**Tìm hiểu về Mã Nguồn mở LUCENE cho .NET**

1. Lucene.Net là gì ?

Lucene.Net là một line-by-line port phổ biến của Apache Lucene, một thư viện công cụ tìm kiếm văn bản đầy đủ tính năng, hiệu suất cao được viết hoàn toàn bằng Java. Nó là một công nghệ phù hợp cho hầu như bất kỳ ứng dụng nào yêu cầu tìm kiếm toàn văn bản. Đặc biệt, một ứng dụng mà bạn muốn đạt được một cái gì đó gần với kết quả tìm kiếm của Google và không chỉ kết quả tìm kiếm, nhưng kết quả tìm kiếm rất nhanh, hoặc có thể chỉ là kết quả tìm kiếm cực nhanh.

Vì vậy, trong khi về mặt kỹ thuật có thể, mặc dù hơi khó khăn, ta có thể tích hợp Apache Lucene gốc vào ứng dụng .NET , và nó sẽ cung cấp cho bạn tìm kiếm cực nhanh. Nhưng nó sẽ mất khá nhiều thời gian, do đó làm cho trang web của bạn quá phức tạp và dễ bị lỗi. Mục đích chính của Lucene.Net là dễ dàng tích hợp vào bất kỳ ứng dụng .NET nào và cung cấp hầu hết tốc độ và tính linh hoạt của thư viện gốc Java.

1. Cài đặt Lucene.Net .

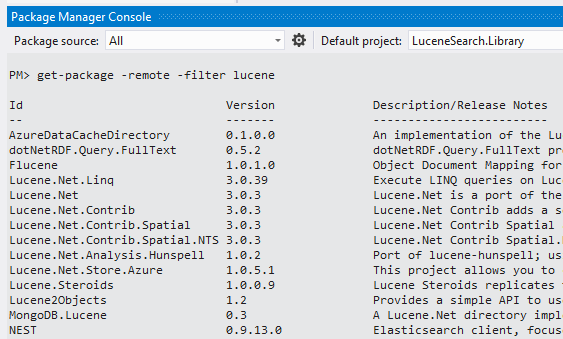
Để cài đặt Lucene.Net ta sẽ cài bằng NuGet

Mở Package Manager Console trong Visual Studio bằng cách nhấp vào:

**View > Other Windows > Package Manager Console**

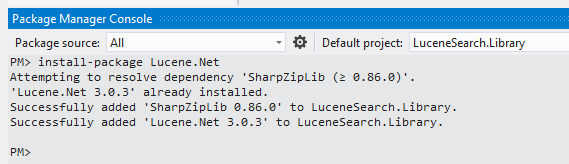
Và một khi bảng điều khiển sẽ bật ở dưới cùng, hãy tìm kiếm 'lucene' đầu tiên bằng cách nhập

'get-package -remote -filter lucene' trong prompt:

 **[1]**

Từ kết quả tìm kiếm, chúng ta có thể thấy rằng có một số gói khác nhau có sẵn cho chúng ta. Đối với hầu hết các phần, các gói này chỉ mở rộng chức năng mặc định của Lucene.Net.Nếu sau này có sử dụng đến thì ta sẽ dùng sau. Gói duy nhất mà chúng ta cần là một Lucene.Net barebone.

Vì vậy, để cài đặt nó, hãy gõ 'install-package Lucene.Net':

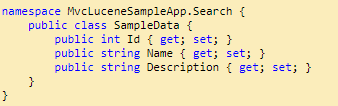
 **[2]**

1. Triển Khai SEARCH dùng Lucene.Net .

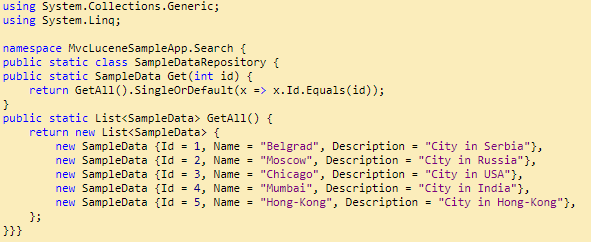
Các bước bạn cần thực hiện để xây dựng tìm kiếm cơ bản Lucene.Net

