Phần 13: Thêm sắp xếp (dự án thực tế)

Như vậy trong bài trước, bạn đã học 1 số cách để thêm sắp xếp theo các trường thuộc tính của lớp và ví dụ kèm theo với giải thích chi tiết. Tuy nhiên, chừng đó là chưa đủ để bạn phát triển ứng dụng Web trên thực tế hoặc các dự án Web công nghiệp. Nội dung bài này tiếp tục mở rộng phần sắp xếp giúp bạn có thể có nhiều tính năng gần giống dự án Web thực tế hơn.

# Biểu tượng tăng giảm bên cạnh các cột sắp xếp

## Sửa tập tin Shared/\_Layout.cshtml

Để chèn các biểu tượng này, bạn cần nhúng vào dự án tập tin font-awesome.min.[**css**](https://www.dammio.com/glossary/css). Bạn mở tập tin Shared/\_Layout.cshtml để thêm dòng sau.

<!DOCTYPE html>

<html>

<head>

<meta charset="utf-8" />

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>@ViewBag.Title - My ASP.NET Application</title>

@Styles.Render("~/Content/css")

@Scripts.Render("~/bundles/modernizr")

<!--Đoạn cần thêm-->

<link href="//maxcdn.bootstrapcdn.com/font-awesome/4.1.0/css/font-awesome.min.css" rel="stylesheet">

<!-- Kết thúc -->

</head>

...

Tại sao lại thêm vào tập tin \_Layout.cshtml, lý do đơn giản là các View của bạn đều mặc định dùng Layout này làm Layout chính (tương tự Master Page ở Web Form hay Master Slide ở Powerpoint). Nếu bạn thay đổi tập tin Layout thì bạn cũng tìm chính tập tin đó để nhúng font-awesome.min.[css](https://www.dammio.com/glossary/css).

## Sửa tập tin Controller/LinkController.cs

Bạn chỉ cần sửa phương thức Index ở tập tin LinkController.cs như sau. Bạn chú ý phần 2.3, tức là chúng ta so điều kiện **sortOrder == “desc” && sortProperty == item.Name** để biết thuộc tính nào cần sắp giảm để thêm biểu tượng giảm **i class=’fa fa-fw fa-sort-desc’** để sắp xếp.

public ActionResult Index(string sortProperty, string sortOrder)

{

// 1. Tạo biến ViewBag SortOrder để giữ trạng thái sắp tăng hay giảm

ViewBag.SortOrder = String.IsNullOrEmpty(sortOrder) ? "desc" : "";

// 2. Lấy tất cả tên thuộc tính của lớp Link (LinkID, LinkName, LinkURL,...)

var properties = typeof(Link).GetProperties();

foreach (var item in properties)

{

// 2.1 Kiểm tra xem thuộc tính nào là virtual (public virtual Category Category...)

var isVirtual = item.GetAccessors()[0].IsVirtual;

// 2.2. Thuộc tính bình thường thì cho phép sắp xếp

if (!isVirtual)

{

// 2.3. So thuộc tính sortProperty và sortOrder để biết thuộc tính nào cần thay biểu tượng sắp giảm

if (sortOrder == "desc" && sortProperty == item.Name)

{

ViewBag.Headings += "<th><a href='?sortProperty=" + item.Name + "&sortOrder=" +

ViewBag.SortOrder + "'>" + item.Name + "<i class='fa fa-fw fa-sort-desc'></i></th></a></th>";

}

else

{

ViewBag.Headings += "<th><a href='?sortProperty=" + item.Name + "&sortOrder=" +

ViewBag.SortOrder + "'>" + item.Name + "<i class='fa fa-fw fa-sort-asc'></a></th>";

}

}

// 2.4. Thuộc tính virtual (public virtual Category Category...) thì không sắp xếp được

// cho nên không cần tạo liên kết

else ViewBag.Headings += "<th>" + item.Name + "</th>";

}

// 3. Truy vấn lấy tất cả đường dẫn

var links = from l in db.Links

select l;

