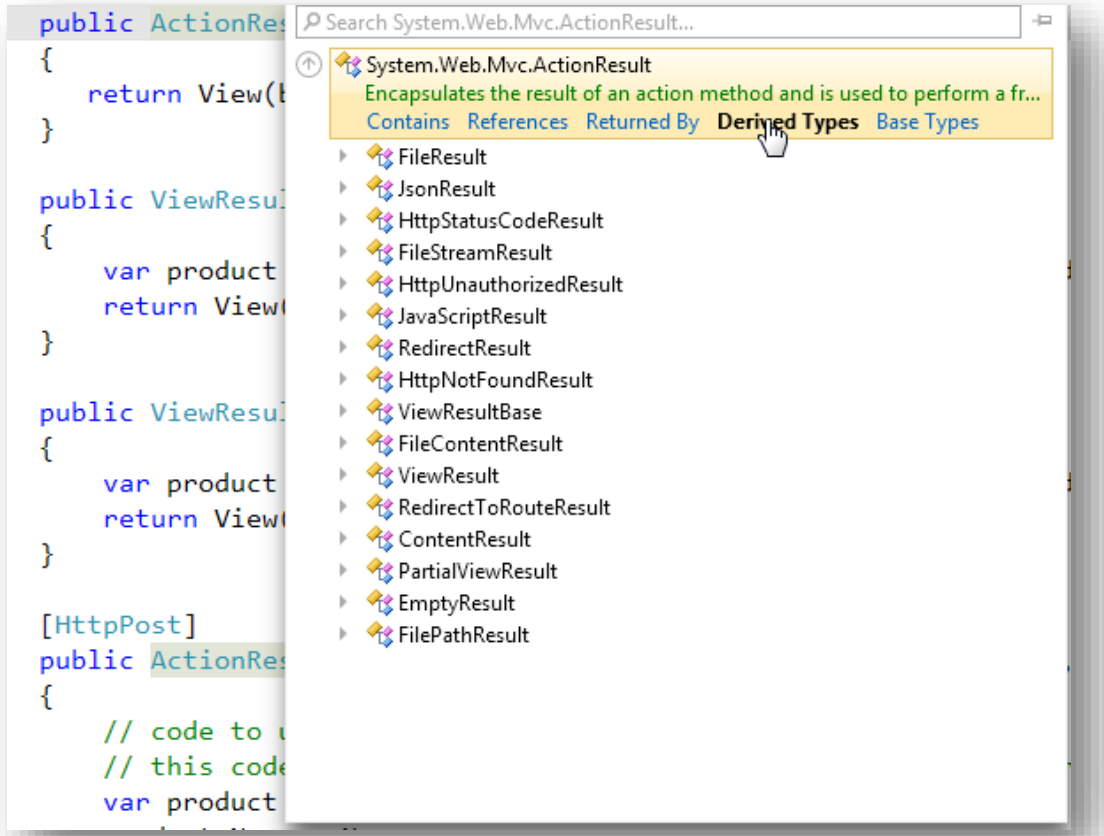
(Leçon 08)
Les contrôleurs - Model Binding

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

Contenu

- Les types dérivés de ActionResult
- Liaison de modèle (Model Binding)

Les types dérivés de ActionResult



ActionResult Types	Purpose	Examples of Use
ViewResult	Displays a View	return View();
PartialViewResult	Displays a Partial View	return PartialView();
RedirectToRouteResult	Redirects to a route	return RedirectToRoute("NamedRoute");
RedirectResult	Simple URL redirection	return Redirect("http://bing.com");
ContentResult	Returns raw text to the browser	return Content(rssString, "application/rss+xml");
FileResult	Returns binary data to browser	return File(@".\Doc1.docx", "application/vnd.ms-word");
JsonResult	Returns an object serialized as Json	return Json(Car);
JavaScriptResult	Returns JavaScript intended to run on the browser. Used with Ajax.	return JavaScript("$(#elementName).hide();");
HttpUnauthorizedResult	Returns code 401 – not authorized	return new HttpUnauthorizedResult();
EmptyResult	Doesn't return. Ends.	Return new EmptyResult();

Sélecteurs

- Ils sont des attributs appliqués aux actions d'un contrôleur afin de contrôler et influencer les actions demandées
- Exemples:
  - ActionName
  - HttpGet et HttpPost

```
[ActionName("Envoie")]
public ActionResult Contact()
{
    ViewBag.Message = "Your contact page.";

    return View("Contact");
}
```

```
//
// GET: /StoreManager/Create

public ActionResult Create()
{
    ViewBag.GenreId = new SelectList(db.Genres, "GenreId", "Name");
    ViewBag.ArtistId = new SelectList(db.Artists, "ArtistId", "Name");
    return View();
}

//
// POST: /StoreManager/Create

[HttpPost]
public ActionResult Create(Album album)
{
    if (ModelState.IsValid)
    {
        db.Albums.Add(album);
        db.SaveChanges();
        return RedirectToAction("Index");
    }

    ViewBag.GenreId = new SelectList(db.Genres, "GenreId", "Name", album.GenreId);
    ViewBag.ArtistId = new SelectList(db.Artists, "ArtistId", "Name", album.ArtistId);
    return View(album);
}
```

Liaison de modèle (Model Binding)

- Le «model binding » simplifie les actions de contrôleur
  - Introduction d'une couche d'abstraction pour:
    - remplir automatiquement les paramètres d'action du contrôleur
    - Effectuer le mappage des propriétés et du code de conversion de type
  - Avantages
    - Association simple des données de l'utilisateur avec le modèle
    - Application de plusieurs règles de validation avant l'enregistrement du modèle

- HttpPost vs HttpGet dans MVC
  - HttpGet affiche un formulaire pour l'entrée utilisateur
  - HttpPost est utilisé lorsque l'utilisateur soumet le formulaire affiché

- L'action associée à HttpPost reçoit généralement un objet du modèle en paramètre
  - Le model binding remplir automatiquement ce paramètre à partir des données du formulaire

- Valider les données
  - Utilisation de l'objet ModelState avec la propriété IsValid
  - On vérifie que le model ne contient pas d'erreur pour l'ajout en base de données ModelState.IsValid

```
[HttpPost]
public ActionResult Create(Client client)
{
    if (ModelState.IsValid)
    {
        db.Clients.Add(client);
        db.SaveChanges();
        return RedirectToAction("Index");
    }
```

Bind

Exclude, Include

- Permet de contrôler quelles propriétés sont éditables.
- Exemple: Vous supprimé StudentID à partir de la liaison d'attribut, car StudentID est la valeur de clé primaire et SQL Server définira automatiquement sa valeur lors de l'insertion

```
[HttpPost]
public ActionResult Create([Bind(Include = "LastName, FirstMidName, EnrollmentDate")]Student student)
{
    if (ModelState.IsValid)
    {
        db.Students.Add(student);
        db.SaveChanges();
        return RedirectToAction("Index");
    }
```