ALLAN THRAEN | 17 years ago | PDF |

LINQ EXTENSIONS FOR EPISERVER

The topic of this post is something that in a short time has climbed very high on my personal wish-list for EPiServer. By now Christmas has both come and gone, without Santa dropping off any kind of ELINQ (nice name I thought off, eh) under my christmas tree.

However, that hasn't really stopped me in starting to use some of the cool functionality in LINQ in my various EPiServer projects, as you'll hopefully see in a few posts in the near future.

One thing I did to get started was to make a class with a few, but vital extension methods for the PageData object in order to make it easier to perform LINQ queries against the page data:

```
    public static class Extensions
    {

        public static IEnumerable<PageData> Descendants(this PageData self)
        {
            foreach (PageData pd in EPiServer.DataFactory.Instance.GetChildren(self.PageLink))
            {
                yield return pd;
                foreach (PageData p2 in pd.Descendants()) yield return p2;
            }
        }

        public static IEnumerable<PageData> Children(this PageData self)
        {
            foreach (PageData pd in EPiServer.DataFactory.Instance.GetChildren(self.PageLink))
            {
                yield return pd;
            }
        }

    }
```

These two IENumerable's will make it possible to do Queries like this:

```
var PageAuthors = from p in page.Descendants()
                 where p.VisibleInMenu &&
                 (p.Status == VersionStatus.Published)
                 group p by p.CreatedBy into g
                 select new {
                     Profile = EPiServerProfile.Get(g.Key),
                     Count = g.Count() };

var TopAuthors = from a in PageAuthors
                 orderby a.Count descending
                 select a;

var Top10Authors = TopAuthors.Take(10);
```

This will for instance allow us to easily get the Top 10 authors from all descending pages to a given "page" - which is a PageData object.
Due to the deferred execution of LINQ expressions and the use of the IEnumerable instead of List, the GetChildren calls will not be made until they are needed - and only if they are really needed.

I imagine that the Extensions class can be expanded with a lot more LINQ helper functions to access all the varies kinds of data in EPiServer in an easier manner. Feel free to suggest more features in the comments here (who knows - this might even in time become an EPiCode project).

Alas, cool as it looks it's still not an ideal solution since it's based solely on LINQ and not fully integrated against the entities / SQL in the back end. Put in another way, performance will probably stink if you use this on life-size sites since all the PageData objects it's working on needs to be loaded into memory.