// 4. Tạo thuộc tính sắp xếp mặc định là "LinkID"

if (String.IsNullOrEmpty(sortProperty)) sortProperty = "LinkID";

// 5. Sắp xếp tăng/giảm bằng phương thức OrderBy sử dụng trong thư viện Dynamic LINQ

if (sortOrder == "desc") links = links.OrderBy(sortProperty + " desc");

else links = links.OrderBy(sortProperty);

// 6. Trả kết quả về Views

return View(links.ToList());

}

# Chạy dự án, xem kết quả

Bạn Build lại dự án, sau đó chạy đường dẫn **http://localhost:45033/Link/Index** để xem kết quả.



Bạn có thể click lên bất kỳ cột nào để sắp xếp thì thấy chỉ có biểu tượng cột đó thay đổi tăng hay giảm. Như vậy khá tuyệt phải không nào.

# Bỏ bớt hay thay đổi thứ tự hiển thị các cột sắp xếp

Nếu bạn muốn thay đổi hay bỏ bớt số lượng các cột sắp xếp, bạn vẫn có thể thực hiện ở phần này. Bạn có thể thấy đoạn **var properties = typeof(Link).GetProperties();** (phần 2, phương thức Index, tập tin LinkController.cs) dùng để lấy thuộc tính của lớp Link. Như vậy, chúng ta chỉ cần thay đổi thứ tự hoặc bỏ bớt thuộc tính không hiển thị ở đây hoặc bạn có thể chỉnh sửa toàn bộ code phần này để hiển thị dự liệu theo ý bạn.

## Bỏ bớt thuộc tính

Để bỏ bớt thuộc tính đơn giản trong vòng **foreach(item in properties)**, bạn chỉ cần thêm mệnh đề if kiểm tra nếu đúng là thuộc tính cần bỏ thì chọn continue, tức không cần làm gì cả. Bạn chỉ cần thêm phần 2.0 như đoạn mã sau đây.

public ActionResult Index(string sortProperty, string sortOrder)

        {

            // 1. Tạo biến ViewBag SortOrder để giữ trạng thái sắp tăng hay giảm

            ViewBag.SortOrder = String.IsNullOrEmpty(sortOrder) ? "desc" : "";

            // 2. Lấy tất cả tên thuộc tính của lớp Link (LinkID, LinkName, LinkURL,...)

            var properties = typeof(Link).GetProperties();

            foreach (var item in properties)

            {

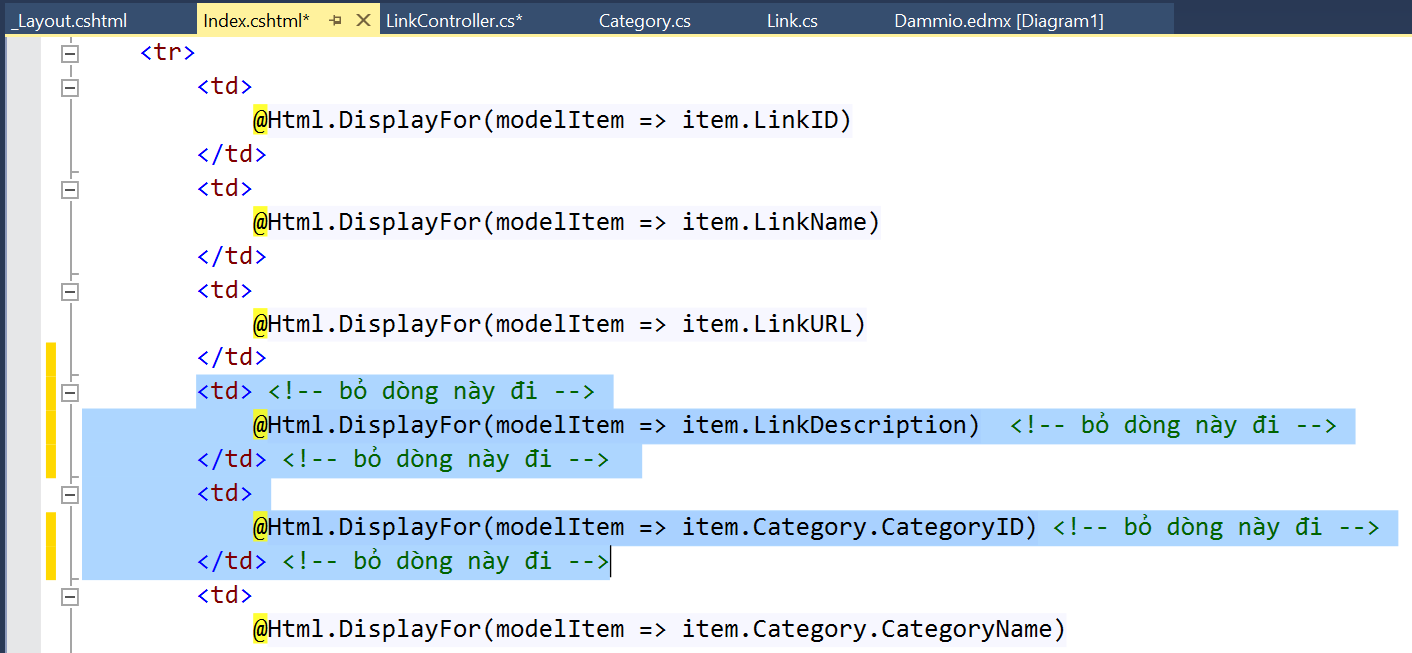
                // 2.0 Bỏ bớt thuộc tính LinkDescription khi hiển thị

