

برای آشنایی مقدماتی با لوسین می‌توان به مقالات فارسی ذیل مراجعه کرد:

[راهنمای گام به گام Lucene در c#](#)

[کتابخانه جستجوی فارسی با Lucene.net](#)

[آشنایی با کتابخانه Lucene.NET - قسمت اول](#)

[آشنایی با کتابخانه Lucene.NET - قسمت دوم](#)

[معرفی کتابخانه‌ی مشهور و قدرتمند Lucene.net -- قسمت اول](#)

به صورت خلاصه اگر نیاز به جستجوی سریع و پیشرفته‌ای بر روی حجم عظیمی از اطلاعات دارید، روش متداول `select * from table where field like something` توصیه نمی‌شود. بسیار کند است؛ مصرف CPU بالایی دارد. از ایندکس استفاده نمی‌کند. راه حل توصیه شده جهت برخورد با این نوع مسایل استفاده از `full text search` است. نگارش کامل SQL Server حاوی یک موتور FTS [توکار هست](#). اگر از بانک اطلاعاتی خاصی استفاده می‌کنید که دارای موتور FTS نیست یا ... FTS مخصوص SQL Server به درد کار شما نمی‌خورد یا نیاز به سفارشی سازی دارد (مثلا امکان تعریف `stop words` فارسی (کلماتی مانند به، از، تا و امثال آن))، از موتور FTS جانبی دیگری به نام [لوسین](#) نیز می‌توان استفاده کرد.

در کنار این‌ها ابزاری برای آنالیز و کوئری گرفتن از فایل‌های ایندکس تهیه شده توسط لوسین نیز وجود دارد به نام [Luke](#). برای نمونه اگر بانک اطلاعاتی سایت جاری را با لوسین به نحو متداولی ایندکس کنیم، در صفحه اول این برنامه، `top ranking terms` به شکل زیر ظاهر می‌شود:

Top ranking terms. (Right-click for more options)			
No	Rank ▼	Field	Text
1	871	Body	br
2	870	Body	div
3	865	Body	style
4	864	Body	align
5	856	Body	dir
6	852	Body	right
7	835	Body	rtl
8	810	Body	در
9	800	Body	از
10	797	Body	و
11	785	Body	text
12	778	Body	src
13	776	Body	به
14	774	Body	http
15	771	Body	href
16	767	Body	img
17	743	Body	1
18	730	Body	class
19	726	Body	های
20	711	Body	post
21	708	Body	width
22	707	Body	را
23	706	Body	height

Show top terms >>

Number of top terms:

100

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.

در اینجا چون متون تهیه شده از نوع HTML هستند، تگ br در آن‌ها زیاد است و یا یک سری حروف و کلمات فارسی هم در صدر قرار دارند که بهتر است از لیست ایندکس حذف شوند. برای اینکار تنها کافی است یک hash table را به نحو زیر تعریف و به StandardAnalyzer لوسین ارسال کنیم:

```
var stopWords = new Hashtable();
stopWords.Add("br","br");
// ...
var analyzer = new StandardAnalyzer(Version.LUCENE_29, stopWords);
```

یا آقای عرب عامری برای حروف و کلمات فارسی که نباید ایندکس شوند، یک لیست نسبتاً جامع را [در اینجا](#) تهیه کرده‌اند. اینبار اگر stop words یاد شده را اعمال و مجدداً ایندکس‌ها را تهیه کنیم به خروجی بهتری خواهیم رسید. در کل حداقل از این لحاظ، لوسین نسبت به FTS توکار SQL Server مناسب‌تر به نظر می‌رسد.

نظرات خوانندگان

نویسنده: شهرز جعفری
تاریخ: ۰۸:۱۳۹۱/۰۴/۲۶

سلام

یه سری از این نمونه‌ها از کد زیر استفاده شده

```
////////// Begin the new section
QueryParser oParser = new QueryParser("Body", new StandardAnalyzer());
string sTitle = "", sWriterID = "", finalQuery = "";

sTitle = " AND (Title:" + titleTerm + ")";

sWriterID = " AND (WriterID:" + writerID + ")";

finalQuery = "(" + bodyTerm + sTitle + sWriterID + ")";
hits = searcher.Search(oParser.Parse(finalQuery));
////////// End of the new section
```

که به نظرم حرفه ای نیست.
در ضمن اگه میشه چند تا سایت برای استفاده از کتابخانه در Entity framework معرفی کنید.

نویسنده: وحید نصیری
تاریخ: ۰۴۷:۱۳۹۱/۰۴/۲۶

کاری به EF نداره. به شکل یک سیستم مستقل بهش نگاه کنید. رکوردها از Db دریافت و به شکل document به لوسین اضافه خواهند شد. در همین حین index هم تشکیل می‌شود.
کوئری‌های آن دقیقاً به همین شکلی هست که در بالا اومده و زبان آن [SQL نیست](#).
البته پروژه LINQ به آن هم وجود دارد: ([^](#))

نویسنده: رضا.ب
تاریخ: ۱۱:۵۴ ۱۳۹۱/۰۴/۲۷

برای full text search در زبان فارسی این 150 کلمه که آقا عرب‌عامری زحمت کشیدن اصلاً کفایت نمی‌کند. زبان فارسی و دیگه‌اش به این نحو اصلاً مناسب ایندکس کردن نیست و بیشتر نیاز به توسعه دارد.
شاید اگر بشه با ابزارهای توسعه‌ای که روی زبان فارسی کار شده - مثل ویراسباز [^](#) - خروجی بگیریم برای ایندکس کردن و در نتیجه تحلیل اونا به وسیله همین کتابخونه Luke یا Solr یا Nutch - که در وبلاگ آقای زبردست معرفی شده [^](#) - آسونتره.

نویسنده: وحید نصیری
تاریخ: ۱۳:۵ ۱۳۹۱/۰۴/۲۷

کار آقای عرب‌عامری فراتر است از 150 کلمه ذکر شده. آنالیز فتحه، کسره، ی فارسی و عربی و غیره هم در آن لحاظ شده.

نویسنده: رضا.ب
تاریخ: ۰:۰ ۱۳۹۱/۰۴/۲۸

جستجو که در سایت همکنون هست، از همین کتابخانه استفاده می‌کنه یا توسعه داده شده؟
یه سوال شاید بی‌ربط. در آنالیز نتایج شاخص‌های لوسین ضربی غیر از تعداد دفعات تکرار شاخص مد نظر هست؟

نویسنده: وحید نصیری
تاریخ: ۰:۲۶ ۱۳۹۱/۰۴/۲۸

- از کتابخانه اصلی lucene.net استفاده شده.
- توضیحات مفصلش رو می‌تونید [اینجا](#) مطالعه کنید.

نویسنده: مهدی پایروند
تاریخ: ۱۳:۳۶ ۱۳۹۱/۰۷/۱۰

مهندس پیشنهاد اجرای Luke در ویندوز 7 چی؟
این آدرس [لینک](#) کلی کامپوننت برای دانلود گذاشته

نویسنده: وحید نصیری
تاریخ: ۱۳:۳۹ ۱۳۹۱/۰۷/۱۰

- فایل‌های jar نیاز به موتور اجرایی جاوا (JRE) دارند.
- luke [نسخه دات نتی](#) هم دارد.

نویسنده: مهدی پایروند
تاریخ: ۱۵:۲۵ ۱۳۹۱/۰۷/۱۲

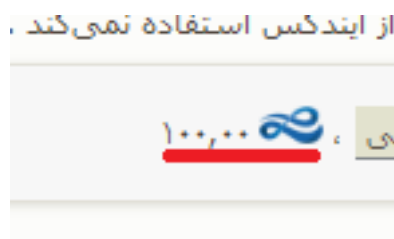
واقعا کتابخانه سریعی هستش برای کاری که من داشتم نزدیک به 16 هزار رکود 10 فیلدی رو توی لوسین ایندکس کرد که برای دریافت نتایجش زمان خیلی کمتری رو گرفت.

نویسنده: مهدی پایروند
تاریخ: ۱۵:۲۹ ۱۳۹۱/۰۷/۱۲

ممون برای من خیلی مورد استفاده بوده

نویسنده: مهدی پایروند
تاریخ: ۱۵:۳۷ ۱۳۹۱/۰۷/۱۲

در مورد این تصویر کمی توضیح میدین که به کدوم بخش لوسین مربوطه ممنون



نویسنده: وحید نصیری
تاریخ: ۱۷:۱۱ ۱۳۹۱/۰۷/۱۲

این عدد درصد نزدیک بودن جواب به جستجوی انجام شده است (رتبه جستجوی لوسین). فرمول محاسبه آن به صورت زیر است:

```
var hits = searcher.Search(query, 10).ScoreDocs;  
var scoreNorm = 100.0f / hits.GetMaxScore();  
foreach (var scoreDoc in hits)  
{  
    var resultScore = scoreNorm * scoreDoc.score;  
}
```

```
}
```

نویسنده: وحید نصیری
تاریخ: ۱۷:۲۶ ۱۳۹۱/۰۷/۱۲

بله. اینقدر باکیفیت است که RavenDB برای سیستم جستجوی خودش از لوسین به صورت توکار استفاده می‌کند.

نویسنده: مهدی پایروند
تاریخ: ۲۳:۵ ۱۳۹۱/۰۷/۱۳

آیا برای صفحه بندی معادلی برای skip و take وجود دارد، با توجه به جستجویی که من انجام دادم ([+](#) و [+](#) و [+](#) و [+](#) و [+](#) و [+](#)) نمونه‌ای دال بر وجود یک همچنین امکانی ندیدم.

نویسنده: وحید نصیری
تاریخ: ۲۳:۳۵ ۱۳۹۱/۰۷/۱۳

- سؤال شما مرتبط با بحث Luke نیست.
- و ... نداره. هر بار باید جستجو کنید. بعد روی نتیجه حاصل [صفحه بندی رو](#) انجام بدید. یا می‌تونید نتیجه جستجوی خاص انجام شده رو کش کنید تا سربار جستجوی مجدد حذف شود. (هرچند اینقدر سریع است که نیازی به کش نیست)

نویسنده: مهدی پایروند
تاریخ: ۱۹:۴۴ ۱۳۹۱/۰۷/۱۴

بابت پاسختون ممنونم

قسمت جستجوی سایت جاری رو با استفاده از لوسین بازنویسی کردم. خلاصه ای از نحوه انجام این کار رو در ادامه ملاحظه خواهید کرد:

1) دریافت کتابخانه های لازم

نیاز به کتابخانه های Lucene.NET و همچنین [Lucene.Net Contrib](http://Lucene.Net.Contrib) است که هر دو مورد را به سادگی توسط NuGet می توانید دریافت و نصب کنید. Highlighter استفاده شده، در کتابخانه Lucene.Net Contrib قرار دارد. به همین جهت این مورد را نیز باید جداگانه دریافت کرد.

2) تهیه منبع داده

در اینجا جهت سادگی کار فرض کنید که لیستی از مطالب را به فرمت زیر در اختیار داریم:

```
public class Post
{
    public int Id { set; get; }
    public string Title { set; get; }
    public string Body { set; get; }
}
```

تفاوتی نمی کند که از چه منبع داده ای استفاده می کنید. آیا قرار است یک سری فایل متنی ساده موجود در یک پوشه را ایندکس کنید یا تعدادی رکورد بانک اطلاعاتی؛ از NHibernate استفاده می کنید یا از Entity framework و یا از ADO.NET. کتابخانه Lucene مستقل است از منبع داده مورد استفاده و تنها اطلاعاتی با فرمت شیء Document معرفی شده به آن را می شناسد.

