

Implementing Site Search with Umbraco Examine

Alex Lindgren

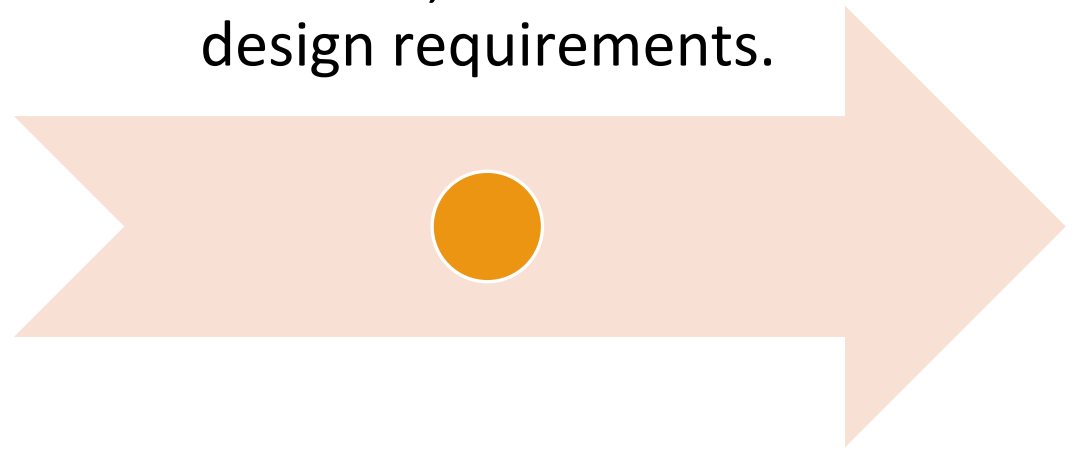
alindgren@flightpath.com

twitter.com/alexlindgren



Why we love Umbraco

We find that it is often the fastest way we know to develop great sites that meet the functional, technical and design requirements.



Why Search?

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minima veniam, quis nostrum exercitationem ullam corporis suscipit laboriosam, Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi Site visitors are in a hurry... ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu nulla pariatur. Ut enim minima veniam nisi ut aliquid, Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit aniSed ut perspiciatis sit unde omnis totam The want to find natus voluptatem accusantium doloremque laudantium, totam rem aperiam, eaque ipsa relevant content quae ab illo inventore veritatis et quasi architecto beatae vitae dicta sunt explicabo. Nemo enim quickly. ipsam voluptatem quia voluptas sit aspernatur aut odit aut fugit, sed quia consequuntur magni dolores eos qui ratione voluptatem sequi nesciunt. quaerat voluptatem. Ut enim ad minima veniam, quis nostrum exercitationem ullam corporis suscipit laboriosam, nisi ut aliquid ex ea commodi consequatur? Quis autem vel eum iure reprehenderit qui in ea voluptate velit esse quam nihil molestiae consequatur. . Ut enim ad minima veniam, quis nostrum

Umbraco Search

One (of many) reasons to love Umbraco is that it comes with a great search and indexing service.

Umbraco provides a powerful API for building custom search for your site.

Umbraco search is built on top of the popular Lucene library.

Lucene

Lucene is an open source Java library for indexing and searching.

Lucene.NET is a .NET port of the Lucene library.

Lucene and Lucene.NET are projects of the Apache Foundation.

<http://lucene.apache.org>

<http://lucenenet.apache.org>

Lucene

Key Features

Full-text indexing

Relevancy searching

Sorting search results (by field value or relevance)

