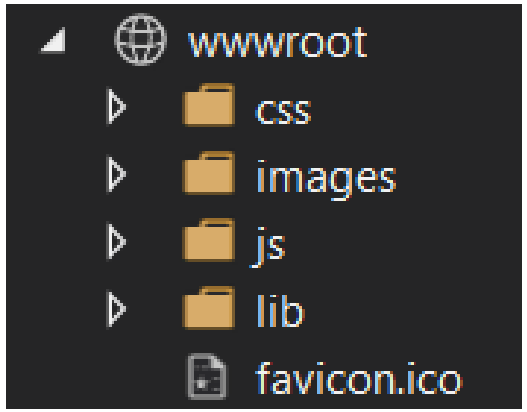
- Serveur IIS
  - un client qui présente l'avantage de simuler un site web ou une application dans sa globalité avant le déploiement massif sur plusieurs serveurs.

Structure d'un projet MVC



Le dossier wwwroot

- Ce dossier est la racine du projet
- Il ne contient que les fichiers statiques du site web.
- Il est donc important d'y placer
  - tous les styles (fichiers CSS),0
  - tous les scripts,
  - toutes les images,
  - toutes les librairies JavaScripts



Model-View-Controller

Modèles

- Les objets de modèle sont les parties de l'application qui implémentent la logique du domaine de données de l'application.
- Souvent, ils récupèrent l'état du modèle et le stockent dans une base de données.

Vues

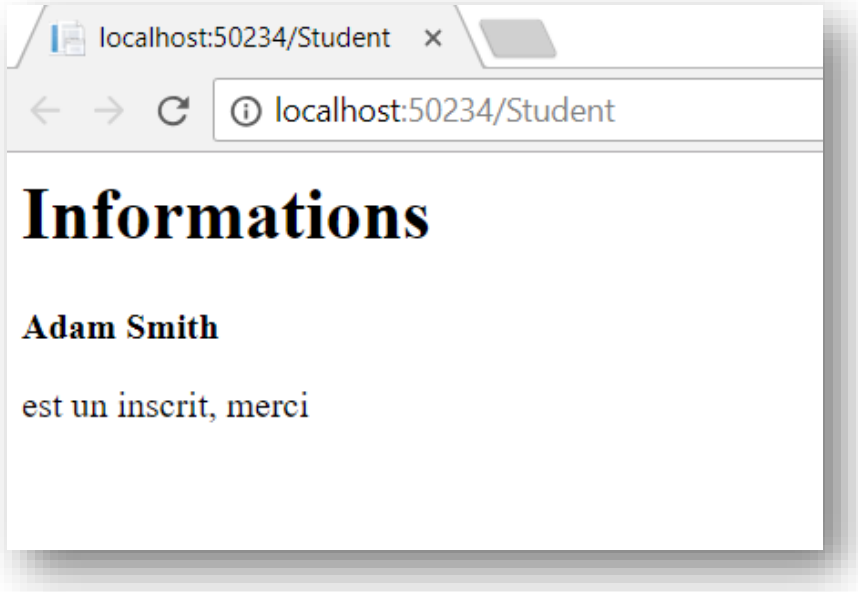
- Les vues sont les composants qui affichent l'interface utilisateur (IU) de l'application.
- En général, cette interface utilisateur est créée à partir des données du modèle.

Contrôleurs

- Les contrôleurs sont les composants qui gèrent les interventions de l'utilisateur, exploitent le modèle et finalement sélectionnent une vue permettant de restituer l'interface utilisateur.
- Dans une application MVC, la vue sert uniquement à afficher les informations ; le contrôleur gère les entrées et interactions de l'utilisateur, et y répond.