3) تبدیل اطلاعات به فرمت Lucene.NET

همانطور که عنوان شد نیاز است هر رکورد از اطلاعات خود را به شیء Document نگاشت کنیم. نمونه ای از اینکار را در متد ذیل مشاهده می نمائید:

```
static Document MapPostToDocument(Post post)
{
    var postDocument = new Document();
    postDocument.Add(new Field("Id", post.Id.ToString(), Field.Store.YES, Field.Index.NOT_ANALYZED));
    postDocument.Add(new Field("Title", post.Title, Field.Store.YES, Field.Index.ANALYZED,
    Field.TermVector.WITH_POSITIONS_OFFSETS));
    postDocument.Add(new Field("Body", post.Body, Field.Store.YES, Field.Index.ANALYZED,
    Field.TermVector.WITH_POSITIONS_OFFSETS));
    return postDocument;
}
```

این متد وهله ای از شیء Post را دریافت کرده و آنرا تبدیل به یک سند Lucene می کند. کار با ایجاد یک وهله از شیء Document شروع شده و سپس اطلاعات به صوت فیلدهایی به این سند اضافه می شوند.

توضیحات آرگومان های مختلف سازنده کلاس Field:

- در ابتدا نام فیلد مورد نظر ذکر می گردد.

- سپس مقدار متناظر با آن فیلد، به صورت رشته باید معرفی شود.

- آرگومان سوم آن مشخص می کند که اصل اطلاعات نیز علاوه بر ایندکس شدن باید در فایل های Lucene ذخیره شوند یا خیر. توسط Field.Store.YES مشخص می کنیم که بله؛ علاقمندیم تا اصل اطلاعات نیز از طریق Lucene قابل بازیابی باشند. این مورد جهت نمایش سریع نتایج جستجوها می تواند مفید باشد. اگر قرار نیست اطلاعاتی را از این فیلد خاص به کاربر نمایش دهید می توانید از گزینه Field.Store.NO استفاده کنید. همچنین امکان فشرده سازی اطلاعات ذخیره شده با انتخاب گزینه

Field.Store.COMPRESS نیز میسر است.

- توسط آرگومان چهارم آن تعیین خواهیم کرد که اطلاعات فیلد مورد نظر ایندکس شوند یا خیر. مقدار Field.Index.NOT_ANALYZED سبب عدم ایندکس شدن فیلد Id می‌شوند (چون قرار نیست روی id در قسمت جستجوی عمومی سایت، جستجویی صورت گیرد). به کمک مقدار Field.Index.ANALYZED، مقدار معرفی شده، ایندکس خواهد شد.

- پارامتر پنجم آن را جهت سرعت عمل در نمایان سازی/برجسته کردن و highlighting عبارات جستجو شده در متن‌های یافت شده معرفی کرده‌ایم. الگوریتم‌های متناظر با این روش در فایل‌های Lucene.Net Contrib قرار دارند.

یک نکته

اگر اطلاعاتی را که قرار است ایندکس کنید از نوع HTML می‌باشند، بهتر است تمام تگ‌های آن را پیش از افزودن به لوسین حذف کنید. به این ترتیب نتایج جستجوی دقیق‌تری را می‌توان شاهد بود. برای این منظور می‌توان از متد ذیل کمک گرفت:

```
public static string RemoveHtmlTags(string text)
{
    return string.IsNullOrEmpty(text) ? string.Empty : Regex.Replace(text, @"<(.|\n)*?>",
    string.Empty);
}
```

4 (تهیه Full text index به کمک Lucene.NET)

تا اینجا توانستیم اطلاعات خود را به فرمت اسناد لوسین تبدیل کنیم. اکنون ثبت و تبدیل آن‌ها به فایل‌های Full text search لوسین به سادگی زیر است:

```
static readonly Lucene.Net.Util.Version _version = Lucene.Net.Util.Version.LUCENE_29;
public static void CreateIdx(IEnumerable<Post> dataList)
{
    var directory = FSDirectory.Open(new DirectoryInfo(Environment.CurrentDirectory +
    "\\LuceneIndex"));
    var analyzer = new StandardAnalyzer(_version);
    using (var writer = new IndexWriter(directory, analyzer, create: true, mfl:
    IndexWriter.MaxFieldLength.UNLIMITED))
    {
        foreach (var post in dataList)
        {
            writer.AddDocument(MapPostToDocument(post));
        }

        writer.Optimize();
        writer.Commit();
        writer.Close();
        directory.Close();
    }
}
```

ابتدا محل ذخیره سازی فایل‌های Full text search مشخص می‌شوند. سپس آنالیز کننده اطلاعات باید معرفی شود. در ادامه به کمک این اطلاعات، شیء IndexWriter ایجاد و مستندات لوسین به آن اضافه می‌شوند. در آخر، این اطلاعات بهینه سازی شده و ثبت نهایی صورت خواهد گرفت.

ذکر version در اینجا ضروری است؛ از این جهت که اگر ایندکسی با فرمت مثلا LUCENE_29 تهیه شود ممکن است با نگارش بعدی این کتابخانه سازگار نباشد و در صورت ارتقاء، نتایج جستجوی انجام شده، کاملاً بی‌ربط نمایش داده شوند. با ذکر صریح نگارش، دیگر این اتفاق رخ نخواهد داد.

نکته

StandardAnalyzer توکار لوسین، امکان دریافت لیستی از واژه‌هایی که نباید ایندکس شوند را نیز دارا است. [اطلاعات بیشتر در اینجا](#).

5) به روز رسانی ایندکس‌ها

به کمک سه متد ذیل می‌توان اطلاعات ایندکس‌های موجود را به روز یا حذف کرد:

```
public static void UpdateIndex(Post post)
{
    var directory = FSDirectory.Open(new DirectoryInfo(Environment.CurrentDirectory +
"\\LuceneIndex"));
    var analyzer = new StandardAnalyzer(_version);
    using (var indexWriter = new IndexWriter(directory, analyzer, create: false, mfl:
IndexWriter.MaxFieldLength.UNLIMITED))
    {
        var newDoc = MapPostToDocument(post);

        indexWriter.UpdateDocument(new Term("Id", post.Id.ToString()), newDoc);
        indexWriter.Commit();
        indexWriter.Close();
        directory.Close();
    }
}

public static void DeleteIndex(Post post)
{
    var directory = FSDirectory.Open(new DirectoryInfo(Environment.CurrentDirectory +
"\\LuceneIndex"));
    var analyzer = new StandardAnalyzer(_version);
    using (var indexWriter = new IndexWriter(directory, analyzer, create: false, mfl:
IndexWriter.MaxFieldLength.UNLIMITED))
    {
        indexWriter.DeleteDocuments(new Term("Id", post.Id.ToString()));
        indexWriter.Commit();
        indexWriter.Close();
        directory.Close();
    }
}

public static void AddIndex(Post post)
{
    var directory = FSDirectory.Open(new DirectoryInfo(Environment.CurrentDirectory +
"\\LuceneIndex"));
    var analyzer = new StandardAnalyzer(_version, getStopWords());
    using (var indexWriter = new IndexWriter(directory, analyzer, create: false, mfl:
IndexWriter.MaxFieldLength.UNLIMITED))
    {
        var searchQuery = new TermQuery(new Term("Id", post.Id.ToString()));
        indexWriter.DeleteDocuments(searchQuery);

        var newDoc = MapPostToDocument(post);
        indexWriter.AddDocument(newDoc);
        indexWriter.Commit();
        indexWriter.Close();
        directory.Close();
    }
}
```

تنها نکته مهم این متدها، استفاده از متد IndexWriter با پارامتر create مساوی false است. به این ترتیب فایل‌های موجود بجای از نو ساخته شدن، به روز خواهند شد. محل فراخوانی این متدها هم می‌تواند در کنار متدهای به روز رسانی اطلاعات اصلی در بانک اطلاعاتی برنامه باشند. اگر رکوردی اضافه یا حذف شده، ایندکس متناظر نیز باید به روز شود.

6 جستجو در اطلاعات ایندکس شده و نمایش آن‌ها به همراه نمایان/برجسته سازی عبارات جستجو شده

قسمت نهایی کار با لوسین و اطلاعات ایندکس‌های تهیه شده، کوئری گرفتن از آن‌ها است. متدهای کامل مورد نیاز را در ذیل مشاهده می‌کنید:

```
public static void Query(string term)
{
    var directory = FSDirectory.Open(new DirectoryInfo(Environment.CurrentDirectory +
"\\LuceneIndex"));
    using (var searcher = new IndexSearcher(directory, readOnly: true))
    {
        var analyzer = new StandardAnalyzer(_version);
        var parser = new MultiFieldQueryParser(_version, new[] { "Body", "Title" }, analyzer);
        var query = parseQuery(term, parser);
        var hits = searcher.Search(query, 10).ScoreDocs;
```



```

if (hits.Length == 0)
{
    term = searchByPartialWords(term);
    query = parseQuery(term, parser);
    hits = searcher.Search(query, 10).ScoreDocs;
}

FastVectorHighlighter fvHighlighter = new FastVectorHighlighter(true, true);
foreach (var scoreDoc in hits)
{
    var doc = searcher.Doc(scoreDoc.doc);
    string bestfragment = fvHighlighter.GetBestFragment(
        fvHighlighter.GetFieldQuery(query),
        searcher.GetIndexReader(),
        docId: scoreDoc.doc,
        fieldName: "Body",
        fragCharSize: 400);
    var id = doc.Get("Id");
    var title = doc.Get("Title");
    var score = scoreDoc.score;
    Console.WriteLine(bestfragment);
}

searcher.Close();
directory.Close();
}
}

private static Query parseQuery(string searchQuery, QueryParser parser)
{
    Query query;
    try
    {
        query = parser.Parse(searchQuery.Trim());
    }
    catch (ParseException)
    {
        query = parser.Parse(QueryParser.Escape(searchQuery.Trim()));
    }
    return query;
}

private static string searchByPartialWords(string bodyTerm)
{
    bodyTerm = bodyTerm.Replace("*", "").Replace("?", "");
    var terms = bodyTerm.Trim().Replace("-", " ").Split(' ');
    terms = terms.Where(x => !string.IsNullOrEmpty(x))
        .Select(x => x.Trim() + "*");
    bodyTerm = string.Join(" ", terms);
    return bodyTerm;
}
}

```

توضیحات:

اکثر سایت‌ها را که بررسی کنید، جستجوی بر روی یک فیلد را توضیح داده‌اند. در اینجا نحوه جستجو بر روی چند فیلد را به کمک MultiFieldQueryParser مشاهده می‌کنید.

نکته‌ی مهمی را هم که در اینجا باید به آن دقت داشت، حساس بودن لوسین به کوچکی و بزرگی نام فیلدهای معرفی شده است و در صورت عدم رعایت این مساله، جستجوی شما نتیجه‌ای را دربر نخواهد داشت.

در ادامه برای parse اطلاعات، از متد کمکی parseQuery استفاده شده است. ممکن است به ParseException بخاطر یک سری حروف خاص بکارگرفته شده در عبارات مورد جستجو برسیم. در اینجا می‌توان توسط متد QueryParser.Escape، اطلاعات دریافتی را اصلاح کرد.

سپس نحوه استفاده از کوئری تهیه شده و متد Search را مشاهده می‌کنید. در اینجا بهتر است تعداد رکوردهای بازگشت داده شده را تعیین کرد (به کمک آرگومان دوم متد جستجو) تا بی‌جهت سرعت عملیات را پایین نیاورده و همچنین مصرف حافظه سیستم را نیز بالا نبریم.

ممکن است تعداد hits یا نتایج حاصل صفر باشد؛ بنابراین بد نیست خودمان دست به کار شده و به کمک متد searchByPartialWords، ورودی کاربر را بر اساس زبان جستجوی ویژه لوسین اندکی بهینه کنیم تا بتوان به نتایج بهتری دست یافت.

در آخر نحوه کار با ScoreDocs یافت شده را مشاهده می‌کنید. اگر محتوای فیلد را در حین ایندکس سازی ذخیره کرده باشیم، به کمک متد doc.Get می‌توان به اطلاعات کامل آن نیز دست یافت.

همچنین نکته دیگری را که در اینجا می‌توان ملاحظه کرد استفاده از FastVectorHighlighter می‌باشد. به کمک این Highlighter ویژه می‌توان نتایج جستجو را شبیه به نتایج نمایش داده شده توسط موتور جستجوی گوگل درآورد. برای مثال اگر شخصی ef code first را جستجو کرد، توسط متد GetBestFragment، بهترین جزئی که شامل بیشترین تعداد حروف جستجو شده است، یافت گردیده و همچنین به کمک تگ‌های B، ضخیم نمایش داده خواهند شد.