***Bước 1*** *- Tạo nguồn dữ liệu mẫu*

Để sử dụng Lucene.Net, trước hết chúng ta cần tạo chỉ mục tìm kiếm Lucene từ một số tập dữ liệu (có thể là từ cơ sở dữ liệu). Lucene chỉ mục tìm kiếm chỉ là một tập hợp các tập tin văn bản mà Lucene.Net tạo ra, và ta sẽ tạo ra nó sau này. Vì vậy, hãy tạo một tệp SampleData.cs trống và thêm một số đối tượng dữ liệu chung vào nó:

**[3]**

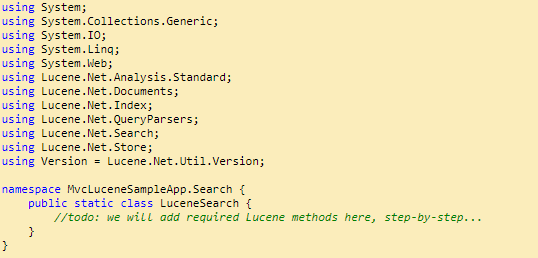
Lớp này có thể đại diện cho bất kỳ dữ liệu nào bạn muốn, và tất nhiên bạn có thể tạo lớp của riêng bạn hoặc sử dụng lớp hiện có. Ngoài ra, bất cứ khi nào nó đến để thử nghiệm Lucene, bạn có thể muốn tạo một kho lưu trữ nguồn dữ liệu tĩnh đơn giản dựa trên một lớp SampleData ở trên, và bạn có thể gắn mã đó vào cùng một tệp SampleData.cs, ​​bạn vừa tạo:

**[4]**

Chúng ta đã tạo nguồn dữ liệu mẫu với lớp dữ liệu chung SampleData và kho mẫu SampleDataRepository để truy xuất dữ liệu tĩnh .

***Bước 2*** *- Tạo lớp tìm kiếm Lucene rỗng*

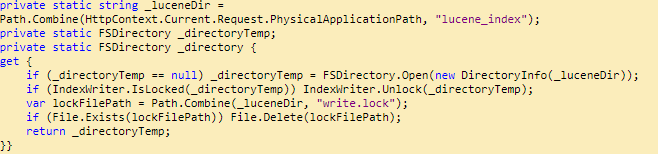
Bây giờ hãy tạo một tệp LuceneSearch.cs rỗng. Hãy thêm những đoạn code này vào file.

**[5]**

Chúng ta sẽ thêm tất cả các phương thức cần thiết để thực hiện tất cả các công việc liên quan đến tìm kiếm đối với lớp đó. Tại thời điểm đó, bạn có thể đảm bảo tất cả các tham chiếu của mình đều chính xác bằng cách tạo trang web của bạn và nếu việc xây dựng không thành công, hãy sửa các tham chiếu của bạn.

***Bước 3*** *- Thêm trình xử lý thư mục chỉ mục tìm kiếm Lucene*

Chúng ta hãy xem những gì chúng ta có bây giờ - lớp SampleData, đại diện cho một số dữ liệu, và lớp LuceneSearch được cho là thực hiện tìm kiếm. Nhưng nó trống rỗng. Vì vậy, nó cần một số code để làm công việc. Một điều kiện tiên quyết nhỏ đầu tiên. Lucene.Net cần xây dựng chỉ mục tìm kiếm của mình, như đã đề cập trước đó, về cơ bản là một tập hợp các tệp được tạo bởi Lucene trong một số thư mục cục bộ. Vì vậy, chúng ta cần thêm một thuộc tính đặc biệt vào lớp LuceneSearch đại diện cho một trình xử lý cho một thư mục cục bộ sẽ lưu trữ chỉ mục tìm kiếm:

**[6]**

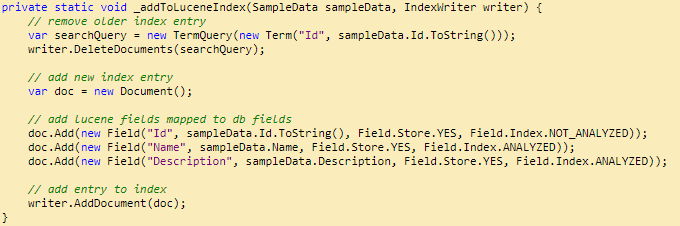
Ở đây \_luceneDir là một đường dẫn vật lý đầy đủ đến thư mục và "lucene\_index" là tên của thư mục đó, nằm trong thư mục gốc của ứng dụng. Vì vậy, tự nhiên bạn cũng cần phải tạo thư mục này bằng tay hoặc viết một số code tạo ra nó tự động. Sau đó, \_directory là một cá thể của lớp FSDirectory của Lucene.Net và sẽ được tất cả các phương thức tìm kiếm sử dụng để truy cập chỉ mục tìm kiếm.