                if (item.Name == "LinkDescription") continue;

                if (item.Name == "CategoryID") continue;

...

Lưu ý, ở tập tin **Views/Link/Index.cshtml** bạn cũng bỏ các trường hiển thị tương ứng như hình sau.



Build dự án Web, chạy đường dẫn http://localhost:xxxx/Link/Index, bạn cũng có thể thấy kết quả thay đổi như sau.



## Thay đổi thứ tự hiển thị các cột thuộc tính

Để thay đổi thứ tự các cột, bạn có thể thay đổi code để thay đổi trên View (Index.cshtml) hoặc là thay đổi code ở phương thức Index của tập tin LinkController.cs.

Lưu ý đây có thể không phải là phương án tốt nhất nhưng cũng là cách để các bạn tham khảo.

Bạn sửa **phương thức Index của tập tin LinkController.cs** như sau.

public ActionResult Index(string sortProperty, string sortOrder)

{

// 1. Tạo biến ViewBag SortOrder để giữ trạng thái sắp tăng hay giảm

ViewBag.SortOrder = String.IsNullOrEmpty(sortOrder) ? "desc" : "";

// 2. Lấy tất cả tên thuộc tính của lớp Link (LinkID, LinkName, LinkURL,...)

var properties = typeof(Link).GetProperties();

// 2.0. Tạo 1 danh List với mỗi phần tử là kiểu Tuple

// Tuple<string, bool, int> với tham số lần lượt là <Name, IsVirtual, Order>

// tức là Tên thuộc tính, Thuộc tính là virtual hay không và Thứ tự thuộc tính

List<Tuple<string, bool, int>> list = new List<Tuple<string, bool, int>>();

foreach (var item in properties)

{

int order = 999; //order mặc định sẽ xếp hạng thấp nhất, 999 chỉ là con số để xem là thấp nhất

var isVirtual = item.GetAccessors()[0].IsVirtual;

if (item.Name == "LinkName") order = 1;

if (item.Name == "LinkID") order = 2;

if (item.Name == "LinkDescription") order = 3;

if (item.Name == "LinkURL") order = 4;

if (item.Name == "CategoryID") continue; // Không hiển thị cột này

Tuple<string, bool, int> t = new Tuple<string, bool, int>(item.Name, isVirtual, order);

list.Add(t);

}

// 2.1. Sắp xếp theo thứ tự ở trên

list = list.OrderBy(x => x.Item3).ToList();

foreach (var item in list)

{

// 2.2. Thuộc tính bình thường thì cho phép sắp xếp

if (!item.Item2) // Item2 dùng để kiểm tra thuộc tính ảo hay không?