نظرات خوانندگان

نویسنده: Humid

تاریخ: ۱۳۹۱/۰۵/۰۵ ۱:۵۰

سلام و خسته نباشید. من از طریق سایت شما با این کتابخونه آشنا شدم. دارم باهاش کار می‌کنم اما یه مشکلی باهاش دارم. و اون اینه که قبلا توی سرچ من اگر کلمه "ما" رو سرچ می‌کردم 2600 تا نتیجه برمیگردوند اما الان 20 تا. چرا؟ و سوال بعد من اینه که چطور می‌تونم ایندکس کلمه پیدا شده توی متن رو پیدا کنم؟ چون نرم افزار خواسته شده رو می‌خوان مته نرم افزار نور باشه. ممنون میشم راهنماییم کنید. تشکر

نویسنده: وحید نصیری

تاریخ: ۱۳۹۱/۰۵/۰۵ ۸:۲۳

- احتمالا به عنوان [stopword](#) معرفی شده. این نوع کلمات ایندکس نخواهند شد. کلمه «ما» ارزش جستجو ندارد مانند «از»، «و»، «به»، «تا» و امثال آن.
- در مطلب فوق به قسمت ذیل دقت کنید. این Id همان Id واقعی یک رکورد در دیتابیس است که به عنوان یک سند لوسین ثبت شده:

```
var id = doc.Get("Id");
```

نویسنده: Humid

تاریخ: ۱۳۹۱/۰۵/۰۵ ۱۱:۱۷

ممنون آقای نصیری اما در زبان عربی "ما" یک کلمه تقریبا مهم هست. چطور می‌تونم اون رو از توی [stopword](#) خارج کنم. سوال دیگه اینکه من با سرچ قبلیم که اول به select می‌زدم با لایک و رکوردهایی که اون کلمه توش بود و پیدا می‌کردم و با استفاده از سرچ حرف به حرف در می‌آوردم کلمه رو. اما الان مثلا اگر "حسین" رو سرچ کنم با لوسین 10 نتیجه و با سرچ قبلی 31 نتیجه میده. چطور می‌تونم نزدیک کنم به نتیجه واقعی؟ البته در سرچ با لوسین از کاراکتر * استفاده هم کردم فرقی نکرد.

چطور می‌تونم در مبحث اعراب‌های کلمات عربی از لوسین استفاده کنم؟ آیا از زبان عربی و اعراب گذاری‌ها پشتیبانی می‌شه در این کتابخانه؟ ممنون

نویسنده: وحید نصیری

تاریخ: ۱۳۹۱/۰۵/۰۵ ۱۱:۳۴

- 10 نتیجه احتمالا به تنظیم زیر مرتبط است:

```
searcher.Search(query, 10)
```

در اینجا فقط 10 نتیجه بازگشت داده می‌شود (پارامتر دوم ذکر شده).

- در مورد اعراب زبان عربی به صورت پیش فرض خیر. اما اگر به کدهای فوق دقت کرده باشید از یک [StandardAnalyzer](#) توکار استفاده شده. این مورد یک سری تنظیمات ابتدایی را به همراه دارد. اگر کارکرد آن مورد قبول شما نیست می‌تونید خودتون یک [Analyzer](#) سفارشی رو توسعه بدید.
برای مثال یک نمونه سورس باز رو [اینجا می‌تونید](#) پیدا کنید که مباحث اعراب گذاری، ی و ک فارسی و عربی، یک سری [stopword](#)

فارسی و مسایل دیگر را هم لحاظ کرده.

نویسنده: عرفان
تاریخ: ۲۱:۵۸ ۱۳۹۱/۰۶/۲۶

سلام آقای نصیری،

دو تا سوال داشتم ازتون:

1- از این توابع مثلاً باید موقع درج مقاله (پست یا ..) در بانک اطلاعاتی برای ایندکس جدید استفاده کرد و موقع ویرایش و حذف مقاله (پست یا ..) هم از توابع معرفی شده متناسب استفاده کرد؟

2- جستجوی پیشرفته به چه صورت هستش؟ مثلاً تاریخ درج مقاله از ... تا ... ، نام نویسنده ، کلمه کلیدی و ... (که هر کدام از این موارد میتونه به صورت اختیاری و یا اجباری باشه).

نویسنده: وحید نصیری
تاریخ: ۲۲:۲ ۱۳۹۱/۰۶/۲۶

- بله. نیاز است مدام این ایندکس را به روز نگه داشت.

- برای این موارد متداول از تاریخ تا تاریخ، از همان SQL معمولی استفاده کنید. هر جایی که امکان تعریف ایندکس و کوئری های SQL ایی که از ایندکس استفاده می کنند، وجود دارد، روش های متداول SQL ایی بهینه ترین روش ها هستند. هدف در اینجا، full text search است بر روی انبوهی text. جستجوی بسیار سریع روی فیلدهای ایندکس نشده حجیم متنی با کیفیتی بالا. این هدف full text search است. چیزی مثل جستجوی گوگل.

در غیر اینصورت نیاز خواهید داشت از عبارات sql به همراه like استفاده کنید که ... بسیار کند هستند؛ چون باید کل جداول و بانک اطلاعاتی را هربار اسکن کنند و در حالت استفاده از like از ایندکس استفاده نمی شود.

نویسنده: عرفان
تاریخ: ۲۲:۴۱ ۱۳۹۱/۰۶/۲۶

-منظورتون از "روش های متداول SQL ایی بهینه ترین روش ها هستند" چیه؟

-پس در جستجوهای پیشرفته باید از روش معمول (like بدون استفاده از full text search) استفاده کرد و در جستجوهای تک کلمه ای مثل همین سایت باید از lucene (یا full text search) استفاده کرد، درسته؟

نویسنده: وحید نصیری
تاریخ: ۲۳:۹ ۱۳۹۱/۰۶/۲۶

- اگر کوئری SQL شما از ایندکس استفاده می کند نیازی به روش های full text search ندارید و موتورهای بانک اطلاعاتی به اندازه کافی برای مدیریت این نوع موارد سریع و بهینه هستند.

- جستجوی این سایت و یا full text search تک کلمه ای نیست. می تونید جمله هم بنویسید. کلاً برای بهبود سرعت، کاهش مصرف CPU و حافظه کوئری های SQL ایی که از like استفاده می کنند، روش full text search پیشنهاد می شود.

استفاده از like در عبارات SQL روش بهینه ای نیست چون هربار full table scan صورت می گیرد (تصور کنید 100 نفر در حال جستجوی مطالبی در سایت هستند. در این حالت مصرف CPU، استهلاک هارد و مصرف بالای حافظه را درحین اسکن کامل جداول بانک اطلاعاتی در نظر بگیرید)

نویسنده: عرفان
تاریخ: ۰:۰ ۱۳۹۱/۰۶/۲۷

"برای این موارد متداول از تاریخ تا تاریخ، از همان SQL معمولی استفاده کنید "

منظورتون اینه که تو جستجوهای پیشرفته باید از روش معمول و like(یعنی بدون استفاده از full text search و lucene)استفاده بشه؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۰۶/۲۷ ۰:۵

برای جستجوی تاریخ از like استفاده نمی‌شود. like معادل متد الحاقی Contains در LINQ to EF است + توضیح دادم چرا like مناسب نیست.

نویسنده: مهدی پایروند
تاریخ: ۱۳۹۱/۰۷/۱۲ ۱۵:۲۳

البته با تلفیقی از جستجوی لوسین و کوئری روی داده‌های رنج دار مثل **از تاریخ تا تاریخ** میتوان نتیجه ای سریعتری بدست آورد چون میتوان شناسه آجکت را در لوسین ذخیره کرد و در زمانی که کوئری دریافت شد بقیه کارها را انجام داد یا تاریخ را هم لوسین ذخیره کرد و روی آجکت‌های گرفته شده کار کرد یا که با توجه به شناسه‌های بدست آمده از بانک کوئری گرفت

نویسنده: مهدی پایروند
تاریخ: ۱۳۹۱/۰۹/۰۷ ۸:۴۵

البته من تست نکردم ولی شاید بتوان با توجه به این که میتوان تاریخ را بصورت عدد ذخیره کرد و نیز با استفاده از عبارات با قاعده تمام وظایف جستجو را به لوسین سپرد.

نویسنده: محسن عباس آبادعربی
تاریخ: ۱۳۹۱/۰۹/۲۰ ۱۷:۴۳

سلام؛

من تازه mvc رو شروع کردم در پروژه ای نیاز به استفاده از لوسین دارم خواهش میکنم یک نمونه برنامه که با استفاده از لوسین می‌باشد را در سایت قرار دهید با تشکر

نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۰۹/۲۰ ۱۷:۴۶

این دو برجسب را در سایت دنبال کنید:

[برجسب مطالب مرتبط با لوسین](#)

[برجسب اشتراک‌گذاری‌های مرتبط با لوسین](#)

نویسنده: حسین غلامی
تاریخ: ۱۳۹۱/۱۰/۱۱ ۰:۱۸

سلام؛

در قسمت FastVectorHighlighter متدی با نام () GetIndexReader شناخته شده نیست.

آیا از ورژن لوسین است؟

نویسنده: حسین غلامی
تاریخ: ۱۳۹۱/۱۰/۱۱ ۰:۲۲

در قسمت متد Query اگر ما خواسته باشیم اطلاعات رو در قالب همون شی (مثلا post) برگردونیم ، چطور باید این کار رو انجام داد؟
مثلا قسمت جستجوی همین سایت.

نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۱۰/۱۱ ۰:۲۵

بله. در نگارش جدید یک سری متدهای Get دار به خاصیت تبدیل شدن. مثلا متد GetIndexReader تبدیل شده به خاصیتی به نام IndexReader و مواردی از این دست.

نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۱۰/۱۱ ۰:۲۸

سورس قسمتی از جستجوی سایت [در دسترس است](#) . ولی در کل مقادیری رو که در ایندکس ذخیره کردید به صورت زیر قابل بازیابی است. پس از آن نگاشت نهایی را خودتان باید انجام دهید.

```
var prop = doc.Get("prop_name");
```

نویسنده: M.Q
تاریخ: ۱۳۹۲/۰۲/۲۱ ۱۶:۴۲

با سلام

من میخوام توی Lucene از عبارات منطقی and و or استفاده کنم ولی هرچی تست می‌کنم جواب دقیق و کاملی نمیگیرم (جستجو انجام میشه ولی مثلا از 3 آیتمی که حتما باید برگردونه 2 تاشو بر میگردونه).

آیا عبارات منطقی رو میشه در Lucene استفاده کرد؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۰۲/۲۱ ۱۷:۰۰

بله. بهترین مرجع برای این مسایل « [Apache Lucene - Query Parser Syntax](#) » است و همچنین [اینجا](#) برای مثال‌های بیشتر. به علاوه کلاس [BooleanQuery](#) هم برای اینکار وجود دارد.

نویسنده: آیمو
تاریخ: ۱۳۹۲/۰۸/۱۵ ۱۸:۱۵

سلام! من پروژه لوسین رو که شما ضمیمه کرده بودین توبخش استفاده از AutoComplete JQuery هم دانلود کردم و عین همونا رو پیاده کردم و همه چیز داره خوب کار میکنه. منتها شما اونجا چند تا post رو یک جا به لوسین دادین تا ایندکس کنه و لوسین هم برای همشون یه فایل میسازه . اما من که هر چند وقت یه بار تو سایت یه مطلبو ایندکس میکنم برای هر کدوم یه فایل ساخته و خب اگه تعداد مطلبام زیاد باشه این همینجور برای همشون تو دایرکتوری خودش فایل‌های یک کیلو بایتی میسازه . آیا این درسته؟ نمیدونم مشکل از کجاست! اگه میشه راهنمایی کنین....