Chúng ta đã thêm trình xử lý thư mục chỉ mục tìm kiếm Lucene để làm cho lớp LuceneSearch của chúng tôi sẵn sàng để thêm các phương thức tìm kiếm.

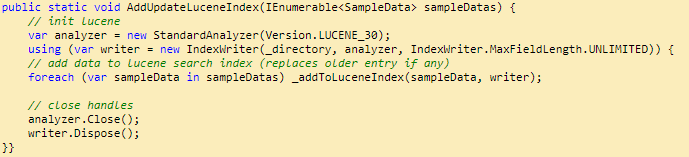
***Bước 4*** *- Thêm các phương thức để Thêm dữ liệu vào chỉ mục tìm kiếm Lucene*

Lucene sẽ tạo chỉ mục tìm kiếm dựa trên một số dữ liệu thực tế, trong trường hợp của chúng ta, nó sẽ là một List <SampleData> với một vài bản ghi, hoặc ghi lại bản ghi SampleData.

Phương pháp đầu tiên chúng ta cần là một phương thức riêng tạo một mục chỉ mục tìm kiếm dựa trên dữ liệu của chúng ta và nó sẽ được tái sử dụng bởi các phương thức công khai mà chúng ta sẽ bổ sung sau:

**[7]**

Về cơ bản, phải mất một bản ghi với một lớp SampleData, ánh xạ nó vào tài liệu lớp Lucene và thêm nó vào chỉ mục tìm kiếm bằng cách sử dụng IndexWriter. Xin lưu ý các trường **<code> Name** và **<code> Description** sử dụng tham số **Field.Index.ANALYZED**, trong khi **Id** sử dụng **Field.Index.NOT\_ANALYZED** một. Về cơ bản, bạn chỉ muốn sử dụng **ANALYZED** trên các thuộc tính văn bản hoặc chuỗi đơn và **NOT\_ANALYZED** trên các giá trị số ít, như các **số nguyên Id**. Bây giờ hãy thêm phương thức công khai sẽ sử dụng \_addToLuceneIndex () để thêm danh sách các bản ghi vào chỉ mục tìm kiếm:

**[8]**

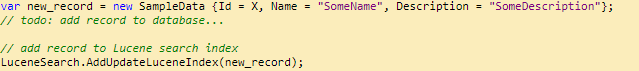
Ngoài ra, hãy thêm phương thức công khai khác sẽ thêm một bản ghi vào chỉ mục tìm kiếm:

**[9]**

Mẫu sử dụng của hai phương thức này rất đơn giản - thêm tất cả các bản ghi cơ sở dữ liệu một lần:

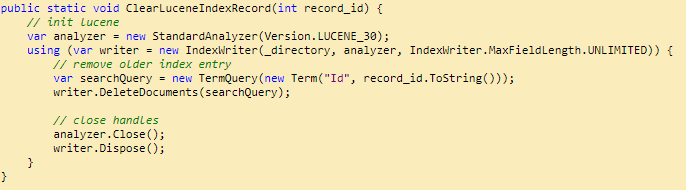
**[10]**

Sau đó, bất cứ khi nào bản ghi mới được thêm vào cơ sở dữ liệu, hãy gọi cùng một phương thức, chỉ cung cấp cho nó bản ghi duy nhất:

**[11]**

***Bước 5*** *- Thêm các phương thức xóa và tối ưu hóa dữ liệu từ chỉ mục tìm kiếm Lucene*

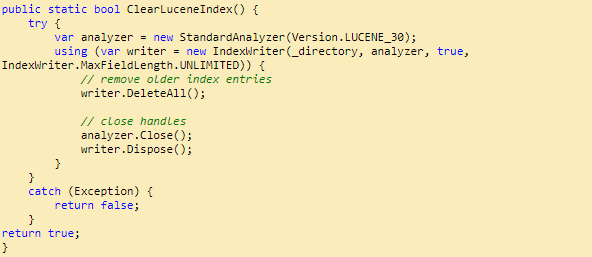
Bất cứ khi nào banr ghi cơ sở dữ liệu của chúng ta bị xóa, chúng cũng sẽ bị xóa khỏi chỉ mục tìm kiếm Lucene, nếu không, tìm kiếm của bạn sẽ trả về các bản ghi có thể không tồn tại trong cơ sở dữ liệu. Trước tiên, hãy thêm một phương thức đơn giản để xóa bản ghi đơn khỏi chỉ mục tìm kiếm Lucene theo trường Id của bản ghi:

**[12]**

Mã trên cơ bản tìm kiếm bản ghi theo trường Id (và bạn có thể sử dụng bất kỳ trường nào), nhận kết quả tìm kiếm từ chỉ mục và xóa tất cả chúng khỏi chỉ mục và trong trường hợp của chúng ta, nó thường sẽ xóa bản ghi duy nhất, miễn là bạn luôn cung cấp giá trị Id đầy đủ và duy nhất.

Lưu ý: các phương thức tìm kiếm chính của Lucene sẽ được thêm vào sau và dĩ nhiên được giải thích chi tiết hơn.

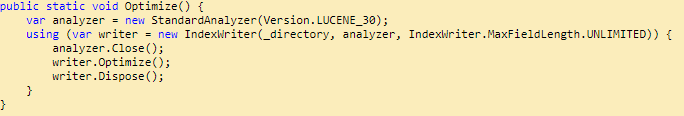
Thứ hai, bất cứ khi nào thay đổi lược đồ cơ sở dữ liệu, hoặc bạn chỉ muốn xóa toàn bộ chỉ mục một cách nhanh chóng, bạn sẽ cần một phương thức để xóa tất cả chỉ mục, vì vậy hãy thêm quá trình đó:

**[13]**

Phương thức này chỉ đơn giản loại bỏ toàn bộ chỉ mục tìm kiếm Lucene thông qua một phương thức được xây dựng trong Lucene IndexWriter. Bây giờ có lẽ là một thời điểm tốt để đề cập đến việc Lucene đặt một "khóa" vào các tệp chỉ mục tìm kiếm, vì vậy khi chúng được cập nhật hoặc tìm kiếm, vì vậy chúng không thể bị thay đổi. Ngoài ra, điều quan trọng là mỗi tệp được "mở khóa" vì nếu một số tệp sẽ bị xóa theo cách thủ công hoặc bằng một số mã và bạn cố gắng tìm kiếm / cập nhật sau đó, nó sẽ khiến bạn thấy nhiều lỗi và chúng ta đều không muốn điều đó.

Vì vậy, chúng ta luôn cần .Close () và .Dispose () bất kỳ trình xử lý Lucene nào như IndexWriter và StandardAnalyzer.

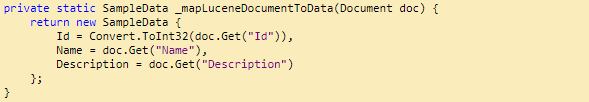
Ngoài ra, sẽ có ích khi chạy tối ưu hóa chỉ mục tìm kiếm Lucene một lần trong một thời gian để tăng tốc tìm kiếm, đặc biệt nếu chỉ mục của bạn ngày càng lớn hơn. Vì vậy, hãy thêm một phương thức nhỏ để làm điều đó:

**[14]**

Chúng ta đã thêm ba phương thức - ClearLuceneIndexRecord () để xóa bản ghi duy nhất khỏi chỉ mục tìm kiếm Lucene, ClearLuceneIndex () để xóa tất cả các bản ghi trong chỉ mục và phương thức Optimize () để tối ưu hóa các chỉ mục lớn để tìm kiếm nhanh hơn.

