

## Action Results

Type	Helper Method
ViewResult	<b>View()</b>
PartialViewResult	<b>PartialView()</b>
ContentResult	<b>Content()</b>
RedirectResult	<b>Redirect()</b>
RedirectToRouteResult	<b>RedirectToAction()</b>
JsonResult	<b>Json()</b>
FileResult	<b>File()</b>
HttpNotFoundResult	<b>HttpNotFound()</b>
EmptyResult	

## Action Parameters

### Sources

- Embedded in the URL: /movies/edit/1
- In the query string: /movies/edit?id=1
- In the form data

## Convention-based Routes

```
routes.MapRoute(
    "MoviesByReleaseDate",
    "movies/released/{year}/{month}",
    new {
        controller = "Movies",
        action = "MoviesReleaseByDate"
    },
    new {
        year = @"\d{4}", month = @"\d{2}"
    },
    isFavorite = false;
}
```

## Attribute Routes

```
[Route("movies/released/{year}/{month}")]
public ActionResult MoviesByReleaseDate(int year, int month)
{
}
```

To apply a constraint use a colon:

```
month:regex(\d{2}):range(1, 12)
```

# ASP.NET MVC Fundamentals

By: Mosh Hamedani

## Passing Data to Views

Avoid using `ViewData` and `ViewBag` because they are fragile. Plus, you have to do extra casting, which makes your code ugly. Pass a model (or a view model) directly to a view:

```
return View(movie);
```

## Razor Views

```
@if (...)
{
    // C# code or HTML
}
```

```
@foreach (...)
{
}
```

Render a class (or any attributes) conditionally:

```
@{
    var className = Model.Customers.Count > 5 ? "popular" : null;
}
<h2 class="@className">...</h2>
```

# ASP.NET MVC Fundamentals

By: Mosh Hamedani

## Partial Views

To render:

```
@Html.Partial("_NavBar")
```