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۰۸/۱۵ ۲۰:۴۱

[Too many open files](#)

نویسنده: آیمو

تاریخ: ۱۳۹۲/۰۸/۱۶ ۱۲:۱

ممنون به خاطر پاسختون . اون لینکو نگاه کردم و ایتم هایی که گفته بود رو من رعایت کرده بودم . اما همین طور فایل اضاف می کرد . بعد اومدم ببینم تا چند تا این همینجوری میخواد فایل بسازه ... یه برنامه ساده نوشتم و توش همون درخواست رو با httpClient بهش میدادم و و اونم می ساخت . جالب اینجاست که تا مثلا 30 تا ایتم فایل با پسوندها مختلف می ساخت بعد یه فایل Write Lock می ساخت . همشونو با هم یکی میکرد با یه اسم جدید و بعد باز شروع به ایندکس کردن میکرد . من با این برنامه چیزی حدود 10 هزار درخواست رو براش فرستادم و اونم ایندکس کرد مثله برق بدونه اینکه تعداد فایل ها از ماکزیمم 50 تا ایتم بیشتر بشه . من بی خود نگران بود . دستتون درد نکنه

نویسنده: جهش

تاریخ: ۱۳۹۳/۰۴/۰۳ ۱۰:۵۰

سلام

از تابع CreateIdx کجا باید استفاده کرد؟

نویسنده: وحید نصیری

تاریخ: ۱۳۹۳/۰۴/۰۳ ۱۱:۰

چون یک لیست را قبول می کند، نیاز است یکبار برای ایندکس کردن لیستی از اسناد موجود ایندکس نشده اجرا شود (در پزل مدیریتی برنامه مثلا). در سایر موارد (مانند افزوده شدن، به روز رسانی یا حذف یک رکورد) از مورد 5 استفاده کنید.

موتور لوسین علاوه بر فراهم آوردن امکان جستجوی سریع بر روی متون حجیم، [امکان یافتن مطالبی مشابه](#) یا مرتبط با مطلبی خاص را نیز فراهم می‌کند. نمونه آن را شاید در بعضی از انجمن‌ها یا وبلاگ‌ها دیده باشید که در ذیل مطلب جاری، چندین لینک را به مطالبی مشابه نیز نمایش می‌دهند. در ادامه نحوه استفاده از این قابلیت را در لوسین بررسی خواهیم کرد.

یافتن شماره سند متناظر لوسین

همان مثال «[استفاده از لوسین برای برجسته سازی عبارت جستجو شده در نتایج حاصل](#)» را در نظر بگیرید. در ابتدا نیاز است شماره یک مطلب را تبدیل به شماره سند لوسین کنیم. برای مثال ممکن است Id یک مطلب 1000 باشد، اما شماره سند متناظر آن در لوسین 800 ثبت شده باشد. بنابراین جستجوی ذیل الزامی است:

```
static readonly Lucene.Net.Util.Version _version = Lucene.Net.Util.Version.LUCENE_29;
static readonly IndexSearcher _searcher = new IndexSearcher(@"c:\path\idx", readOnly: true);
private static int GetLuceneDocumentNumber(int postId)
{
    var analyzer = new StandardAnalyzer(_version);
    var parser = new QueryParser(_version, "Id", analyzer);
    var query = parser.Parse(postId.ToString());
    var doc = _searcher.Search(query, 1);
    if (doc.totalHits == 0)
    {
        return 0;
    }
    return doc.scoreDocs[0].doc;
}
```

در اینجا بر اساس شماره یک مطلب، کوئری متناظر با آن تشکیل شده و جستجویی بر روی اسناد ثبت شده در ایندکس‌های لوسین صورت می‌گیرد. اگر اطلاعاتی یافت شد، شماره سند متناظر بازگشت داده می‌شود. از این جهت به شماره سند یاد شده نیاز داریم که قرار است مطالب مرتبط با کل این سند را بیابیم.

ساختن کوئری‌های MoreLikeThis

امکانات یافتن مطالب مشابه یک مطلب در اسمبلی Lucene.Net.Contrib.Queries.dll قرار دارد. بنابراین در اینجا نیاز به فایل‌های پروژه [Lucene.Net Contrib](#) وجود دارد. پس از یافتن شماره سند متناظر با یک مطلب، اکنون نوبت به ساخت کوئری‌های پیشرفته MoreLikeThis است که نحوه انجام تنظیمات آن را در ذیل مشاهده می‌کنید:

```
private static Query CreateMoreLikeThisQuery(int postId)
{
    var docNum = GetLuceneDocumentNumber(postId);
    if (docNum == 0)
        return null;

    var analyzer = new StandardAnalyzer(_version);
    var reader = _searcher.GetIndexReader();

    var moreLikeThis = new MoreLikeThis(reader);
    moreLikeThis.SetAnalyzer(analyzer);
    moreLikeThis.SetFieldNames(new[] { "Title", "Body" });
    moreLikeThis.SetMinDocFreq(1);
    moreLikeThis.SetMinTermFreq(1);
    moreLikeThis.SetBoost(true);

    return moreLikeThis.Like(docNum);
}
```


}

در اینجا فیلدهایی که قرار است در جستجو حضور داشته باشند توسط متد `SetFieldNames` معرفی می‌شوند. توسط متد `SetMinDocFreq` مشخص می‌کنیم که واژه‌های مشابه و مرتبط باید حداقل در چند سند ظاهر شده باشند. همچنین توسط متد `SetMinTermFreq` تعیین می‌گردد که یک واژه باید چندبار در این اسناد وجود داشته باشد. متد `SetBoost` سبب می‌شود که آنالیز بهتری بر اساس رتبه بندی‌های حاصل صورت گیرد.

نمایش مطالب مرتبط توسط کوئری `MoreLikeThis`

پس از این تنظیمات، متد `moreLikeThis.Like`، یک شیء `Query` را در اختیار ما قرار خواهد داد. از اینجای کار به بعد همانند سایر مطالب مشابه است. بر اساس این کوئری، جستجویی صورت گرفته و سپس اطلاعات یافت شده نمایش داده می‌شود:

```
private static void ShowMoreLikeThisPostItems(int postId)
{
    var query = CreateMoreLikeThisQuery(postId);
    if (query == null)
        return;

    var hits = _searcher.Search(query, n: 10);
    foreach (var item in hits.scoreDocs)
    {
        var doc = _searcher.Doc(item.doc);
        var id = doc.Get("Id");
        var title = doc.Get("Title");
        Console.WriteLine(title);
    }
}
```

نظرات خوانندگان

نویسنده: دل محسن
تاریخ: ۱۰:۴۸ ۱۳۹۳/۰۷/۰۸

سلام؛ میشه در Lucene جستجویی شبیه sql انجام داد؟ منظور اینکه همزمان item1 رو داخل field1 و item2 رو در field2 و.... جستجو کرد؟

نویسنده: وحید نصیری
تاریخ: ۱۰:۵۳ ۱۳۹۳/۰۷/۰۸

بله. زبان مخصوص خودش را دارد: [Query Parser Syntax](#) و [Lucene Query Syntax](#)
ضمناً بر همین مبنا LINQ to Lucene هم طراحی شده: [^](#) و [^](#)

نحوه اضافه کردن قابلیت غلط گیر املائی شبیه به جستجوی گوگل توسط لوسین

عنوان:

وحید نصیری

نویسنده:

۲۱:۱۰ ۱۳۹۱/۰۹/۰۶

تاریخ:

www.dotnettips.info

آدرس:

Lucene.NET

گروه‌ها:

پیشنیاز:

[چگونه با استفاده از لوسین مطالب را ایندکس کنیم؟](#)

مقدمه

اگر به جستجوی سایت دقت کرده باشید، قابلیتی تحت عنوان پیشنهاد «عبارات مشابه» به آن اضافه شده است:



این مورد بر اساس ماژول غلط یاب املائی لوسین تهیه شده و بسیار شبیه به "did you mean" جستجوی گوگل است. در ادامه به نحوه پیاده سازی آن خواهیم پرداخت.

کتابخانه‌های مورد نیاز

علاوه بر [کتابخانه لوسین](#)، نیاز به دریافت پروژه [Contrib](#) آن نیز می‌باشد تا بتوان از اسمبلی Lucene.Net.ContribSpellChecker.dll موجود در آن استفاده کرد.

نحوه کار با غلط یاب املائی لوسین

خلاصه کار با غلط یاب املائی لوسین همین چند سطر ذیل است:

```
var indexReader = IndexReader.Open(FSDirectory.Open(indexPath), readOnly: true);  
  
// Create the SpellChecker  
var spellChecker = new SpellChecker.Net.Search.Spell.SpellChecker(FSDirectory.Open(indexPath +  
"\\Spell"));  
  
// Create SpellChecker Index  
spellChecker.ClearIndex();  
spellChecker.IndexDictionary(new LuceneDictionary(indexReader, "Title"));  
spellChecker.IndexDictionary(new LuceneDictionary(indexReader, "Body"));
```

```
//Suggest Similar Words  
var results = spellChecker.SuggestSimilar(term, number, null, null, true);
```

کار بر اساس یک ایندکس از پیش موجود لوسین شروع می‌شود. در اینجا فرض شده است که این ایندکس در پوشه indexPath قرار دارد.

در ادامه شیء spellChecker را آغاز خواهیم کرد. بهتر است پوشه تولید فایل‌های آن با پوشه ایندکس اصلی یکسان نباشد. اگر یکسان در نظر گرفته شود، تمام مداخل جدید به ایندکس موجود اضافه خواهند شد که می‌تواند سرعت جستجوی معمولی را کاهش دهد.

سپس کار تهیه ایندکس جدید غلط یاب املایی، شروع خواهد شد. متد spellChecker.ClearIndex، اطلاعات موجود در ایندکسی قدیمی را حذف کرده و سپس spellChecker.IndexDictionary، فیلدهایی را که نیاز داریم در تهیه غلط یاب املایی حضور داشته باشند، مشخص می‌کند.

همانطور که ملاحظه می‌کنید ایندکس جدید تهیه شده، بر اساس بانک اطلاعاتی واژه‌های موجود در ایندکس اصلی برنامه که توسط indexReader معرفی شده، تهیه می‌شود. برای نمونه در تصویر ابتدای مطلب جاری، واژه‌های پیشنهادی، واژه‌هایی هستند که پیشتر یکبار تایپ شده و در بانک اطلاعاتی برنامه موجود بوده‌اند.

و در آخر برای استفاده از امکانات تهیه شده، تنها کافی است متد spellChecker.SuggestSimilar را فراخوانی کنیم (در زمانیکه جستجوی اصلی سایت نتیجه‌ای را ارائه نداده است). حاصل لیستی از واژه‌های مشابه است.

نظرات خوانندگان

نویسنده: علی رادمان فر
تاریخ: ۱۶:۵ ۱۳۹۱/۰۹/۰۷

Spell Checker فارسی هم با این امکانات وجود داره؟!

نویسنده: وحید نصیری
تاریخ: ۱۶:۲۹ ۱۳۹۱/۰۹/۰۷

با فارسی هم کار می‌کنه: (، و یا)

نویسنده: بافکر
تاریخ: ۱۷:۷ ۱۳۹۳/۰۲/۳۱

با سلام

حالتی هست که اینجا هم بشه اینسرت، دیلیت و آپدیت کرد؟ (همانند ایندکس اصلی)

نویسنده: محسن تقی پور
تاریخ: ۱۴:۱۳ ۱۳۹۳/۰۶/۱۳

با سلام

ببخشید یه سوال داشتم میخوام بدونم Create SpellChecker Index رو چه مواقعی باید فراخوانی کنم ؟
موقعی که خبر درج شد و ادیت شد و پاک شد باید فراخوانی بشه ؟ یا تو یه بازه زمانی خواص مثلا هر 24 ساعت ایجاد بشه ؟
میخوام بدونم ایا میشه این ایندکس ها رو آپدیت کرد یا اینکه هر دفعه تو ایندکس اصلیم تغییری ایجاد شد باید از اول ساخته شه ؟

نویسنده: وحید نصیری
تاریخ: ۱۴:۲۲ ۱۳۹۳/۰۶/۱۳

این مورد خاص SpellChecker قابلیت به روز رسانی ندارد. هر بار باید از صفر انجام شود. روزی یکبار کافی است.

