

# ***ASP.NET - INTRODUCTION***

## Sommaire

ASP.NET Kesako ? .....	1
ASP.NET WebForms .....	1
ASP.NET MVC .....	1
Choisir entre WebForms et MVC pour un projet.....	2
Différences entre WebForms et MVC .....	2
Cycle de vie .....	2
Dépendances .....	2
Testabilité .....	2
Gestion de l'état .....	3
Contrôles visuels.....	3
Evènements .....	3
Connaissances HTML, JS et CSS.....	3
Apprentissage .....	3
Historique du document .....	4
Crédits .....	4

## ASP.NET Kesako ?

---

ASP.NET est la plateforme de développement de Microsoft permettant la réalisation d'applications web.

Une plateforme de développement est un ensemble de composants permettant de construire une application (ici, une application web) qui comprend :

- un outil de développement (exemple : Visual Studio)
- une boîte à fonctionnalités pour développer (exemple : Le framework .NET)
- une logique de développement (exemple : respect de la structure d'une application web)
- un langage de programmation (exemple : C#)

ASP propose 2 manières de concevoir une application Web :

- ASP WebForms
- ASP MVC

ASP.NET est le socle de la plateforme de développement de Microsoft pour réaliser des applications web. WebForms et MVC sont deux logiques différentes de développement ASP.NET.

## ASP.NET WebForms

---

La 1<sup>ère</sup> version d'ASP.NET WebForms est sortie en 2002 et en est aujourd'hui (2018) à sa 5<sup>ème</sup> version. C'est une plateforme mature et robuste largement utilisée dans le monde professionnel. À l'origine, ASP.NET WebForms a été créé par Microsoft afin que les développeurs d'applications Windows puissent facilement créer des applications web à partir de leurs connaissances de l'environnement de développement Windows.

Concrètement, tout développeur d'applications Windows C# est capable de produire une application ASP.NET.

ASP.NET WebForms est une logique de développement très puissante pour réaliser des applications web. Elle ajoute une couche d'abstraction permettant de masquer une grande partie de la complexité du travail à réaliser ainsi que toutes les spécificités de la programmation d'applications web.

## ASP.NET MVC

---

ASP.NET MVC est plus récent et a fait son apparition en 2009. La logique de réalisation des applications web s'appuie sur un patron de conception très célèbre, MVC : Modèle Vue Contrôleur.

Là où les WebForms offrent des composants « préfabriqués » afin de simplifier votre vie de développeur, ASP.NET MVC propose un cadre pour la conception d'applications web.

L'utilisation du design pattern « MVC » impose de suivre une ligne directrice qui guidera le développeur dans la réalisation d'une application web.

ASP.NET MVC donne également plus de liberté dans le rendu du HTML en permettant notamment d'utiliser des composants externes (Javascript, CSS, appels AJAX).

## Choisir entre WebForms et MVC pour un projet

---

Si vous êtes à l'aise dans le développement d'applications Windows Forms et que le HTML/CSS est un peu obscur pour vous, ASP.NET WebForms est le meilleur choix. Il offre un panel de composants préfabriqués qui vous permettront de développer des pages web aussi rapidement qu'un formulaire Windows.

Si vous êtes à l'aise avec HTML/CSS et que vous souhaitez utiliser des composants externes (jQuery, Bootstrap...) ou personnalisés, ASP.NET MVC est le choix à faire. Avec ASP.NET MVC, les composants d'interface sont à développer entièrement.

A noter qu'il est possible, pour les gros projets, de coupler ASP.NET WebForms et ASP.NET MVC. Ce concept s'appelle « One ASP.NET ».

Pour vous aider dans votre choix, la suite de ce document détaille les principales différences entre ASP.NET WebForms et ASP.NET MVC

## Différences entre WebForms et MVC

---

ASP.NET WebForms et ASP.NET MVC diffèrent sur de nombreux points. Les différences les plus notables sont :

### Cycle de vie

---

#### **Webforms**

Le cycle de vie de la requête est fortement basé sur la « page active ».

Le fonctionnement est géré par des événements sur lesquels vous pouvez vous abonner.

#### **MVC**

Le cycle de vie est différent car il fait intervenir un contrôleur qui va charger des modèles de données puis sélectionner la vue à utiliser pour générer le rendu.

### Dépendances

---

#### **Webforms**

Les pages sont composées d'une vue et de code-behind. Ceci a un impact important car la vue et le contrôleur sont fortement dépendants.

#### **MVC**

Le contrôleur sélectionne la vue à afficher. Celle-ci ne dépend que du modèle qu'elle va utiliser pour le rendu.

### Testabilité

---

#### **Webforms**

Les pages sont difficiles à instancier en dehors d'un fonctionnement normal. Ceci a pour effet de les rendre impossible à tester de manière automatique.

#### **MVC**

Les contrôleurs sont de « simples » classes que vous pouvez instancier sans contexte HTTP. Elles sont donc plus faciles à tester avec des tests unitaires par exemple.

## Différences entre WebForms et MVC (Suite)

### Gestion de l'état

#### **Webforms**

Les contrôles stockent leur état dans le ViewState. Le ViewState est transmis dans chaque page. Il a tendance à grossir très rapidement.

#### **MVC**

MVC ne maintient pas d'état des pages, ni de ViewState. Vous avez le contrôle complet de votre application.

### Contrôles visuels

#### **Webforms**

Webforms est fortement basé sur des composants visuels comme Winforms. Ceci implique que le code est généré de manière automatique (sans grand contrôle).

#### **MVC**

MVC n'utilise pas d'évènement ni de code généré. Vous pouvez donc écrire du code HTML propre comme vous le souhaitez.

### Evènements

#### **Webforms**

L'extensibilité est gérée par les évènements. Vous devez donc vous abonner pour modifier certains comportements.

#### **MVC**

L'extensibilité est gérée par héritage ou des filtres. Le fonctionnement est simple à comprendre et à déboguer.

### Connaissances HTML, JS et CSS

#### **Webforms**

Etant donné la présence de contrôles visuels, vous n'avez pas besoin d'avoir de connaissances approfondies en HTML, JS et CSS pour créer des applications. Les composants vont générer le code pour vous.

#### **MVC**

L'absence de contrôles visuels et de drag & drop nécessite des compétences en développement HTML, CSS et Javascript. En contrepartie, vous pourrez écrire du code jQuery et des requêtes Ajax très simplement.

### Apprentissage

#### **Webforms**





Webforms est plus facile à apprendre et permet de créer rapidement des écrans visuels grâce au drag & drop de composants. Le framework est intéressant pour débiter avec le Web quand on connaît déjà WinForms.

#### **MVC**

Le temps d'apprentissage est un peu plus long car il faut bien comprendre les concepts du framework. Une fois que vous aurez bien compris ces concepts, vous pourrez développer très rapidement des applications web modernes.

--- FIN DU DOCUMENT ---

La reproduction partielle ou intégrale du présent document sur un support, quel qu'il soit, est formellement interdite sans l'accord écrit et préalable du Centre de Réadaptation de Mulhouse.

Légende des icônes	
	Information complémentaire
	Point d'attention particulier
	Intervention du formateur possible
	Lien vers une ressource externe

## Historique du document

Auteur	Date	Observations
MD v1.0.0	28/09/2018	Création du document

## Crédits

<http://www.arfp.asso.fr>