{

// 2.3. So thuộc tính sortProperty và sortOrder để biết thuộc tính nào cần thay biểu tượng sắp giảm

if (sortOrder == "desc" && sortProperty == item.Item1)

{

ViewBag.Headings += "<th><a href='?sortProperty=" + item.Item1 + "&sortOrder=" +

ViewBag.SortOrder + "'>" + item.Item1 + "<i class='fa fa-fw fa-sort-desc'></i></th></a></th>";

}

else

{

ViewBag.Headings += "<th><a href='?sortProperty=" + item.Item1 + "&sortOrder=" +

ViewBag.SortOrder + "'>" + item.Item1 + "<i class='fa fa-fw fa-sort-asc'></a></th>";

}

}

// 2.4. Thuộc tính virtual (public virtual Category Category...) thì không sắp xếp được

// cho nên không cần tạo liên kết

else ViewBag.Headings += "<th>" + item.Item1 + "</th>";

}

// 3. Truy vấn lấy tất cả đường dẫn

var links = from l in db.Links

select l;

// 4. Tạo thuộc tính sắp xếp mặc định là "LinkID"

if (String.IsNullOrEmpty(sortProperty)) sortProperty = "LinkID";

// 5. Sắp xếp tăng/giảm bằng phương thức OrderBy sử dụng trong thư viện Dynamic LINQ

if (sortOrder == "desc") links = links.OrderBy(sortProperty + " desc");

else links = links.OrderBy(sortProperty);

// 6. Trả kết quả về Views

return View(links.ToList());

}

Trong đoạn code trên, bạn có thể phần 2.0, chúng ta tạo **1 danh sách List với mỗi phần tử là 1 Tuple với 3 tham số: Tên thuộc tính (string), Thuộc tính có phải là virtual hay không (bool) và Thứ tự sắp xếp (int)**. Bạn có thể định nghĩa thứ tự thuộc tính trong đó hoặc dùng phương pháp enum, sau đó thay đổi code kèm theo ở phần 2.1, 2.2, 2.3 và 2.4.

Tiếp theo, bạn cũng phải sửa **tập tin Views/Link/Index.cshtml** tương ứng như sau.

@model IEnumerable<DammioMVC1.Models.Link>

@{

ViewBag.Title = "Index";

}

<h2>Index</h2>

<p>

@Html.ActionLink("Create New", "Create")

</p>

<table class="table">

<tr>

@Html.Raw(ViewBag.Headings)

</tr>

@foreach (var item in Model) {

<tr>

<td>

@Html.DisplayFor(modelItem => item.LinkName)

</td>

<td>

@Html.DisplayFor(modelItem => item.LinkID)

</td>

<td>

@Html.DisplayFor(modelItem => item.LinkDescription)

</td>

<td>

@Html.DisplayFor(modelItem => item.LinkURL)

</td>

<td>

@Html.DisplayFor(modelItem => item.Category.CategoryName)

</td>

<td>

@Html.ActionLink("Edit", "Edit", new { id=item.LinkID }) |

@Html.ActionLink("Details", "Details", new { id=item.LinkID }) |

@Html.ActionLink("Delete", "Delete", new { id=item.LinkID })