پیشنیازها:

[چگونه با استفاده از لوسین مطالب را ایندکس کنیم؟](#)

[چگونه از افزونه jQuery Auto-Complete استفاده کنیم؟](#)

[نحوه استفاده صحیح از لوسین در ASP.NET](#)

اگر به جستجوی سایت دقت کرده باشید، قابلیت ارائه پیشنهاداتی به کاربر توسط یک Auto-Complete به آن اضافه شده‌است. در مطلب جاری به بررسی این مورد به همراه دو مثال Web forms و MVC پرداخته خواهد شد.



قسمت عمده مطلب جاری با پیشنیازهای یاد شده فوق یکی است. در اینجا فقط به ذکر تفاوت‌ها بسنده خواهد شد.

الف) دریافت لوسین

از طریق [NuGet](#) آخرین نگارش را دریافت و به پروژه خود اضافه کنید. همچنین Lucene.NET Contrib را نیز به همین نحو دریافت نمایید.

ب) ایجاد ایندکس

کدهای این قسمت با مطلب برجسته سازی قسمت‌های جستجو شده، یکی است:

```
using System.Collections.Generic;
using System.IO;
using Lucene.Net.Analysis.Standard;
using Lucene.Net.Documents;
using Lucene.Net.Index;
using Lucene.Net.Store;
using LuceneSearch.Core.Model;
using LuceneSearch.Core.Utills;

namespace LuceneSearch.Core
{
    public static class CreateIndex
    {
        static readonly Lucene.Net.Util.Version _version = Lucene.Net.Util.Version.LUCENE_30;

        public static Document MapPostToDocument(Post post)
        {
            var postDocument = new Document();
            postDocument.Add(new Field("Id", post.Id.ToString(), Field.Store.YES,
            Field.Index.NOT_ANALYZED));
            var titleField = new Field("Title", post.Title, Field.Store.YES, Field.Index.ANALYZED,
            Field.TermVector.WITH_POSITIONS_OFFSETS);
```

```

        titleField.Boost = 3;
        postDocument.Add(titleField);
        postDocument.Add(new Field("Body", post.Body.RemoveHtmlTags(), Field.Store.YES,
Field.Index.ANALYZED, Field.TermVector.WITH_POSITIONS_OFFSETS));
        return postDocument;
    }

    public static void CreateFullTextIndex(IEnumerable<Post> dataList, string path)
    {
        var directory = FSDirectory.Open(new DirectoryInfo(path));
        var analyzer = new StandardAnalyzer(_version);
        using (var writer = new IndexWriter(directory, analyzer, create: true, mfl:
IndexWriter.MaxFieldLength.UNLIMITED))
        {
            foreach (var post in dataList)
            {
                writer.AddDocument(MapPostToDocument(post));
            }

            writer.Optimize();
            writer.Commit();
            writer.Close();
            directory.Close();
        }
    }
}

```

تنها تفاوت آن اضافه شدن `titleField.Boost = 3` می‌باشد. توسط Boost به لوسین خواهیم گفت که اهمیت عبارات ذکر شده در عناوین مطالب، بیشتر است از اهمیت متون آن‌ها.

ج) تهیه قسمت منبع داده Auto-Complete

```

namespace LuceneSearch.Core.Model
{
    public class SearchResult
    {
        public int Id { set; get; }
        public string Title { set; get; }
    }
}

```

```

using System.Collections.Generic;
using System.IO;
using Lucene.Net.Index;
using Lucene.Net.Search;
using Lucene.Net.Store;
using LuceneSearch.Core.Model;
using LuceneSearch.Core.Utills;

namespace LuceneSearch.Core
{
    public static class AutoComplete
    {
        private static IndexSearcher _searcher;

        /// <summary>
        /// Get terms starting with the given prefix
        /// </summary>
        /// <param name="prefix"></param>
        /// <param name="maxItems"></param>
        /// <returns></returns>
        public static IList<SearchResult> GetTermsScored(string indexPath, string prefix, int maxItems
= 10)
        {
            if (_searcher == null)
                _searcher = new IndexSearcher(FSDirectory.Open(new DirectoryInfo(indexPath)), true);

            var resultsList = new List<SearchResult>();
            if (string.IsNullOrEmpty(prefix))
                return resultsList;

```

```

        prefix = prefix.ApplyCorrectYeKe();
        var results = _searcher.Search(new PrefixQuery(new Term("Title", prefix)), null, maxItems);
        if (results.TotalHits == 0)
        {
            results = _searcher.Search(new PrefixQuery(new Term("Body", prefix)), null, maxItems);
        }

        foreach (var doc in results.ScoreDocs)
        {
            resultsList.Add(new SearchResult
            {
                Title = _searcher.Doc(doc.Doc).Get("Title"),
                Id = int.Parse(_searcher.Doc(doc.Doc).Get("Id"))
            });
        }

        return resultsList;
    }
}

```

توضیحات:

برای نمایش Auto-Complete نیاز به منبع داده داریم که نحوه ایجاد آن را در کدهای فوق ملاحظه می‌کنید. در اینجا توسط جستجوی سریع لوسین و امکانات PrefixQuery آن، به تعدادی مشخص (maxItems)، رکوردهای یافت شده را بازگشت خواهیم داد. خروجی حاصل لیستی است از SearchResult ها شامل عنوان مطلب و Id آن. عنوان را به کاربر نمایش خواهیم داد؛ از Id برای هدایت او به مطلبی مشخص استفاده خواهیم کرد.

د) نمایش Auto-Complete در ASP.NET MVC

```

using System.Text;
using System.Web.Mvc;
using LuceneSearch.Core;
using System.Web;

namespace LuceneSearch.Controllers
{
    public class HomeController : Controller
    {
        static string _indexPath = HttpRuntime.AppDomainAppPath + @"App_Data\idx";

        public ActionResult Index(int? id)
        {
            if (id.HasValue)
            {
                //todo: do something
            }
            return View(); //Show the page
        }

        public virtual ActionResult ScoredTerms(string q)
        {
            if (string.IsNullOrEmpty(q))
                return Content(string.Empty);

            var result = new StringBuilder();
            var items = AutoComplete.GetTermsScored(_indexPath, q);
            foreach (var item in items)
            {
                var postUrl = this.Url.Action(actionName: "Index", controllerName: "Home", routeValues:
                new { id = item.Id }, protocol: "http");
                result.AppendLine(item.Title + "|" + postUrl);
            }

            return Content(result.ToString());
        }
    }
}

```



```

ViewBag.Title = "جستجو";
var scoredTermsUrl = Url.Action(actionName: "ScoredTerms", controllerName: "Home");
var bulletImage = Url.Content("~/Content/Images/bullet_shape.png");
}
<h2>
    جستجو</h2>

<div align="center">
    @Html.TextBox("term", "", htmlAttributes: new { dir = "ltr" })
    <br />
    را وارد نمائید lu جهت آزمایش
</div>

@section scripts
{
    <script type="text/javascript">
        EnableSearchAutocomplete('@scoredTermsUrl', '@bulletImage');
    </script>
}

```

```

function EnableSearchAutocomplete(url, img) {
    var formatItem = function (row) {
        if (!row) return "";
        return "<img src='" + img + "' /> " + row[0];
    }

    $(document).ready(function () {
        $("#term").autocomplete(url, {
            dir: 'rtl', minChars: 2, delay: 5,
            mustMatch: false, max: 20, autoFill: false,
            matchContains: false, scroll: false, width: 300,
            formatItem: formatItem
        }).result(function (evt, row, formatted) {
            if (!row) return;
            window.location = row[1];
        });
    });
}

```

توضیحات:

- ابتدا ارجاعاتی را به jQuery، افزونه Auto-Complete و اسکریپت سفارشی تهیه شده، در فایل layout پروژه تعریف خواهیم کرد.
- در اینجا سه قسمت را مشاهده می‌کنید: کدهای کنترلر، View متناظر و اسکریپتی که Auto-Complete را فعال خواهد ساخت.
- قسمت مهم کدهای کنترلر، دو سطر زیر هستند:

```

result.AppendLine(item.Title + "|" + postUrl);
return Content(result.ToString());

```

- مطابق نیاز افزونه انتخاب شده در مثال جاری، فرمت خروجی مدنظر باید شامل سطرهایی حاوی متن قابل نمایش به همراه یک Id (یا در اینجا یک آدرس مشخص) باشد. البته ذکر این Id اختیاری بوده و در اینجا جهت تکمیل بحث ارائه شده است.
- return Content هم سبب بازگشت این اطلاعات به افزونه خواهد شد.
- کدهای View متناظر بسیار ساده هستند. تنها نام TextBox تعریف شده مهم می‌باشد که در متد جاوا اسکریپتی EnableSearchAutocomplete استفاده شده است. به علاوه، نحوه مقدار دهی آدرس دسترسی به اکشن متد ScoredTerms نیز مهم می‌باشد.

- در متد EnableSearchAutocomplete نحوه فراخوانی افزونه autocomplete را ملاحظه می‌کنید.
- جهت آن، به راست به چپ تنظیم شده است. با 2 کاراکتر ورودی فعال خواهد شد با وقفه‌ای کوتاه. نیازی نیست تا انتخاب کاربر از لیست ظاهر شده حتما با عبارت جستجو شده صد در صد یکی باشد. حداکثر 20 آیتم در لیست ظاهر خواهند شد. اسکرول بار لیست را حذف کرده‌ایم. عرض آن به 300 تنظیم شده است و نحوه فرمت دهی نمایشی آن را نیز ملاحظه می‌کنید. برای این منظور از متد formatItem استفاده شده است. آرایه row در اینجا در برگیرنده اعضای Title و Id ارسالی به افزونه است. اندیس صفر آن به عنوان دریافتی اشاره می‌کند.
- همچنین نحوه نشان دادن عکس العمل به عنصر انتخابی را هم ملاحظه می‌کنید (در متد result مقدار دهی شده).
- window.location را به عنصر دوم آرایه row هدایت خواهیم کرد. این عنصر دوم مطابق کدهای اکشن متد تهیه شده، به آدرس یک صفحه اشاره می‌کند.

ه) نمایش Auto-Complete در ASP.NET WebForms

قسمت عمده مطالب فوق با وب فرم‌ها نیز یکی است. خصوصاً توضیحات مرتبط با متد EnableSearchAutocomplete ذکر شده.

```
<%@ Page Language="C#" AutoEventWireup="true" CodeBehind="Default.aspx.cs"
Inherits="LuceneSearch.WebForms.Default" %>

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-
transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
    <meta charset="utf-8" />
    <meta name="viewport" content="width=device-width" />
    <title>جستجو</title>
    <link href="Content/Site.css" rel="stylesheet" type="text/css" />
    <script src="Scripts/jquery-1.7.1.min.js" type="text/javascript"></script>
    <script src="Scripts/jquery.autocomplete.js" type="text/javascript"></script>
    <script src="Scripts/custom.js" type="text/javascript"></script>
</head>
<body dir="rtl">
    <h2>
        جستجو</h2>
    <form id="form1" runat="server">
    <div align="center">
        <asp:TextBox runat="server" dir="ltr" ID="term"></asp:TextBox>
        <br />
        را وارد نمائید lu جهت آزمایش
    </div>
    </form>
    <script type="text/javascript">
        EnableSearchAutocomplete('Search.ashx', 'Content/Images/bullet_shape.png');
    </script>
</body>
</html>
```

```
using System.Text;
using System.Web;
using LuceneSearch.Core;

namespace LuceneSearch.WebForms
{
    public class Search : IHttpHandler
    {
        static string _indexPath = HttpRuntime.AppDomainAppPath + @"App_Data\idx";

        public void ProcessRequest(HttpContext context)
        {
            string q = context.Request.QueryString["q"];
            if (string.IsNullOrEmpty(q))
            {
                context.Response.Write(string.Empty);
                context.Response.End();
            }

            var result = new StringBuilder();
            var items = AutoComplete.GetTermsScored(_indexPath, q);
            foreach (var item in items)
            {
                var postUrl = "Default.aspx?id=" + item.Id;
                result.AppendLine(item.Title + "|" + postUrl);
            }

            context.Response.ContentType = "text/plain";
            context.Response.Write(result.ToString());
            context.Response.End();
        }

        public bool IsReusable
        { get { return false; } }
    }
}
```

در اینجا بجای Controller از یک Generic handler استفاده شده است (Search.ashx).

```
result.AppendLine(item.Title + "|" + postUrl);  
context.Response.Write(result.ToString());
```

در آن، عنوان مطالب یافت شده به همراه یک آدرس مشخص، تهیه و در Response نوشته خواهند شد.

