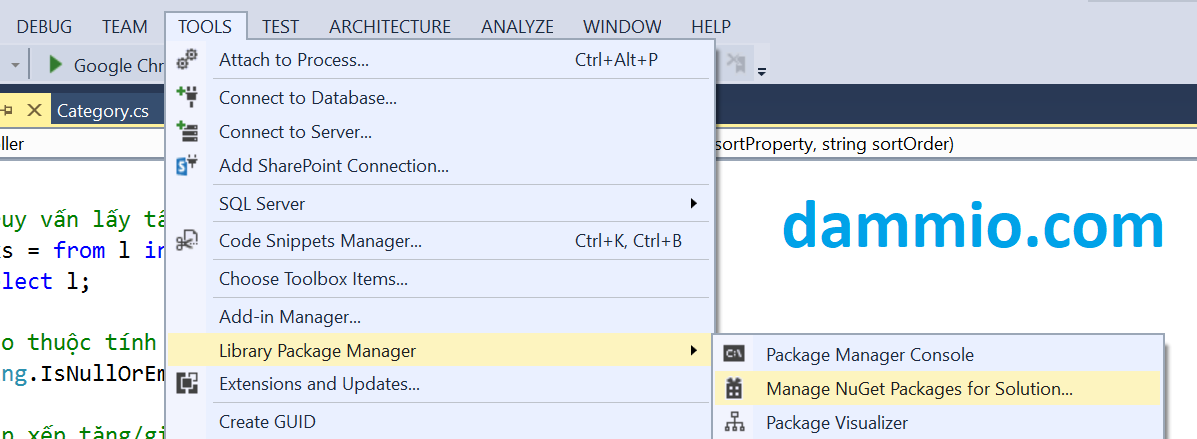
Phần 15: Phân trang trong ASP.NET MVC

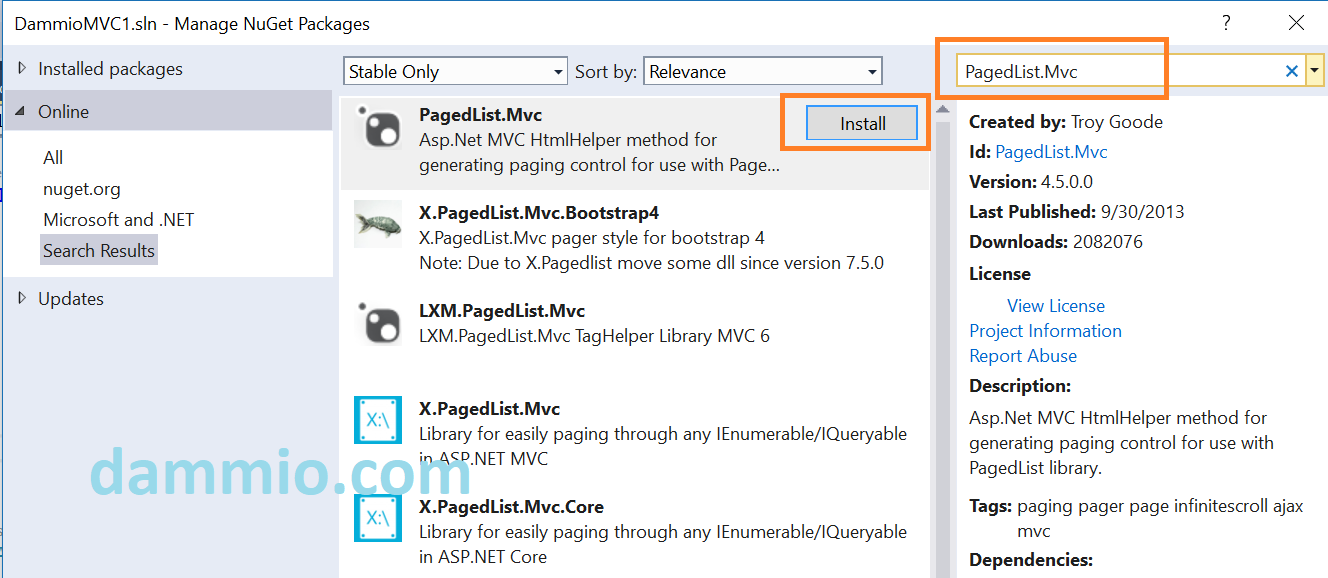
Trong phần này, chúng ta sẽ học cách phân trang trong ASP.NET [MVC](https://www.dammio.com/glossary/mvc) và cũng tái sử dụng dự án ở phần 13. Để thực hiện phân trang, trước hết bạn phải cài đặt thêm gói **[PagedList.Mvc](https://www.nuget.org/packages/PagedList.Mvc/" \t "_blank)** ở NuGet.