</td>

</tr>

}

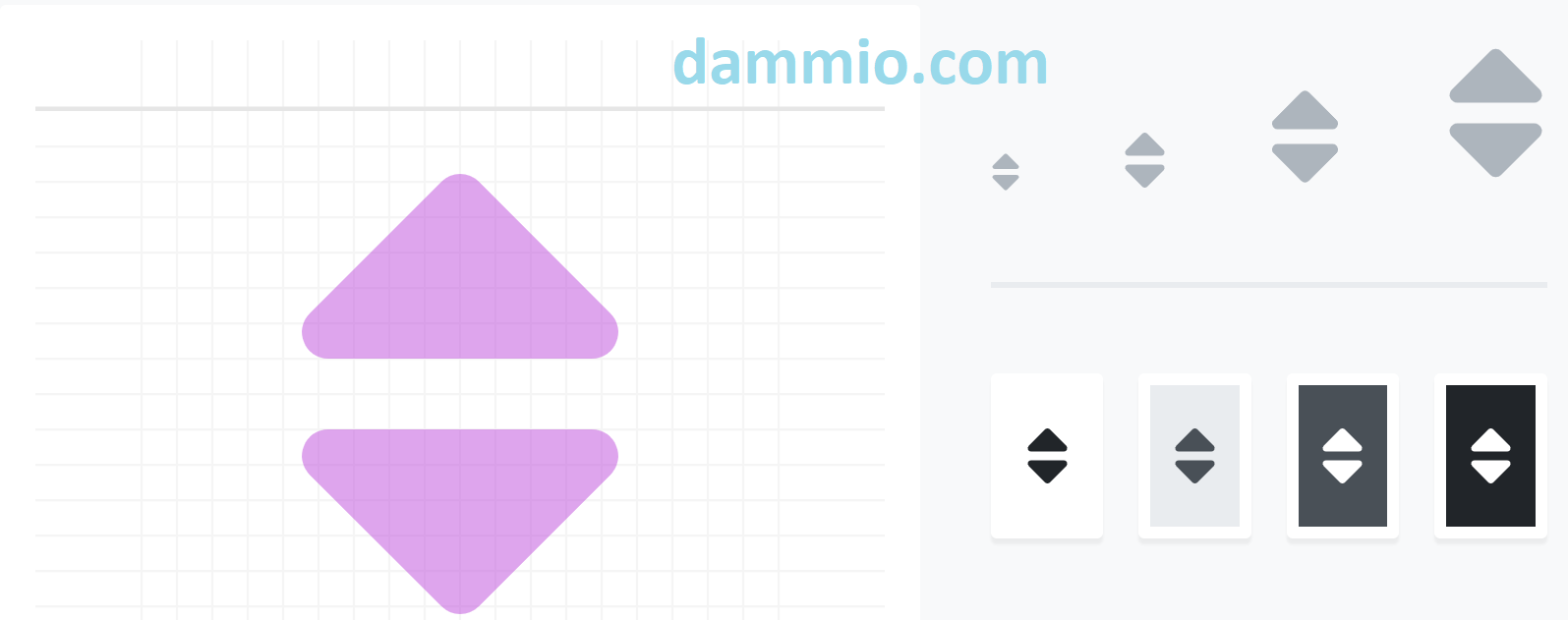
</table>

Cuối cùng, bạn build dự án, sau đó chạy đường dẫn **http://localhost:xxxx/Link/Index** để xem kết quả thay đổi.



# Biểu tượng tăng/giảm và bình thường

Các phần trước, bạn đã học thêm các biểu tượng tăng/giảm ở các cột sắp xếp. Ngoài ra, bạn có thể thêm biểu tượng sort như hình sau, tức là lúc đó là chúng ta chưa sắp xếp gì cả.



## Sửa phương thức Index của tập tin Controller/LinkController.cs

public ActionResult Index(string sortProperty, string sortOrder)

       {

           // 1. Tạo biến ViewBag SortOrder để giữ trạng thái sắp tăng hay giảm

           //ViewBag.SortOrder = String.IsNullOrEmpty(sortOrder) ? "desc" : sortOrder;

           if (sortOrder == "asc") ViewBag.SortOrder = "desc";

           if (sortOrder == "desc") ViewBag.SortOrder = "";

           if (sortOrder == "") ViewBag.SortOrder = "asc";

           // 2. Lấy tất cả tên thuộc tính của lớp Link (LinkID, LinkName, LinkURL,...)

           var properties = typeof(Link).GetProperties();

           // 2.0. Tạo 1 danh List với mỗi phần tử là kiểu Tuple

           // Tuple<string, bool, int> với tham số lần lượt là <Name, IsVirtual, Order>

           // tức là Tên thuộc tính, Thuộc tính là virtual hay không và Thứ tự thuộc tính

           List<Tuple<string, bool, int>> list = new List<Tuple<string, bool, int>>();

           foreach (var item in properties)