کدهای کامل مثال فوق را از اینجا می‌توانید دریافت کنید:

[LuceneSearch.zip](#)

همچنین باید دقت داشت که پروژه MVC آن از نوع MVC4 است (VS2010) و فرض براین می‌باشد که IIS Express 7.5 را نیز پیشتر نصب کرده‌اید.

کلمه عبور فایل: dotnettips91

نظرات خوانندگان

نویسنده:

محسن

تاریخ:

۱۳۹۱/۱۰/۰۷ ۱۲:۵۹

ممنون از این مطلب مفید. آیا با استفاده روش فوق میشه چند جدول رو با هم ایندکس کرد؟

نویسنده:

وحید نصیری

تاریخ:

۱۳۹۱/۱۰/۰۷ ۱۳:۳

بله. در اینجا مشکلی با ثبت اطلاعات از چندین جدول مختلف، در یک ایندکس وجود ندارد. برای مدیریت جستجوی بهتر روی آن‌ها یک روش این است که در متد MapPostToDocument، فیلد دیگری را به نام مثلا TableName اضافه کنید تا بشود در حین جستجو از آن استفاده کرد.

نویسنده:

محمد حبیبی

تاریخ:

۱۳۹۱/۱۰/۲۵ ۱۴:۴

من یک تکست باکس ایجاد کردم که از همین Autocomplete شما استفاده میکنه. منتها یک دکمه جستجو هم قرار دادم که با کلیک کردن روی اون دوباره از لوسین استفاده میکنه و چندین رکورد رو هم بازگشت میده. اما چون باید یک سری رشته که نتیجه‌ی لوسین هست رو برگردونم از return Content استفاده میکنم ولی نتایج رو در یک صفحه‌ی جدید و بدون Master Page میاره. نحوه‌ی نمایش نتایج اطلاعات لوسین در یک View که حاوی یک Master Page هم هست چطوره؟ مثلا نتایج زیر همون تکست باکس نمایش داده بشه. آیا باید نوع اکشن چیز دیگه ای باشه؟

نویسنده:

وحید نصیری

تاریخ:

۱۳۹۱/۱۰/۲۵ ۱۴:۲۱

نتیجه جستجوی لوسین مثلا در مطلب فوق لیستی از SearchResult ها است. همین لیست را در اکشن متد بازگشت دهید. سپس در View متناظر یک حلقه درست کرده و حاصل را به صورت دلخواهی فرمت کنید. علت استفاده از return Content در مثال بالا، نیاز افزونه جی کوئری استفاده شده به خروجی ساده متنی است. در حالت‌های دیگر از return View معمولی استفاده کنید.

نویسنده:

مرتضی

تاریخ:

۱۳۹۱/۱۱/۰۵ ۱۲:۴۵

سلام آقای نصیری ممنون از اینکه اطلاعاتتون رو در اختیار دیگران قرار میدید بنده از کدهای بالا تویه سایتم استفاده کردم و جستجوی سایت مثل ساعت داره کار میکنه فقط تویه ظاهرش به مشکل برخوردم بنده میخوام اون box رو که اطلاعات توش نمایش داده میشه از سمت راست تکس باکس تراز بشه مثل همون عکسی که بالا گذاشتید از بنده از سمت چپ تراز شده و یه سوال دیگه، اون عکس loading رو هنگام جستجو نمایش نمیده آدرس دهی اون هم درسته البته تکس باس بنده position:absolute هستش نمیدونم ماله اینه که نمایش نمیده یا مثلا میره یه قسمت دیدگی از صفحه. ممنون میشم راهنمایی کنید.

نویسنده:

وحید نصیری

تاریخ:

۱۳۹۱/۱۱/۰۵ ۱۳:۱

این موارد را باید با اصلاح اسکریپت یا css مربوط به auto-complete مدیریت کنید. مثلا محل قرارگیری منوی بازشونده به صورت زیر مقدار دهی شده. این را در فایل jquery.autocomplete.js یافته و اصلاح کنید:

```
left: offset.left - options.width + 125
```

یا تصویر loading در css به نحو زیر تعیین شده:

```
.ac_loading
{
background: white url('Images/indicator.gif') left center no-repeat;
}
```

نویسنده: امیر

تاریخ: ۱۷:۴۲ ۱۳۹۱/۱۱/۰۹

سلام؛ من به مشکلی دارم وقتی بعد از جستجو گزینه ای رو انتخاب می‌کنی صفحه refresh می‌شه و متن انتخاب شده پاک می‌شه میشه لطفاً منو راهنمایی کنید مرسی

نویسنده: وحید نصیری

تاریخ: ۱۸:۰۶ ۱۳۹۱/۱۱/۰۹

بله. به همین نحو طراحی شده. زمانیکه یک گزینه انتخاب می‌شود، سطر زیر، کاربر را به صفحه متناظر هدایت می‌کند:

```
window.location = row[1];
```

این سطر در سمت کلاینت، مساوی Response.Redirect سمت سرور است. می‌تونید اینجا متن انتخابی رو به صورت مثلاً یک کوئری استرینگ تعریف کنید و بعد در صفحه‌ای دیگر دریافت و نمایش بدید.

نویسنده: محسن.د

تاریخ: ۱:۰۵ ۱۳۹۱/۱۱/۱۷

بسیار عالی بود. تنها یک مشکل برای من در ارتباط با کدهای آموزشی که برای استفاده از لوسین در اینترنت پیدا کردم وجود داره و اون هم اینکه در اون‌ها کلاس و توابع به صورت استاتیک تعریف و در کنترلرها و یا کلاس‌های لایه سرویس فراخوانی شدن. این مسئله باعث اشکال در نوشتن آزمون واحد برای کنترلرها و یا متدها همیشه ؟

آیا در استاتیک معرفی کردن کلاس و توابع علت و مزیت خاصی وجود داره ؟

نویسنده: وحید نصیری

تاریخ: ۸:۵۵ ۱۳۹۱/۱۱/۱۷

- اگر از لایه سرویس استفاده می‌کنید که نهایتاً با یک سری اینترفیس در کنترلرها کار خواهید کرد.
- بله. توضیح دادم در مطلب « [نحوه استفاده صحیح از لوسین در ASP.NET](#) ».

نویسنده: میهمان

تاریخ: ۲۰:۱۵ ۱۳۹۱/۱۱/۲۱

با سلام

امکانش هست که به توضیح هم در مورد نحوه کار مستقیم با دیتابیس بدین؟
کدام قسمت‌ها باید تغییر کند و کدام قسمت‌ها باید حذف و اضافه شوند
با تشکر

نویسنده: وحید نصیری

تاریخ: ۲۰:۳۰ ۱۳۹۱/۱۱/۲۱

لطفاً اولین پیشنهاد عنوان شده را مطالعه کنید.

نویسنده: میهمان

تاریخ: ۱۳۹۱/۱۱/۲۲ ۰:۱۷

با سلام مجدد؛ تمامی پیشنیازهای این مقاله را مطالعه کردم و خط به خط کدهای پروژه ای را که زحمت کشیدید و تهیه کرده اید را بررسی کردم. شما در اولین پیشنیاز عنوان شده فرموده بودید " کتابخانه Lucene مستقل است از منبع داده مورد استفاده و تنها اطلاعاتی با فرمت شیء Document معرفی شده به آن را می شناسد. " و در پروژه، در ابتدا کل داده ها را به یک document تبدیل کردید و سپس آنها را به IndexWriter اضافه کردید. و سپس جستجوهای خود را روی این دیکشنری انجام می دادید. حالا سوال من این است: من در ابتدا باید تمامی اطلاعات را از طریق عنوان شده به یک فایل دیکشنری تبدیل کنم و هر مطلب جدید که اضافه میشود ، به این دیکشنری نیز اضافه شود ؟ اگر تعداد رکوردها بالای میلیون باشد؟ راهی وجود ندارد که ما مستقیم به خود دیتابیس کار کنیم ؟ و یا نه من ابتدا باید روی دیتابیس کلمه مورد نظر را جستجو کنم و پس از یافتن آن، آنرا به دیکشنری اضافه کنم ؟ بهترین راه چیست که برای داده های بالا جوابگو باشد با تشکر فراوان

نویسنده: وحید نصیری

تاریخ: ۱۳۹۱/۱۱/۲۲ ۰:۵۱

- لوسین مستقل است از بانک اطلاعاتی. همچنین یکبار باید این ایندکس را تهیه کنید. اگر تعداد رکوردهای شما بالا است، فقط همان بار اول است که کار تهیه زمانبر خواهد بود. برای دفعات بعد در حد اضافه کردن چند سند لوسین به آن یا به روز رسانی و حذف است و کار دیگری ندارد.

- پس از تهیه ایندکس، جستجوی لوسین کاری به بانک اطلاعاتی شما ندارد. بر روی ایندکس خودش انجام می شود و نیازی به جستجوی مجدد در بانک اطلاعاتی شما نیست. یک سیستم مستقل است.

این روش متداول کار با لوسین است و حالت دیگری هم ندارد. این مستقل بودن هم یک مزیت است. برای مثال SQL Server CE یا خیلی از بانک های اطلاعاتی دیگر Full Text Search توکار ندارند. اینجا لوسین خوب جواب می ده.

ضمن اینکه من در یک دمو استفاده از لوسین برای ایندکس کردن کل اطلاعات ویکی پدیا رو دیدم. تهیه ایندکس آن یک روز کار برده بوده (با توجه به حجم اطلاعات بالای ویکی پدیا)، اما جستجوی آن فوق العاده سریع و با کیفیت بود. این ویدیو رو در اینجا می تونید مشاهده کنید:

[Full-text search with Lucene and neat things you can do with it](#)

نویسنده: morteza

تاریخ: ۱۳۹۱/۱۲/۰۱ ۲۲:۳۰

سلام آقای نصیری میخواستم ببینم آیا شما اطلاعی از نحوه غیر فعال کردم اینتر تویه این پلاگین دارید بنده میخوام وقتی کاربر اینتر رو فشار داد جستجوی معمولی انجام بشه و اگه خواست از مواردی که توسط autocomplete براش آمده توسط موس یکی رو انتخاب کنه و به صفحه مربوطه هدایت بشه ممنون.

نویسنده: وحید نصیری

تاریخ: ۱۳۹۱/۱۲/۰۱ ۲۲:۴۹

کلید enter با case KEY.RETURN در این افزونه جی کوئری مدیریت می شود.

نویسنده: علیرضا پایدار

تاریخ: ۱۳۹۱/۱۲/۳۰ ۱۰:۵۵

یک سوال خدمت شما داشتم اینکه :

وقتی بخوایم از 2 جدول متفاوت جستجو کنیم به نظر شما اطلاعات هر جدول را جداگانه ایندکس کنیم(منظورم همان کاری که متد CreateFullTextIndex شما انجام میده) یا خیر؟ البته منظور دو جدولی که با هم رابطه دارند.