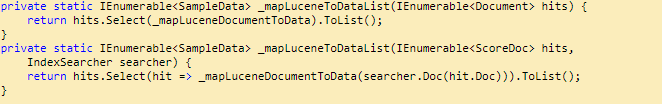
***Bước 6*** *- Thêm các phương thức để ánh xạ dữ liệu chỉ mục tìm kiếm Lucene đến SampleData*

Dừng ở đó, mọi người, bước này là bước cuối cùng, trước khi chúng ta thêm các phương pháp tìm kiếm chính của mình! Bây giờ trước khi chúng ta có thể tìm kiếm, bạn có nhớ phương thức **\_addToLuceneIndex ()** của chúng ta từ **Bước 4**, ánh xạ dữ liệu cơ sở dữ liệu của chúng tôi tới chỉ mục tìm kiếm Lucene không? Vâng, để có được kết quả tìm kiếm của chúng ta dưới dạng **ListDampleData** hoặc tương tự, chúng ta cần một hàm sẽ ánh xạ chỉ mục tới lớp **SampleData** của chúng ta, và ở đây nó là:

**[15]**

Vì vậy, phương thức trên sẽ lấy tài liệu Lucene từ chỉ mục (trong đó mỗi trường được biểu diễn dưới dạng một chuỗi) và sẽ ánh xạ nó tới SampleData. Khá đơn giản.

Ngoài ra, chúng ta cần thêm hai phương thức nữa để ánh xạ một danh sách các tài liệu Lucene và một danh sách các Lucene ScoreDocs, mỗi phương thức được trả về bởi phương thức tìm kiếm Lucene khác nhau, và nhiều hơn nữa trong bước 7 và 8. Cả hai phương thức đó sử dụng lại phương thức **\_mapLuceneDocumentToData ()** được xác định ở trên:

**[16]**

Chúng ta đã thêm các phương thức để ánh xạ kết quả do Lucene trả về cho lớp dữ liệu của chúng tôi để được sử dụng lại trên trang web. Phương thức **\_mapLuceneDocumentToData ()** ánh xạ Tài liệu Lucene với các kết quả tìm kiếm từ chỉ mục đến lớp của chúng ta **SampleData** và phương thức **\_mapLuceneToDataList ()** lần lượt ánh xạ một danh sách các tài liệu Lucene hoặc ScoreDocs.

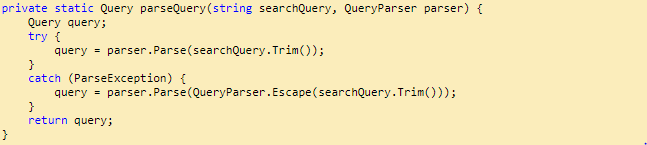
***Bước 7*** *- Thêm phương pháp tìm kiếm chính*

Và cuối cùng, phương pháp tìm kiếm chính. Nó sẽ tìm kiếm chỉ mục tìm kiếm Lucene theo một trường cụ thể (Id, Name hoặc Description) bất cứ khi nào chúng ta cung cấp tên, hoặc, nó sẽ tìm kiếm cả ba trường, đây là cơ sở cho tìm kiếm phổ biến tương tự như công cụ tìm kiếm internet your\_favourite . Bạn có thể nhận thấy rằng nó vẫn là một phương pháp riêng. Lý do cho điều đó là phương thức này là phổ quát cho bất kỳ truy vấn Lucene nào.

Truy vấn Lucene không chỉ là văn bản mà bạn đang tìm kiếm. Ví dụ cơ bản là khi truy vấn trông giống như "Mumbai", Lucene sẽ tìm kiếm đối sánh chính xác cho từ này và nếu truy vấn giống như "Mum \*" thì tất cả các trường có từ bắt đầu bằng "Mẹ" sẽ được trả lại dưới dạng kết quả tìm kiếm. Chắc chắn có nhiều cách để viết các quả Lucene tiên tiến, nhưng điều đó sẽ không được đề cập trong bài viết này.

Trong trường hợp của chúng ta, truy vấn sẽ được cung cấp bởi phương thức công khai, định dạng bạn sẽ yêu cầu tìm kiếm một trường hợp tìm kiếm cụ thể và nó sẽ được thêm vào Bước 8. Kịch bản của chúng ta là phương pháp tìm kiếm riêng dưới đây không thay đổi nhiều, và chỉ một công chúng sẽ được điều chỉnh cho nhu cầu tìm kiếm của chúng tôi.