Boosting documents and fields

Stemming analysis

Synonyms

Lucene

How Lucene computes
relevance

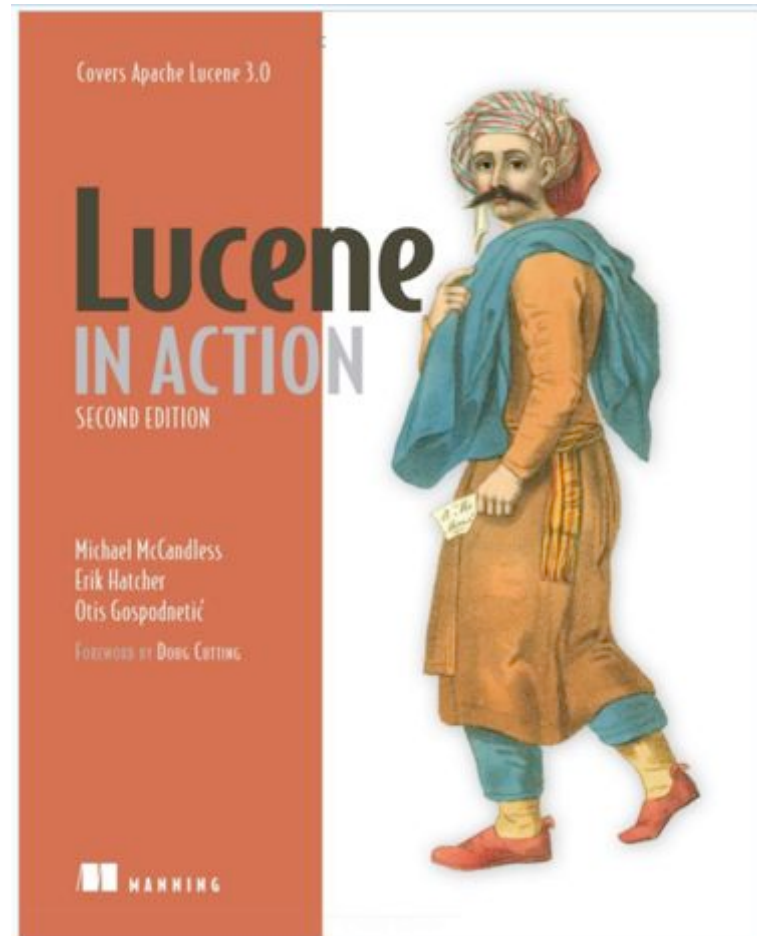
Lucene computes a score of how similar a document in the index matches the query.

$$\sum_{t \text{ in } q} (t \text{ f}(t \text{ in } d) \times \text{idf}(t^2) \times \text{boost}(t.\text{field in } d) \times \text{lengthNorm}(t.\text{field in } d)) \times \text{coord}(q,d) \times \text{queryNorm}(q)$$

Some Factors:

- How many times the term occurs in the document.
- How 'unique' the term is.
- Fields and documents can be boosted.
- Shorter fields get a bigger boost.

Lucene



Lucene in Action, 2nd edition
by Michael McCandless, Erik Hatcher and
Otis Gospodnetić (Manning 2010)

Luke

Lucene diagnostic tool

The screenshot shows the Luke Lucene diagnostic tool interface. At the top, there are menu options: File, Tools, Settings, Help. Below the menu is a toolbar with icons for Overview, Documents, Search, Files, and Plugins. The main content area displays index statistics for the index located at C:\Users\Alex\Documents\Projects\Mankoff\BobMankoff.Web\App_Data\TEMP\ExamineIndexes\Cartoons (Read-Only).

Index name: C:\Users\Alex\Documents\Projects\Mankoff\BobMankoff.Web\App_Data\TEMP\ExamineIndexes\Cartoons (Read-Only)
Number of fields: 18
Number of documents: 973
Number of terms: 17927
Has deletions? / Optimized?: Yes (4) / No
Last modified: Sat Mar 15 23:38:55 EDT 2014
Index version: 36c7d39d1c05
Index format: 0 (Lucene 2.9)
Index functionality: look-less, single norms, shared doc store, checksum, del count, omitTf, user data, diagnostics
TermInfos index divisor: N/A
Directory implementation: org.apache.lucene.store.SimpleFSDirectory
Currently opened commit point: segments_g (Sat Mar 15 23:38:55 EDT 2014)
Current commit user data: --

Select fields from the list below, and press button to view top terms in these fields. No selection means all fields.