چندتا لینک در مورد لوسین:

<http://www.thebestcsharpprogrammerintheworld.com/blogs/how-to-create-and-search-a-lucene-net-index-in-4-simple-steps-using-c-sharp-step-1.aspx>

<http://www.codeproject.com/Articles/272309/Lucene-Search-Programming>

<http://www.codeproject.com/Articles/320219/Lucene-Net-ultra-fast-search-for-MVC-or-WebForms>

ممنون

نویسنده: وحید نصیری
تاریخ: ۱۱:۳۲ ۱۳۹۱/۱۲/۲۰

من همه رو داخل یک ایندکس ثبت می‌کنم. فقط یک فیلد اضافه‌تر به نام «نام جدول» مورد نظر نیاز هست تا بشود روی آن کوئری خاص گرفت یا اینکه کلا روی تمام رکوردها جستجو کرد به یکباره.

نویسنده: مهمان
تاریخ: ۷:۵۵ ۱۳۹۲/۰۳/۱۹

ضمن خسته نباشید
اگر نیاز داشته باشیم که بعد از انتخاب گزینه کاربر به صفحه دیگه هدایت نشه و فقط آیتم انتخابی در تکست باکس بمونه باید چه کاری انجام داد. مرسی

نویسنده: وحید نصیری
تاریخ: ۱۷:۱۳ ۱۳۹۲/۰۳/۱۹

سطر window.location رو حذف کنید.

نویسنده: imo0
تاریخ: ۱۰:۵۵ ۱۳۹۲/۰۷/۲۷

سلام؛ تویه اولین پیشنیازها که کار با لوسین رو گفته بودین توابعی مثله SearchByPartialWords و Query و غیره ایجاد شده ولی تو اینجا یه جور دیگه جستجو کردین توی ایندکس‌ها . میخواستم بدونم او توابع رو مگه واسه سرچ تو ایندکس‌ها ننوشتین؟ تازه اینجا شما دارین متن Title رو میفرستین سمت کاربر . این highlight کردن عبارت جستجو شدش پس چی شد؟

نویسنده: وحید نصیری
تاریخ: ۱۱:۳۵ ۱۳۹۲/۰۷/۲۷

- این هم یک روش دیگه هست. از آن روش‌های ذکر شده در پیشنیازها هم می‌توانید استفاده کنید.
- highlight کردن عبارت جستجو شده در عنوان بازگشت داده شده، توسط افزونه auto-complete انجام می‌شود؛ خودکار است.

نویسنده: imo0
تاریخ: ۱۱:۴۶ ۱۳۹۲/۰۷/۲۷

تو اون تابع Query که نوشتین یه دونه FastVectorHighlighter ایجاد کردین . اگه highlight توسط auto-complete خودکار انجام میشه پس نقش این چیه؟ میشه بگین کدوم روش بهتره ؟ یا فرقی نمیکنه؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۰۷/۲۷ ۱۱:۵۲

دو بحث وجود دارد:

- برجسته سازی قسمتی از عبارت جستجو شده در لیست نمایش داده شده توسط افزونه auto-complete. این مورد خودکار است و توسط افزونه انجام می شود.
- برجسته سازی قسمتی از عبارت جستجو شده در نتایج یک جستجوی کامل بعدی که قرار است highlight آن توسط ما با کدنویسی خاصی انجام شود. مراجعه کنید [به این مطلب](#) برای توضیحات بیشتر.

نویسنده: ایمان اسلامی
تاریخ: ۱۳۹۳/۰۲/۲۲ ۱۵:۵۶

با سلام

من از لوسین در asp.net web form استفاده کردم و کاملاً هم راضی هستم. فقط یه مشکلی داشتم. اگه بخوام غیر از عبارت مورد نظر برای جستجو یه مقدار دیگه رو هم بفرستم باید چکار کنم؟
مثلاً یه Dropdown داشته باشیم و بخواهیم مثلاً مقدار categoryid رو هم پاس بدیم تا به query مربوطه اضافه بشه. من dropdown رو اضافه کردم اما مقدارش رو همیشه در هندلر مربوط به Search دریافت کرد مثل q.

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۲/۲۲ ۱۷:۲۶

افزونه‌ی مورد استفاده در آن به نام jquery.autocomplete.js سورسش پیوست هست. ajax را در آن جستجو کنید به مورد ذیل خواهید رسید:

```
$.ajax({  
  //...  
  data: $.extend({  
    q: lastWord(term),  
    limit: options.max  
  }, extraParams),  
  //...  
});
```

در اینجا q مشخص است و extraParams آن متغیر. بنابراین برای ارسال اطلاعات اضافی باید extraParams را مانند کدهای ذیل مقدار دهی کرد. نحوه‌ی دریافت عناصر آن در سمت سرور، مانند نحوه‌ی دریافت q است.

```
$(document).ready(function () {  
  $("#term").autocomplete(url, {  
    // .....  
    extraParams: { id: 12, xyz: "test" }  
  }).result(function (evt, row, formatted) {  
    // .....  
  });  
});
```

نویسنده: ایمان اسلامی
تاریخ: ۱۳۹۳/۰۳/۰۷ ۱۸:۱۱

اگر بخوام لیستی که برای نتیجه جستجو باز میشه رو Custom کنم مثلاً یه عکس بهش اضافه کنم یا یه جدول ساخته شده با HTML رو توش نشون بدم با یه سری جزئیات، آیا امکان پذیره؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۳/۰۷ ۱۹:۳۰

نحوه سفارشی سازی هر آیتم لیست را در مطلب فوق، در متد formatItem می‌توانید مشاهده کنید.

نویسنده: علیرضا

تاریخ: ۱۰:۵۸ ۱۳۹۳/۰۴/۰۳

با سلام؛ تکست باکسی که می‌خام جستجو از طریق اون انجام بشه در یک مستر پیج قرار داره و زمانی که صفحه و مستر پیج در یک پوشه قرار دارند مشکلی نیست ولی اگر صفحه در یک پوشه دیگر قرار گیرد جواب نمی‌ده. توضیح اینکه از متد ResolveClientUrl هم استفاده کردم ولی جواب نداد :

```
EnableSearchAutocomplete('<%=ResolveClientUrl("~/Handlers/find.ashx")%>',  
'<%=ResolveClientUrl("Images/bullet_shape.png")%>');
```

نویسنده: وحید نصیری

تاریخ: ۱۱:۱۴ ۱۳۹۳/۰۴/۰۳

- [نحوه استفاده از افزونه Firebug برای دیباگ برنامه‌های ASP.NET مبتنی بر jQuery](#)
- بررسی کنید آیا ID جعبه متنی در آن صفحه، term است یا [چیز دیگری](#) ؟ اگر تغییر کرده، ClientIDMode=Static بهتر است تنظیم شود.

نویسنده: علیرضا

تاریخ: ۱۸:۳۱ ۱۳۹۳/۰۴/۰۳

با تشکر از پاسخ شما ، id درست بود و با فایرباگ هم بررسی کردم و خطای زیر رو می‌داد :

```
TypeError: $(...).autocomplete is not a function
```

نویسنده: وحید نصیری

تاریخ: ۱۸:۵۴ ۱۳۹۳/۰۴/۰۳

این خطا یعنی اسکریپت‌های اصلی به صفحه پیوست نشده‌اند. برای تعریف آن‌ها هم از ResolveClientUrl استفاده کنید:

```
<script type="text/javascript" src='<%= ResolveClientUrl("~/path...") %>'></script>
```

احتمالا یک سری از کارهای اینفوگرافیک مانند tags cloud و words cloud را دیده‌اید. برای مثال در یک سخنرانی خاص، سخنران بیشتر از چه واژه‌هایی استفاده کرده است و سپس ترسیم درشت‌تر واژه‌هایی با تکرار بیشتر در یک تصویر نهایی. محاسبات آماری این نوع بررسی‌ها را توسط لوسین نیز می‌توان انجام داد که در ادامه به نحوه انجام آن خواهیم پرداخت.

بررسی آماری واژه‌های بکار رفته در شاهنامه

مرحله اول: ایجاد ایندکس

```
using System;
using System.Collections.Generic;
using System.IO;
using Lucene.Net.Analysis.Standard;
using Lucene.Net.Documents;
using Lucene.Net.Index;
using Lucene.Net.Store;

namespace ShaahnamehAnalysis
{
    public static class CreateIndex
    {
        static readonly Lucene.Net.Util.Version _version = Lucene.Net.Util.Version.LUCENE_CURRENT;

        static HashSet<string> getStopWords()
        {
            var result = new HashSet<string>();
            var stopWords = new[]
            {
                "به",
                "با",
                "از",
                "تا",
                "و",
                "است",
                "هست",
                "هستم",
                "هستیم",
                "هستید",
                "هستند",
                "نیست",
                "نیستم",
                "نیستیم",
                "نیستید",
                "نیستند",
                "اما",
                "یا",
                "این",
                "آن",
                "اینجا",
                "آنجا",
                "بود",
                "یاد",
                "برای",
                "که",
                "دارم",
                "داری",
                "دارد",
                "داریم",
                "دارید",
                "دارند",
                "چند",
                "را",
                "ها",
                "های",
                "می",
                "هم",
                "در",
                "باشم",
            };
            result.AddRange(stopWords);
            return result;
        }
    }
}
```

,"باشی"
,"باشد"
,"باشیم"
,"باشید"
,"باشند"
,"اگر"
,"مگر"
,"بجز"
,"جز"
,"آلا"
,"اینکه"
,"چرا"
,"چگی"
,"چه"
,"چطور"
,"چی"
,"چیسٔ"
,"آیا"
,"چنین"
,"اینچنین"
,"نخست"
,"اول"
,"آخر"
,"انتهٔ"
,"صد"
,"هزار"
,"میلیون"
,"ملیون"
,"میلیارد"
,"مليارد"
,"یکهزار"
,"تریلیون"
,"تریلیارد"
,"میان"
,"بین"
,"زیر"
,"پیش"
,"روی"
,"ضمن"
,"همانا"
,"ای"
,"بعد"
,"پس"
,"قبل"
,"پیش"
,"هیچ"
,"همه"
,"واما"
,"شد"
,"شده"
,"شدم"
,"شدی"
,"شدیم"
,"شدند"
,"یکی"
,"یکی"
,"نبود"
,"میکند"
,"میکنم"
,"میکنیم"
,"میکنید"
,"میکنند"
,"میکنی"
,"طور"
,"اینطور"
,"آنطور"
,"هر"
,"حال"
,"مثل"
,"خواهم"
,"خواهی"
,"خواهد"
,"خواهیم"
,"خواهید"
,"خواهند"
,"داشته"
,"داشت"
,"داشتی"
,"داشتیم"
,"داشتید"

```

        "داشتند",
        "آنکه",
        "مورد",
        "کنید",
        "کنم",
        "کنی",
        "کنند",
        "کنیم",
        "نکنم",
        "نکنی",
        "نکنند",
        "نکنیم",
        "نکنید",
        "نکنند",
        "نکن",
        "نگو",
        "مگو",
        "بنابراین",
        "بدین",
        "من",
        "تو",
        "او",
        "ما",
        "شما",
        "ایشان",
        "ی",
        "-",
        "های",
        "خیلی",
        "بسیار",
        "1",
        "1",
        "1",
        "شود",
        "کرد",
        "کرده",
        "نیز",
        "خود",
        "شوند",
        "اند",
        "داد",
        "دهد",
        "گشت",
        "ز",
        "گفت",
        "آمد",
        "اندر",
        "چون",
        "بد",
        "چو",
        "همی",
        "پر",
        "سوی",
        "دو",
        "گر",
        "بی",
        "گرد",
        "زین",
        "کس",
        "زان",
        "جای",
        "آید"
    };

    foreach (var item in stopWords)
        result.Add(item);

    return result;
}

public static void CreateShaahnamehIndex(string file = "shaahnameh.txt")
{
    var directory = FSDirectory.Open(new DirectoryInfo(Environment.CurrentDirectory +
        "\\LuceneIndex"));
    var analyzer = new StandardAnalyzer(_version, getStopWords());
    using (var writer = new IndexWriter(directory, analyzer, create: true, mfl:
        IndexWriter.MaxFieldLength.UNLIMITED))
    {
        var section = string.Empty;
        foreach (var line in File.ReadAllLines(file))

```

```

        {
            int result;
            if (int.TryParse(line, out result))
            {
                var postDocument = new Document();
                postDocument.Add(new Field("Id", result.ToString(), Field.Store.YES,
Field.Index.NOT_ANALYZED));
                postDocument.Add(new Field("Body", section, Field.Store.YES,
Field.Index.ANALYZED, Field.TermVector.WITH_POSITIONS_OFFSETS));
                writer.AddDocument(postDocument);
                section = string.Empty;
            }
            else
                section += line;
        }

        writer.Optimize();
        writer.Commit();
        writer.Close();
        directory.Close();
    }
}
}
}

```

با ایجاد ایندکس‌های لوسین پیشتر در این سایت [آشنا شده‌اید](#). روش کار نیز همانند سابق است. اطلاعات خود را، به هر فرمتی که تهیه شده باید تبدیل به اشیاء Document لوسین کرد. برای مثال در اینجا فقط یک فایل txt داریم که تشکیل شده است از تمام صفحات. به ازای هر صفحه، یک شیء Document تهیه و نوشته خواهد شد. همچنین در تهیه ایندکس از یک سری از واژه‌های بسیار متداول مانند «از»، «به»، «اندر»، (stopWords) صرف‌نظر شده است.

