# Using Knockout with the MVC4 Template

Written by Paul Wroe, Brent Stewart, Nate Eckardt 4/4/2013

"*Developers familiar with Ruby on Rails, ASP.NET MVC, or other MV* technologies may see MVVM as a real-time form of MVC with declarative syntax. In another sense, you can think of KO as a general way to make UIs for editing JSON data… whatever works for you :)"* –from the knockout documentation found at www.knockoutjs.com

Knockout (regular Knockout) is a user experience technology. It is useful for arranging data and dynamically presenting choices to the user. For example, a user can be presented with an editable list and upon editing, the changes are reflected in the viewmodel. Regular Knockout can be useful for showing updated choices and filtering already present data as well as facilitating the return of a Json object.

KnockoutMVC (KMVC) is a wrapper for regular Knockout. KMVC is used with the Razor engine to auto generate front end knockout.js and HTML attributes.

Regular Knockout 2.1.0 is included by default in the MVC4 template but it is not added to the BundleConfig file in App_Start. Additionally, KMVC is not included.

**How to include KMVC into a project:**

Install with package manager PM> *Install-Package kMVC*
The following files get modified.

```
Attempting to resolve dependency 'knockoutjs (≥ 2.1.0)'.
Attempting to resolve dependency 'Knockout.Mapping (≥ 2.3.2)'.
Attempting to resolve dependency 'DelegateDecompiler (≥ 0.6.0.0)'.
Attempting to resolve dependency 'Mono.Reflection (≥ 1.0.0.0)'.
Attempting to resolve dependency 'Newtonsoft.Json (≥ 4.5.11)'.
Attempting to resolve dependency 'jQuery (≥ 1.7)'.
Successfully installed 'Knockout.Mapping 2.3.4'.
Successfully installed 'Mono.Reflection 1.0.0.0'.
Successfully installed 'DelegateDecompiler 0.6.0.0'.
Successfully installed 'Newtonsoft.Json 4.5.11'.
Successfully installed 'jQuery 1.7.2'.
Successfully installed 'kMVC 0.5.4'.
Successfully added 'Knockout.Mapping 2.3.4' to testRun.
Successfully added 'Mono.Reflection 1.0.0.0' to testRun.
Successfully added 'DelegateDecompiler 0.6.0.0' to testRun.
Successfully removed 'Newtonsoft.Json 4.5.6' from testRun.
Successfully added 'Newtonsoft.Json 4.5.11' to testRun.
Successfully removed 'jQuery 1.7.1.1' from testRun.
Successfully added 'jQuery 1.7.2' to testRun.
Successfully added 'kMVC 0.5.4' to testRun.
Successfully uninstalled 'jQuery 1.7.1.1'.
```

**How to implement KMVC in an MVC4 project:**

1. **Create a bundle for the knockout scripts in BundleConfig.cs:**
```
bundles.Add(new ScriptBundle("~/bundles/knockoutGroup").Include(
        "~/Scripts/knockout-{version}.js",
        "~/Scripts/knockout.mapping-latest.js",
        "~/Scripts/perpetuum.knockout.js"
        ));
```

2. **Add a using statement to the cshtml page**:
```
@using PerpetuumSoft.Knockout;
```

3. **Add Scripts.Render for jquery and knockout:**
```
@Scripts.Render("~/Scripts/bundles/jquery")
@Scripts.Render("~/bundles/knockoutGroup")
```

4. **Add reference to Html.CreateKnockoutContext() for readability:**
```
@{
       var ko = Html.CreateKnockoutContext();
}
```

5. **Implement KMVC features in cshtml:**
```
(from KMVC examples)
<h2>Index</h2>
Number : @ko.Html.Span(m => m.Number)<br />
@ko.Html.Button("Inc 1", "Increment", "ParametersToServer", new {value = 1})
@ko.Html.Button("Inc 2", "Increment", "ParametersToServer", new {value = 2})
@ko.Html.Button("Inc 3", "Increment", "ParametersToServer", new {value = 3})
<br />
```

   (see appendix 1 for full source code)

**Regular Knockout Annotation in .cshtml:**
@Html.TextBox("username", Model.Username, new { data_bind = "value: name" })
The key here is adding new { data_bind = "value: name"} attribute
**MVC will convert the underscore to a dash**.  (Thanks masterchief117 for finding that)

**Tutorials and sources:**

- Knockout home page http://knockoutjs.com/
  However, it does not show how to implement in MVC4 and is a little lacking if you want to implement from the tutorial – some of the environment requirements are not discussed.

- KMVC, a wrapper for regular Knockout, is covered here: http://knockoutmvc.com/

- Information integrating Knockout into MVC4
  http://blogs.msdn.com/b/amar/archive/2012/12/12/hello-world-with-knockout-js-and-asp-net-mvc-4.aspx

## Appendix 1 – Parameters to server example from knockoutmvc.com/ParametersToServer

### Model

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;

namespace KnockoutTest.Models
{
    public class ParametersToServerModel
    {
        public int Number { get; set;}
        public string String { get; set; }

        public void Increment (int value)
        {
            Number += value;
        }

        public void AddToString(string s, int count)
        {
            for (int i = 0; i < count; i++)
            {
                String += s;
            }
        }

    }
}
```

### View

```
@using PerpetuumSoft.Knockout;
@model KnockoutTest.Models.ParametersToServerModel
@Scripts.Render("~/Scripts/bundles/jquery")
@Scripts.Render("~/bundles/knockoutGroup")
@{
    ViewBag.Title = "Index";
    var ko = Html.CreateKnockoutContext();
}

<h2>Index</h2>
Number : @ko.Html.Span(m => m.Number)<br />
@ko.Html.Button("Inc 1", "Increment", "ParametersToServer", new {value = 1})
@ko.Html.Button("Inc 2", "Increment", "ParametersToServer", new {value = 2})
@ko.Html.Button("Inc 3", "Increment", "ParametersToServer", new {value = 3})
<br />
<br />
String: @ko.Html.Span(m => m.String)<br />
@ko.Html.HyperlinkButton("Add 1 'a'", "AddToString", "ParametersToServer", new {letter = 'a', count = 1
})<br />
@ko.Html.HyperlinkButton("Add 2 'b'", "AddToString", "ParametersToServer", new {letter = 'b', count = 2
})<br />
@ko.Html.HyperlinkButton("Add 3 'c'", "AddToString", "ParametersToServer", new {letter = 'c', count = 3
})<br />

@ko.Apply(Model)
```

## Controller

```
using KnockoutTest.Models;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.Mvc;

namespace KnockoutTest.Controllers
{
    public class ParametersToServerController : Controller
    {
        //
        // GET: /ParametersToServe/

        public ActionResult Index()
        {
            return View(new ParametersToServerModel());
        }

        public ActionResult Increment(ParametersToServerModel model, int value)
        {
            model.Increment(value);
            return Json(model);
        }

        public ActionResult AddToString(ParametersToServerModel model, char letter, int count)
        {
            model.AddToString(letter + "", count);
            return Json(model);
        }
    }
}
```