Available fields and term counts per field:

Name	Term count	%	Decoder
__IndexType	1	0.01 %	string utf8
__NodeId	973	5.43 %	string utf8
__NodeType	1	0.01 %	string utf8
__Path	973	5.43 %	string utf8
__Sort_public	910	5.08 %	string utf8
caption	2,625	14.64 %	string utf8
categories	8	0.04 %	string utf8
featured	2	0.01 %	string utf8
id	973	5.43 %	string utf8
imageFileName	973	5.43 %	string utf8
keywords	6,722	37.56 %	string utf8

Top ranking terms. (Right-click for more options)

No	Rank	Field	Text
1	977	writerName	admin
2	977	nodeTypeAllis	cartoon
3	977	__NodeType	cartoon
4	977	__IndexType	content
5	977	parentID	11525
6	977	publicationDate	00
7	964	featured	0
8	458	keywords	artkey
9	397	keywords	rma
10	386	keywords	mankoff
11	370	keywords	man
12	363	keywords	robert
13	280	keywords	i

Number of top terms: 20
Hint: use Shift-Click to select ranges, or Ctrl-Click to select multiple fields (or unselect all).
Tokens marked in red indicate decoding errors, likely due to a mismatched decoder.

Select a field and set its value decoder: string utf8 Set

Index name: C:\Users\A...EMP\ExamineIndexes\Cartoons (R)

<http://code.google.com/p/luke/>

Umbraco Examine

Umbraco Examine is an API for searching that uses **Lucene.NET**.

Umbraco Examine is an implementation of an **Examine** provider.

Examine is a separate project that provides a **Fluent API** for using **Lucene.NET**.

<https://github.com/Shandem/Examine>

Note: having Examine and Lucene.NET built into Umbraco makes it much easier to manage compared to Solr or other search index services that are commonly used for search applications.

Use Cases

Site Search

Custom Data Search

Site Search

Challenge: Our sites that have complex pages built with blocks of content that are pulled from various nodes.

Custom Data Search

Our sites often have the requirement to search specific sets of data.

Examples: blog search, news and events search.

Cartoon Search Example

BobMankoff.com

Firefox

Cartoons | Bob Mankoff

www.bobmankoff.com/cartoons/39212fLjyY3MVRSM

Order The Book

BOB MANKOFF
CARTOON EDITOR OF THE NEW YORKER

BLOG **CARTOONS** APPEARANCES NEWS ABOUT

← BACK TO CARTOONS

Tweet

Reddit

Share

LICENSE THIS CARTOON

“One question: If this is the Information Age, how come nobody knows anything?”

GET LICENSE

“One question: If this is the Information Age, how come nobody knows anything?”



Filter By:

Best Match [Recently Added](#)

< 1 of 65 >



< 1 of 65 >

Cartoon Search Example

BobMankoff.com

Examine Settings

ExamineIndex.config

```
<?xml version="1.0"?>
<ExamineLuceneIndexSets>
  <!-- The internal index set used by Umbraco back-office - DO NOT REMOVE -->
  <IndexSet SetName="InternalIndexSet" IndexPath="~/App_Data/TEMP/ExamineIndexes/Internal/">

  <!-- The internal index set used by Umbraco back-office for indexing members - DO NOT REMOVE -->
  <IndexSet SetName="InternalMemberIndexSet" IndexPath="~/App_Data/TEMP/ExamineIndexes/InternalMember/">
    <IndexAttributeFields>
      <add Name="id" />
      <add Name="nodeName"/>
      <add Name="updateDate" />
      <add Name="writerName" />
      <add Name="loginName" />
      <add Name="email" />
      <add Name="nodeTypeAlias" />
    </IndexAttributeFields>
  </IndexSet>

  <!-- Default Indexset for external searches, this indexes all fields on all types of nodes-->
  <IndexSet SetName="ExternalIndexSet" IndexPath="~/App_Data/TEMP/ExamineIndexes/External/" />
</ExamineLuceneIndexSets>
```

Examine Settings

ExamineSettings.config

```
<?xml version="1.0"?>
<Examine>
  <ExamineIndexProviders>

  ...

  </ExamineIndexProviders>

  <ExamineSearchProviders>

  ...

  </ExamineSearchProviders>
</Examine>
```

Examine Settings

ExamineSettings.config (Index Providers)

```
<ExamineIndexProviders>
  <providers>
    <add name="InternalIndexer" type="UmbracoExamine.UmbracoContentIndexer, UmbracoExamine"
      supportUnpublished="true"
      supportProtected="true"
      analyzer="Lucene.Net.Analysis.WhitespaceAnalyzer, Lucene.Net"/>

    <add name="InternalMemberIndexer" type="UmbracoExamine.UmbracoMemberIndexer, UmbracoExamine"
      supportUnpublished="true"
      supportProtected="true"
      analyzer="Lucene.Net.Analysis.Standard.StandardAnalyzer, Lucene.Net"/>

    <!-- default external indexer, which excludes protected and unpublished pages-->
    <add name="ExternalIndexer" type="UmbracoExamine.UmbracoContentIndexer, UmbracoExamine"/>

  </providers>
</ExamineIndexProviders>
```