           {

               int order = 999;

               var isVirtual = item.GetAccessors()[0].IsVirtual;

               if (item.Name == "LinkName") order = 1;

               if (item.Name == "LinkID") order = 2;

               if (item.Name == "LinkDescription") order = 3;

               if (item.Name == "LinkURL") order = 4;

               if (item.Name == "CategoryID") continue; // Không hiển thị cột này

               Tuple<string, bool, int> t = new Tuple<string, bool, int>(item.Name, isVirtual, order);

               list.Add(t);

           }

           // 2.1. Sắp xếp theo thứ tự ở trên

           list = list.OrderBy(x => x.Item3).ToList();

           foreach (var item in list)

           {

               // 2.2. Thuộc tính bình thường thì cho phép sắp xếp

               if (!item.Item2) // Item2 dùng để kiểm tra thuộc tính ảo hay không?

               {

                   // 2.3. So thuộc tính sortProperty và sortOrder để biết thuộc tính nào cần thay biểu tượng sắp giảm

                   if (sortOrder == "desc" && sortProperty == item.Item1)

                   {

                       ViewBag.Headings += "<th><a href='?sortProperty=" + item.Item1 + "&sortOrder=" +

                           ViewBag.SortOrder + "'>" + item.Item1 + "<i class='fa fa-fw fa-sort-desc'></i></th></a></th>";

                   }

                   else if (sortOrder == "asc" && sortProperty == item.Item1)

                   {

                       ViewBag.Headings += "<th><a href='?sortProperty=" + item.Item1 + "&sortOrder=" +

                           ViewBag.SortOrder + "'>" + item.Item1 + "<i class='fa fa-fw fa-sort-asc'></a></th>";

                   }

                   else

                   {

                       ViewBag.Headings += "<th><a href='?sortProperty=" + item.Item1 + "&sortOrder=" +

                          ViewBag.SortOrder + "'>" + item.Item1 + "<i class='fa fa-fw fa-sort'></a></th>";

                   }

               }

               // 2.4. Thuộc tính virtual (public virtual Category Category...) thì không sắp xếp được

               // cho nên không cần tạo liên kết

               else ViewBag.Headings += "<th>" + item.Item1 + "</th>";

           }

           // 3. Truy vấn lấy tất cả đường dẫn

           var links = from l in db.Links

                 select l;

           // 4. Tạo thuộc tính sắp xếp mặc định là "LinkID"

           if (String.IsNullOrEmpty(sortProperty)) sortProperty = "LinkID";

           // 5. Sắp xếp tăng/giảm bằng phương thức OrderBy sử dụng trong thư viện Dynamic LINQ

           if (sortOrder == "desc") links = links.OrderBy(sortProperty + " desc");

           else if (sortOrder == "asc") links = links.OrderBy(sortProperty);

           // 6. Trả kết quả về Views

           return View(links.ToList());

       }

Bạn chú ý phần 2.1, 2.3 và phần 5 chúng ta thêm 1 trường hợp sort như trên, tức là thêm  khi hiển thị.

Sau đó Build dự án, chạy đường dẫn **http://localhost:xxxx/Link/** để xem kết quả cuối cùng.



# Kết luận

Thông qua 1 số cách sắp xếp nâng cao ở phần này, hi vọng bạn nắm bắt và làm được như chỉ dẫn. Lưu ý phần này kiến thức khá nâng cao và bạn rất hiếm thấy ở các trang lập trình ASP.NET [MVC](https://www.dammio.com/glossary/mvc) khác. Vì vậy, vui lòng trích dẫn nguồn [**DAMMIO.COM**](https://www.dammio.com/) nếu bạn tái sử dụng nội dung. Xin mời bạn theo dõi phần tiếp theo đang được viết.

[1]. [DammioMVC1\_phan13.rar](https://www.dammio.com/wp-content/uploads/2018/10/DammioMVC1_phan13.rar)