Model-View-Controller : Exemple

- Modèle
  - Les données d'un étudiant
- Vue
  - S'il s'agit d'un étudiant inscrit, le remercier
- Contrôleur
  - Récupérer les données d'un étudiant et afficher le rendu

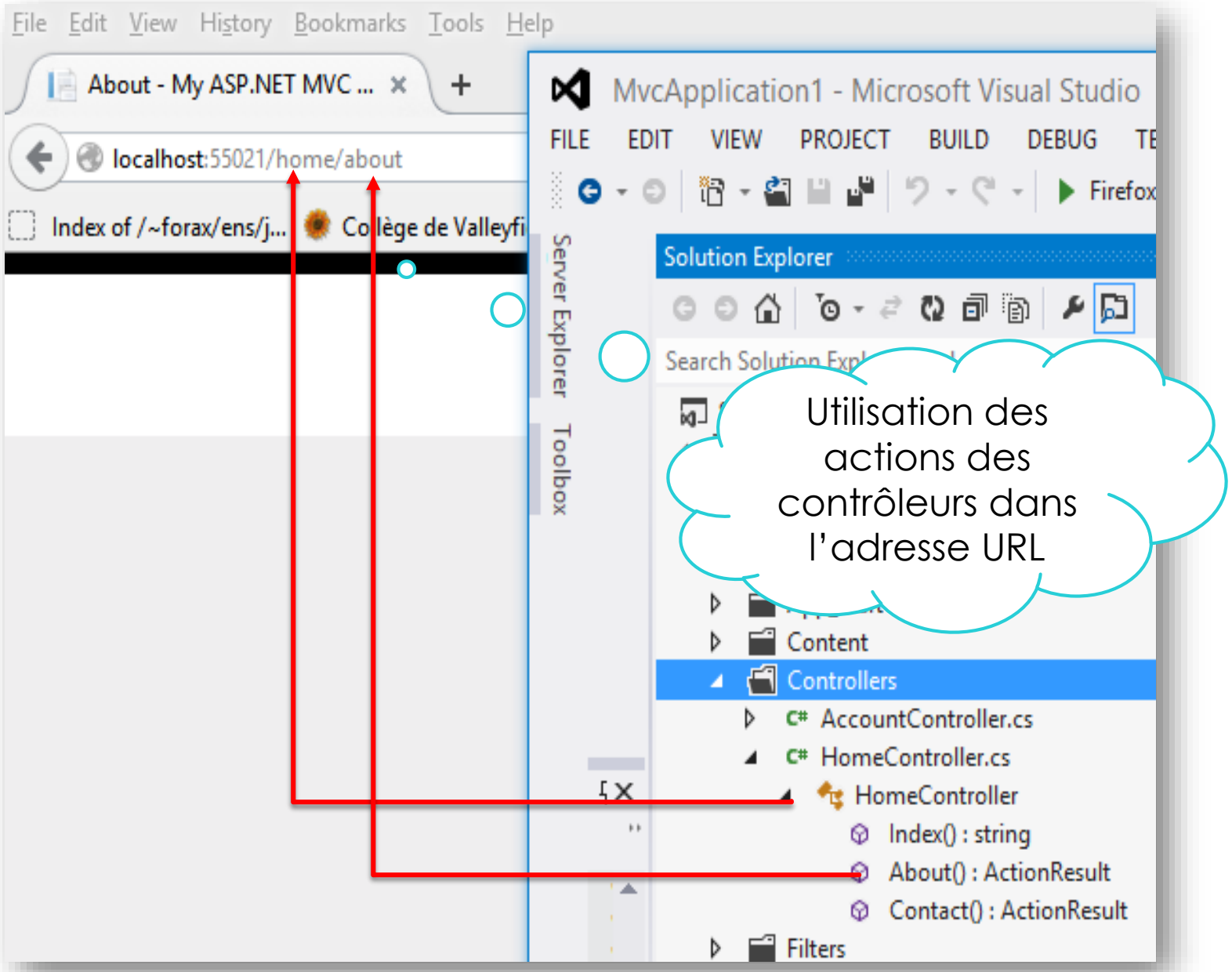



```
public class StudentController : Controller
    {
        public IActionResult Index()
        {
            Student student = new Student(1, "Adam Smith");

            string msg = "<h1>Informations</h1>";
            msg += "<b>" + student.Name + "</b><br />";
            msg += "<p>est un inscrit, merci</p>";

            return Content(msg, "text/html");
        }
    }
```

Model-View-Controller



Passer des paramètres en QueryString

- /Restaurant/ModifierRestaurant?id=1
  - 1 ère façon

```
public IActionResult ModifierRestaurant()
        {
            string idStr = Request.QueryString["id"];
```

  - Autre façon

```
public IActionResult ModifierRestaurant(int id)
        {
            //on utilise directement l'id
```