Examine Settings

ExamineSettings.config (Search Providers)

```
<ExamineSearchProviders defaultProvider="ExternalSearcher">
  <providers>
    <add name="InternalSearcher" type="UmbracoExamine.UmbracoExamineSearcher, UmbracoExamine"
      analyzer="Lucene.Net.Analysis.WhitespaceAnalyzer, Lucene.Net"/>

    <add name="ExternalSearcher" type="UmbracoExamine.UmbracoExamineSearcher, UmbracoExamine" />

    <add name="InternalMemberSearcher" type="UmbracoExamine.UmbracoExamineSearcher, UmbracoExamine"
      analyzer="Lucene.Net.Analysis.Standard.StandardAnalyzer, Lucene.Net" enableLeadingWildcards="true"/>

  </providers>
</ExamineSearchProviders>
```

Examine Settings

Documentation

<http://our.umbraco.org/documentation/Reference/Searching/Examine/full-configuration>

Examine Settings

Analyzers

“An analyzer tokenizes text by performing any number of operations on it, which could include extracting words, discarding punctuation, removing common words, reducing words to a root form (stemming), or changing words into the basic form” (from chapter 4 of *Lucene In Action*, 2nd edition).

Common Analyzers:

- `Lucene.Net.Analysis.WhitespaceAnalyzer`
- `Lucene.Net.Analysis.Standard.StandardAnalyzer`

You must use the same analyzer for both indexing and searching.

Managing Indexes

The screenshot displays the Umbraco Developer interface. The browser address bar shows `http://localhost:5555/umbraco/?/developer`. The interface is divided into a left sidebar and a main content area.

Left Sidebar (Developer):

- DEVELOPER
- Data Types
- Macros
- Packages
- Relation Types
- Scripting Files
- XSLT Files
- Partial View Macro Files

Main Content Area:

- Developer
- Get Started | Examine Management
- Examine Management
- Indexers
 - InternalMemberIndexer
 - InternalIndexer
 - ExternalIndexer
- Index info & tools
 - Documents in Index 37
 - Fields in Index 37
 - Has deletions? / Optimized? false (0) / true
 - Rebuild index
 - Optimize Index
- Node types
 - System fields
 - User fields
 - Provider properties
- Searchers
 - InternalSearcher

Cartoon Search Example

Queries using Examine API

Cartoon Search Requirements

- Keyword search on both captions and keywords fields.
- Filter by category
- 'all' category returns all cartoons
- Sortable by relevance and publication date

Cartoon Search Example

ExamineIndex.config (IndexSet)

```
<IndexSet SetName="CartoonIndexSet" IndexPath="~/App_Data/TEMP/ExamineIndexes/Cartoons/">
  <IndexAttributeFields>
    <add Name="id" />
    <add Name="nodeName" />
    <add Name="updateDate" />
    <add Name="writerName" />
    <add Name="path" />
    <add Name="nodeTypeAlias" />
    <add Name="parentID" />
  </IndexAttributeFields>
  <IndexUserFields>
    <add Name="caption"/>
    <add Name="publicationDate" EnableSorting="true"/>
    <add Name="keywords"/>
    <add Name="categories"/>
    <add Name="featured"/>
  </IndexUserFields>
  <IncludeNodeTypes>
    <add Name="Cartoon"/>
  </IncludeNodeTypes>
  <ExcludeNodeTypes>
  </ExcludeNodeTypes>
</IndexSet>
```

Cartoon Search Example

ExamineSettings.config (Index Providers)

```
<add name="CartoonIndexer" type="UmbracoExamine.UmbracoContentIndexer, UmbracoExamine"  
    indexSet="CartoonIndexSet"  
    supportUnpublished="false"  
    supportProtected="false"  
    analyzer="Lucene.Net.Analysis.Standard.StandardAnalyzer, Lucene.Net" />
```

ExamineSettings.config (Search Providers)

```
<add name="CartoonSearcher" type="UmbracoExamine.UmbracoExamineSearcher, UmbracoExamine"  
    analyzer="Lucene.Net.Analysis.Standard.StandardAnalyzer, Lucene.Net" enableLeadingWildcards="true"/>
```

Note: For comma-separated fields (such as Checkbox List), use the StandardAnalyzer – not the WhitespaceAnalyzer.