# Cài đặt gói PagedList.Mvc

Ở Visual Studio, bạn chọn mục Tools -> Library Package Manager -> Manage NuGet Packages… như hình sau.



Sau đó gõ **PagedList.**[**Mvc**](https://www.dammio.com/glossary/mvc) bên ô Search góc trái trên cùng, chọn gói PagedList.[Mvc](https://www.dammio.com/glossary/mvc) để cài đặt, nhấn nút Install.



# Phân trang

Sau khi cài đặt xong gói phân trang PagedList.[Mvc](https://www.dammio.com/glossary/mvc), bạn sẽ sử dụng gói này để phân trang.

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

Bạn mở tập tin Controller/LinkController.cs, nhúng thêm thư viện using PagedList; ở đầu trang, sau đó ở phương thức Index sửa như sau.

... nội dung code ...

using PagedList; // thêm thư viện này

namespace DammioMVC1.Controllers

{

    public class LinkController : Controller

    {

        private DammioEntities db = new DammioEntities();

        // GET: /Link/

        public ActionResult Index(int? page)

        {

            // 1. Tham số int? dùng để thể hiện null và kiểu int

            // page có thể có giá trị là null và kiểu int.

            // 2. Nếu page = null thì đặt lại là 1.

            if (page == null) page = 1;

            // 3. Tạo truy vấn, lưu ý phải sắp xếp theo trường nào đó, ví dụ OrderBy

            // theo LinkID mới có thể phân trang.

            var links = (from l in db.Links

                  select l).OrderBy(x=>x.LinkID);

            // 4. Tạo kích thước trang (pageSize) hay là số Link hiển thị trên 1 trang

            int pageSize = 3;

            // 4.1 Toán tử ?? trong C# mô tả nếu page khác null thì lấy giá trị page, còn

            // nếu page = null thì lấy giá trị 1 cho biến pageNumber.

            int pageNumber = (page ?? 1);

            // 5. Trả về các Link được phân trang theo kích thước và số trang.

            return View(links.ToPagedList(pageNumber, pageSize));

        }

... nội dung code ...

Trong đoạn mã trên, bạn có thể chú ý chúng ta thêm 1 số tham số **page** với kiểu **int?**. Kiểu int? mô tả 1 giá trị có thể là 1 số kiểu nguyên (int) kèm theo giá trị null. Như vậy **tham số page** có thể là null hoặc bất kỳ giá trị kiểu int nào.

Tại sao lại dùng như vậy? Đơn giản là vì đường dẫn **http://localhost:xxxx/Link** có lúc kèm theo tham số truy vấn page như thế này [**http://localhost:45033/Link?page=1**](http://localhost:45033/Link?page=1)

hoặc có thể là

**http://localhost:45033/Link?page=abc** hoặc không mô tả gì cả như link đầu tiên. Như vậy cách sử dụng int? rất tinh tế, giúp ép biến page vào 2 kiểu: một là int, hai là null (dành cho trường hợp tham số truy vấn là chuỗi). **Điều này loại luôn trường hợp bị hack SQL Injection trong ASP.NET**[**MVC**](https://www.dammio.com/glossary/mvc)**.**

Tiếp theo, nếu page là null, chúng ta đặt mặc định là 1 ở mục 2. Ở mục 3, chúng ta tạo câu truy vấn, lưu bạn **phải sort theo 1 trường nào đó bất kỳ** (ví dụ là LinkID) vì thư viện phân trang yêu cầu như vậy. Mục 4 mô tả kích thước trang hay số lượng record (Link) hiển thị trên 1 trang.