مرحله دوم: ایجاد ابر واژه‌ها

```

using System;
using System.Collections.Generic;
using System.Diagnostics;
using System.Linq;
using Lucene.Net.Index;
using Lucene.Net.Store;

namespace ShaahnamehAnalysis
{
    [DebuggerDisplay("{Frequency}, {Text}")]
    public class Tag
    {
        public string Text { set; get; }

        /// <summary>
        /// The frequency of a term is defined as the number of
        /// documents in which a specific term appears.
        /// </summary>
        public int Frequency { set; get; }
    }

    public static class WordsCloud
    {
        /// <summary>
        /// Create Words Cloud
        /// </summary>
        /// <param name="threshold">every term that appears in more than x Body</param>
        public static IList<Tag> Create(int threshold = 200)
        {
            var path = Environment.CurrentDirectory + "\\LuceneIndex";

            var results = new List<Tag>();
            var field = "Body";

            IndexReader indexReader = IndexReader.Open(FSDirectory.Open(path), true);

            var termFrequency = indexReader.Terms();
            while (termFrequency.Next())
            {
                if (termFrequency.DocFreq() >= threshold && termFrequency.Term.Field == field)
                {

```

```

        results.Add(new Tag { Text = termFrequency.Term.Text, Frequency =
termFrequency.DocFreq() });
    }
    }
    return results.OrderByDescending(x => x.Frequency).ToList();
}
}
}

```

پس از اینکه ایندکس لوسین تهیه شد، می‌توان به مداخل موجود در آن توسط متد `indexReader.Terms` دسترسی یافت. نکته جالب آن فراهم بودن `DocFreq` هر واژه ایندکس شده است (فرکانس تکرار واژه؛ تعداد اشیاء `Document` ایی که واژه مورد نظر در آن‌ها تکرار شده است). برای مثال در اینجا اگر واژه‌ای 200 بار یا بیشتر در صفحات مختلف شاهنامه تکرار شده باشد، به عنوان یک واژه پر اهمیت انتخاب شده و به ابر واژه‌های نهایی اضافه می‌گردد.

مرحله سوم: استفاده از نتایج

```

using System;
using System.Diagnostics;
using System.IO;
using System.Linq;

namespace ShaahnamehAnalysis
{
    class Program
    {
        static void Main(string[] args)
        {
            CreateIndex.CreateShaahnamehIndex();
            var wordsCloudList = WordsCloud.Create();

            var data = wordsCloudList.Select(x => x.Text + ", " + x.Frequency)
                .Aggregate((s1, s2) => s1 + Environment.NewLine + s2);
            var output = "ShaahnamehAnalysis.txt";
            File.WriteAllText(output, data);
            Process.Start(output);
        }
    }
}

```

که نتیجه 15 مورد اول آن به صورت زیر است:

واژه | فرکانس
 شاه, 1191
 دل, 1088
 سر, 1070
 کار, 840
 لشکر, 801
 تخت, 755
 روز, 745
 ایران, 740
 جهان, 724
 مرد, 660
 دست, 630
 تاج, 623
 نزدیک, 623
 گیتی, 585
 راه, 584

فایل‌های کامل این مثال را از اینجا می‌توانید دریافت کنید:

[ShaahnamehAnalysis.zip](#)

عنوان: نحوه استفاده صحیح از لوسین در ASP.NET
نویسنده: وحید نصیری
تاریخ: ۹:۳۵ ۱۳۹۱/۰۴/۲۹
آدرس: www.dotnettips.info
برچسب‌ها: ASP.Net, Design patterns, Lucene.NET

بر مبنای پیاده سازی متداولی که در n هزار سایت اینترنتی می‌توان یافت، نحوه کار با جستجوگر لوسین حدوداً به این شکل است:

```
var directory = FSDirectory.Open(new DirectoryInfo(Environment.CurrentDirectory + "\\LuceneIndex"));
using (var searcher = new IndexSearcher(directory, readOnly: true))
{
    //do something ...

    searcher.Close();
    directory.Close();
}
```

و ... اینکار به این شکل غلط است!

مطابق [مستندات رسمی](#) لوسین، این کتابخانه thread-safe است. به این معنا که در آن واحد چندین و چند کاربر می‌توانند از یک وهله از شیء‌های Reader و Searcher استفاده کنند و نباید به ازای هر جستجو، یکبار این اشیاء را ایجاد و تخریب کرد. البته در اینجا تنها یک Writer در آن واحد می‌تواند مشغول به کار باشد. مشکلاتی که به همراه باز و بسته کردن بیش از حد IndexSearcher وجود دارد، مصرف بالای حافظه است (به ازای هر کاربر مراجعه کننده، یکبار باید ایندکس‌ها در حافظه بارگذاری شوند) و همچنین تاخیر اولیه این بارگذاری و کندی آن‌را نیز باید مدنظر داشت.

نتیجه گیری:

برای کار با جستجوگر لوسین نیاز است از الگوی [Singleton](#) استفاده شود و تنها یک وهله از این اشیاء بین تردهای مختلف به اشتراک گذاشته شود.

نظرات خوانندگان

نویسنده: حسین غلامی
تاریخ: ۱۳۹۱/۱۰/۱۱ ۱:۲۹

آقای نصیری میشه نمونه ای رو با استفاده از این الگو ، مثال بزنید؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۱۰/۱۱ ۱۰:۹

مراجعه کنید به مثال [Auto Complete](#) .

برای بهبود قسمت ثبت نام در یک سایت بهتر است بین «وحید» و «وَحید» تفاوتی قائل نشد. این مورد ممکن است خصوصا حین ارسال پیام‌های خصوصی در آینده جهت تشخیص افراد مشکل ساز شود. همچنین [در تهیه slug](#) برای نمایش در Urlها نیز باید اعراب را حذف کرد. منظور از slug، عنوان کوتاهی است که در انتهای یک آدرس ممکن است ذکر شود.

<http://www.site.com/post/12/slug>

سؤال: چگونه می‌توان اعراب را از متون فارسی یا عربی حذف کرد؟

متد انجام اینکار را در ذیل مشاهده می‌کنید:

```
using System.Globalization;
using System.Text;

static string RemoveDiacritics(string text)
{
    var normalizedString = text.Normalize(NormalizationForm.FormD);
    var stringBuilder = new StringBuilder();

    foreach (var c in normalizedString)
    {
        var unicodeCategory = CharUnicodeInfo.GetUnicodeCategory(c);
        if (unicodeCategory != UnicodeCategory.NonSpacingMark)
        {
            stringBuilder.Append(c);
        }
    }

    return stringBuilder.ToString().Normalize(NormalizationForm.FormC);
}
```

توضیحات

متد [Normalize](#) با پارامتر `NormalizationForm.FormD`، سبب می‌شود تا کاراکترها به گلیف‌های اصلی تشکیل دهنده‌ی آن‌ها تجزیه شوند. به عبارتی، حروف از اعراب جدا خواهند شد. در ادامه این کاراکترها اسکن شده و صرفا مواردی که حروف پایه را تشکیل می‌دهند، جمع آوری و بازگشت داده می‌شوند. حالت `NormalizationForm.FormC` که در انتها بکار گرفته شده، برعکس است. در یونیکد یک حرف می‌تواند از یک یا چند [code point](#) تشکیل شود. در حالت `FormC`، هر حرف با اعراب آن یک `code point` را تشکیل می‌دهند. در حالت `FormD`، حرف با اعراب آن دو `code point` را تشکیل خواهند داد. به همین جهت در ابتدای کار، رشته تبدیل به حالت `D` شده تا بتوان اعراب آن‌را مجزای از حروف پایه حذف کرد. البته اعراب در اینجا به اعراب عربی ختم نمی‌شود. یک سری حروف اروپایی مانند "ö", "ä", و "ü" را نیز شامل می‌شود.

نظرات خوانندگان

نویسنده: امیر هاشم زاده
تاریخ: ۱۶:۱۲ ۱۳۹۳/۰۲/۱۱

اطلاعات بیشتر در [این پرسش و پاسخ](#) .
[لیست کاراکترهای یونیکد](#) از نوع NonSpacingMark

نویسنده: امیر هاشم زاده
تاریخ: ۱۶:۴۴ ۱۳۹۳/۰۲/۱۱

یک سوال: علت استفاده از حالت FormC در انتهای کد چیست؟ چرا فقط به کد زیر بسنده نکردیم:

```
return stringBuilder.ToString();
```

بوسیله Normalize، می‌توانیم خروجی را با مقدار string دیگر مقایسه نماییم یا بعبارت دیگر خروجی مقایسه پذیر خواهد شد. [در این پرسش و پاسخ](#) بیشتر درباره Normalize بحث شده است.

نویسنده: داوود
تاریخ: ۸:۱۳ ۱۳۹۳/۰۲/۱۳

با سلام
آیا تنوین و تشدید در این حالت جز اعراب محسوب میشوند
و همچنین ی (یای عربی) جز حروف اعراب دار است
تشکر بابت مطلب مفیدتون

نویسنده: وحید نصیری
تاریخ: ۹:۰۲ ۱۳۹۳/۰۲/۱۳

- بله.
- خیر.

نویسنده: علیرضا
تاریخ: ۱۴:۳۹ ۱۳۹۳/۰۲/۱۳

با سلام. برای سرچ یک کلمه بدون اعراب در متنی پر از اعراب باید به چه صورت عمل کرد که بهینه باشد؟
مثلا کلمه‌ی محمد را بخواهیم در دیتابیس‌ی که متن کل قرآن است سرچ کنیم.

نویسنده: وحید نصیری
تاریخ: ۱۴:۵۶ ۱۳۹۳/۰۲/۱۳

جستجوی بهینه‌ی متنی بر روی حجم بالایی از اطلاعات بهتر است توسط روش‌های full text search انجام شود. مثلا از [لوسین](#) استفاده کنید، به همراه [Lucene.Net.Analysis.Analyzer.ArabicAnalyzer](#) آن که مخصوص جستجو بر روی متون عربی است. همچنین اگر از [FTS در SQL Server](#) استفاده می‌کنید باید از [accent insensitive collate](#) استفاده کنید.

نویسنده: وحید نصیری
تاریخ: ۲۳:۱۹ ۱۳۹۳/۰۵/۲۴

اصلاحیه!

کدهای فوق «آ» را تبدیل به «ا» می‌کنند. مشکلی بود که در حین ثبت نام پیش آمده بود. «آفتاب» برای مثال تبدیل به «افتاب»

می‌شد. برای رفع، داخل حلقه:

```
if (unicodeCategory != UnicodeCategory.NonSpacingMark)
{
    stringBuilder.Append(c);
}
else
{
    //اسامی مانند آفتاب نباید خراب شوند
    if (c == 1619) //آ
    {
        stringBuilder.Append(c);
    }
}
```