Cartoon Search Example

```
var searcher = ExamineManager
                .Instance
                .SearchProviderCollection["CartoonSearcher"];

var searchCriteria = searcher.CreateSearchCriteria(
    UmbracoExamine.IndexTypes.Content
);

var query = searchCriteria.RawQuery("nodeTypeAlias:cartoon");
var results = searcher.Search(query);
```


Cartoon Search Example

```
var results = searcher.Search(query);
```

Note: results are of type `Examine.ISearchResults`

`results.Count()` to get number of results

For paging:

```
var searchResults = results
    .Skip(((pageNumber - 1) * pageSize))
    .Take(pageSize);
```

Lucene Query (for debugging):

```
string luceneQuery = searchCriteria.ToString();
```

Cartoon Search Example

```
if (category != "All")
{
    query = criteria.Field("categories", category).Compile();
}
query = criteria.Field("keywords", keywords).Compile();
query.OrderBy(new string[] { "publicationDate" });
```

Note: searchResults is of type `IEnumerable<SearchResult>`

```
foreach (var item in searchResults)
{
    string cats = item.Fields["categories"];
}
```

Cartoon Search Example

Custom Index Field for Keyword search

```
public class ExamineEventHandlers : ApplicationEventHandler
{
    protected override void ApplicationStarted(UmbracoApplicationBase app,
        ApplicationContext applicationContext)
    {
        ExamineManager.Instance.IndexProviderCollection["Cartoons"].GatheringNodeData
            += ExamineEvent_GatheringNodeData;
    }

    void ExamineEvent_GatheringNodeData(object sender, IndexingNodeDataEventArgs e)
    {
        e.Fields.Add("combined", e.Fields["keywords"] +
            " " + e.Fields["caption"]);
    }
}
```

TypedSearch()

Instead of using the Examine API directly, UmbracoHelper has a method that returns Umbraco typed items.

```
UmbracoHelper helper = new UmbracoHelper(UmbracoContext);  
  
IEnumerable<IPublishedContent> results  
    = helper.TypedSearch(query);
```

Site Search Example

WageWorks.com

The screenshot shows the WageWorks website homepage. At the top left is the WageWorks logo with the tagline "everyone benefits". To the right is a navigation menu with links for HELP, SALES, LOG IN/REGISTER, EMPLOYEES, EMPLOYERS, BROKERS, ABOUT, and BLOG. The main banner features a photograph of a woman carrying a child on her shoulders, with the text "You're Our Top Priority" and "How You Save Money Is Your Business - and Ours Too". Below the banner is a section titled "How Can We Help You?" with a sub-headline: "WageWorks® offers easy-to-use spending accounts that enable you to pay for essential healthcare, child and elder care, commuter, and wellness benefits with money deducted from your paycheck before you pay taxes on it." This section contains six tiles: "What Are WageWorks Benefits?", "Need Help?", "How Do I Sign Up?", "What Does Pre-Tax Mean?", "How Do I Save Money? By Reducing Your Taxes", and "How Do I Use WageWorks?". Below these tiles is a list of three steps: "Using your WageWorks account is as easy as 1-2-3:", "1 Estimate how much you plan to spend on healthcare, child or elder daycare, commuter, and other types of eligible expenses for the year.", "2 Pay for eligible expenses using several no-hassle payment reimbursement options.", and "3 Reduce your tax burden and take home more pay." The bottom section is titled "Follow WageWorks" with the text "Good news travels fast. We want to make sure you're the first to know about important benefit updates and new ways to save." Below this are three boxes: "Latest News", a Twitter icon, and "DID YOU KNOW?". At the very bottom is a dark grey footer with a subscription form: "Get the latest updates from WageWorks." followed by a dropdown menu for "I am an Employee", a text input for "Your Email", and a "Subscribe" button. Social media icons for Facebook and @WageWorks are also present.