Mục 5 mô tả số trang hiện tại. Mục này chúng ta thấy sự xuất hiện của **toán tử ??** và đây là 1 toán tử tắt của trường hợp điều kiện so sánh null. Toán tử này mô tả nếu vế trái so sánh là null thì lấy giá trị vế phải, ngược lại lấy giá trị vế trái. Để hiểu rõ hơn bạn có thể xét ví dụ sau.

Ví dụ nếu x = null thì y sẽ lấy kết quả vế trái của toán tử ??, tức là 1. Ngược lại, kết quả b là 5 do a khác null.

int? x = null;

int y = x ?? 1; // Kết quả y là 1, do x = null.

int? a = 5;

int b = a ?? 1; // Kết quả b là 5 do a không phải là null.

Cuối cùng, mục 5 mô tả kết quả trả về bằng phương thức ToPagedList() với tham số pageNumber và pageSize.

## Sửa tập tin Views/Link/Index.cshtml

Bạn mở tập tin này và sửa nội dung như sau. Bạn có thể copy, paste vào code cho nhanh.

<!-- Thêm thư viện -->

@model PagedList.IPagedList<DammioMVC1.Models.Link>

@using PagedList.Mvc;

<!-- Kết thúc-->

@{

ViewBag.Title = "Index";

}

<h2>Index</h2>

<p>

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

</p>

<table class="table">

<tr>

<th>

LinkID

</th>

<th>

LinkName

</th>

<th>

LinkDescription

</th>

<th>

LinkURL

</th>

<th>

CategoryName

</th>

</tr>

@foreach (var item in Model) {

<tr>

<td>

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

</td>

<td>

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

</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>

<!-- Thêm mã phân trang -->

<br />

Trang @(Model.PageCount < Model.PageNumber ? 0 : Model.PageNumber) / @Model.PageCount

@Html.PagedListPager(Model, page => Url.Action("Index",

new { page }))

<!-- Kết thúc -->

Ở đoạn mã trên, chú phần đầu chúng ta thay đổi mô hình để sử dụng thư viện phân trang, đó là.

@model PagedList.IPagedList<DammioMVC1.Models.Link>

@using PagedList.Mvc;

Ở phần cuối, chúng ta thêm đoạn mã để hiển thị số trang và số trang hiện tại đó là.

<br />

Trang @(Model.PageCount < Model.PageNumber ? 0 : Model.PageNumber) / @Model.PageCount

@Html.PagedListPager(Model, page => Url.Action("Index",

    new { page }))

Ở control PagedListPager, chúng ta mô tả mô hình với số trang là liên kết với tham số Controller (Index) và tham số trang.

# Kiểm thử trình duyệt

Sau khi sửa mã nguồn như trên, bạn build dự án, chạy link **http://localhost:xxxx/Link** và bấm thử số trang để xem kết quả.



# Kết luận

Bài viết đã hướng dẫn bạn cách phân trang qua một ví dụ đơn giản với hướng dẫn chi tiết. Mời bạn tiếp tục theo dõi bài tiếp theo để học thêm nhiều điều thú vị từ ASP.NET [MVC](https://www.dammio.com/glossary/mvc).

# References

[1]. [DammioMVC1\_phan15.rar](https://www.dammio.com/wp-content/uploads/2018/10/DammioMVC1_phan15.rar)