Site Search Example

WageWorks.com

Challenging Requirements:

- Composite pages (composed of node components)
- Faceted Search results

The screenshot displays a search interface for WageWorks.com. On the left, a search bar contains the text "SEARCH" and a magnifying glass icon. Below it, a faceted search sidebar is visible, showing filters for Audience and Content Type. The Audience filter includes "Employers (27)", "Employees (5)", and "Brokers (1)". The Content Type filter includes "Article (33)", "News (12)", "FAQ (3)", "Support (2)", and "Blog (1)". Below these, a "Product" filter shows "HSA (2)".

The main search results area is titled "Search Results: You searched for: 'HSA'" and shows "52 RESULTS". The results are sorted by "BEST MATCH" and "TITLE". The first result is "WageWorks Health Savings Account (HSA) Services", followed by "WageWorks and HSA Bank Team Up to Take Consumer-driven Plans to the Next Level", "Health Savings Account (HSA)", "Health Savings Account", "HSA FAQs", "HSA Limits to Remain the Same for 2011", "FAQs Landing Page", "HSA Limits to Remain the Same for 2011", "Using an HSA for retirement savings", and "Using an HSA for retirement savings".

At the bottom right of the page, there is a pagination control showing "1 2 3 4 5 6".

Site Search Example

Custom Index Field for Composite pages search

```
public class ExamineEventHandlers : ApplicationEventHandler
{
    protected override void ApplicationStarted(UmbracoApplicationBase app,
        ApplicationContext applicationContext)
    {
        ExamineManager.Instance.IndexProviderCollection["SiteSearch"].GatheringNodeData
            += ExamineEvent_GatheringNodeData;
    }

    void ExamineEvent_GatheringNodeData(object sender, IndexingNodeDataEventArgs e)
    {
        // Create a StringBuilder to store indexed content
        // For each Document Type, add the doc types fields to be indexed
        // Get children nodes (page components) and add searchable fields
        // Set index field to StringBuilder string
    }
}
```

Site Search Example

Custom Index Field for Composite pages search

```
public class ExamineEventHandlers : ApplicationEventHandler
{
    protected override void ApplicationStarted(UmbracoApplicationBase app,
        ApplicationContext applicationContext)
    {
        ExamineManager.Instance.IndexProviderCollection["SiteSearch"].GatheringNodeData
            += ExamineEvent_GatheringNodeData;
    }

    void ExamineEvent_GatheringNodeData(object sender, IndexingNodeDataEventArgs e)
    {
        // A better way... ?
        // RenderTemplate()
        // Load page from web request
    }
}
```


Faceted Search

Faceted search gives the user immediate feedback so they can make good filtering decisions.

Narrow Your Results

Product Reviews (5,770)

Computer Components
(6,185)

Computer Systems
Accessories (9,098)

Laptops (5,770)

Paper forms and
envelopes (81)

**See all matching
products**

News, Photos & Videos
(21,367)

Past 7 days (21)

Past 30 days (82)

Past year (1,373)

Audience 

Employers (27)

Employees (9)

Brokers (1)

Content Type 

Article (33)

News (12)

FAQ (3)

Support (2)

Blog (1)

Product 

HSA (2)

Faceted Search

Bobo-Browse

Original (Java) project powers LinkedIn search

<http://code.google.com/p/bobo-browse/>

BoboBrowse.Net is a .Net port

<http://bobo.codeplex.com/>

Cogworks blog post on using BoboBrowse.Net for faceted search in Umbraco:

<http://thecogworks.co.uk/blog/posts/2013/january/examiness-hints-and-tips-from-the-trenches-part-6/>

There is experimental support for facets in the 'Facet' branch of Examine on Github.

ezSearch

ezSearch is the easiest way to add search to an Umbraco site.

Available from the Umbraco package repository.

<https://github.com/mattbrailsford/ezsearch>

Basic steps:

- Install the package
- Create a search results page
- Add the macro to the search results page
- Wire up the global search form

Works well for simple sites but only searches all the nodes.

Resources

- Umbraco Examine documentation
<http://our.umbraco.org/documentation/Reference/Searching/Examine/>
- Examine documentation
<https://github.com/Shandem/Examine/wiki>
- Examining Examine blog post
<http://umbraco.com/follow-us/blog-archive/2011/9/16/examining-examine.aspx>