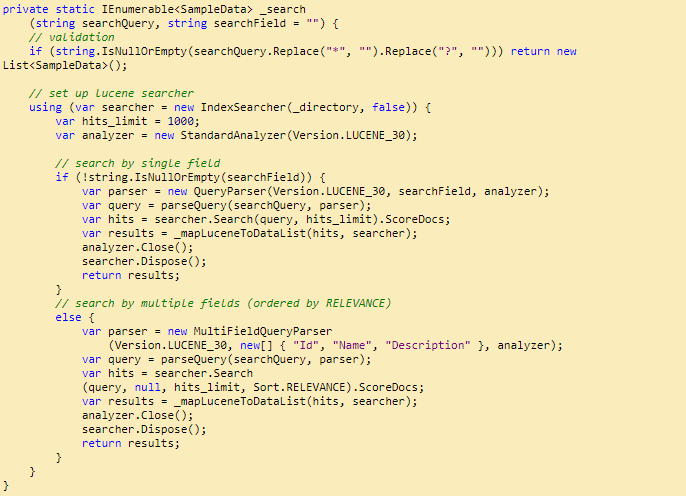
Nhưng một điều kiện tiên quyết nhỏ, hãy thêm phương thức này trước:

**[17]**

Phương pháp này sẽ làm gì, về cơ bản phân tích chuỗi truy vấn tìm kiếm của bạn thành đối tượng Lucene Query và nếu phân tích cú pháp không thành công, nó sẽ sửa bất kỳ ký hiệu nào gây ra lỗi và trả về đối tượng Query cố định.

Lưu ý: Nếu bạn muốn Lucene tìm các từ tương tự, bạn sẽ cần sử dụng **FuzzyQuery** thay vì Truy vấn thông thường, xem ví dụ tại đây: *Http://www.codeproject.com/Messages/4643090/Re-Find- tương tự-Words.aspx*

Và bây giờ hãy thêm phương thức tìm kiếm chính của chúng ta:

**[18]**

Tất nhiên phương thức \_search () riêng của chúng ta có nhiều điểm để được điều chỉnh và tối ưu hóa, và nó sẽ tùy thuộc vào bạn để điều chỉnh nó theo nhu cầu cụ thể của bạn.

Bạn có nhận thấy **(var hits\_limit = 1000 ; )** ? Như đã đề cập trong Bước 5, khi Lucene nhận được hơn 1000 kết quả tìm kiếm, nó trở nên ngày càng chậm, vì vậy bạn muốn giới hạn nó thành một số có liên quan trong trường hợp của bạn.

Nếu bạn thêm các trường mới vào chỉ mục của mình, đừng quên thêm tên trường để tìm kiếm vào dòng này trong nhiều trường tìm kiếm như sau:

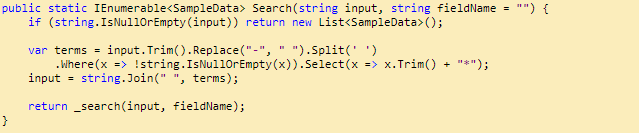
**[19]**

Chỉ cần nhớ rằng các kết quả trả về bởi phương thức **\_search ()** sẽ được sắp xếp theo **Sort.RELEVANCE**, có nghĩa là kết quả chính xác hơn sẽ được trả về trước. Một tùy chọn khác là **Sort.INDEXORDER**, trả về kết quả theo thứ tự mà chúng đã được thêm vào chỉ mục tìm kiếm. Tuy nhiên đối với hầu hết các trường hợp, **Sort.RELEVANCE** sẽ hoạt động tốt.

Chúng ta đã thêm phương thức **private \_search ()** chính sẽ thực hiện tìm kiếm theo trường đơn hoặc nhiều trường trong chỉ mục tìm kiếm Lucene dựa trên truy vấn tìm kiếm được cung cấp.

***Bước 8*** *- Thêm phương thức công khai gọi phương thức tìm kiếm chính*

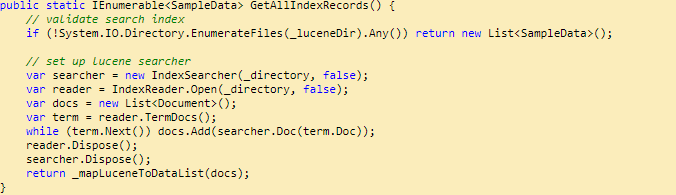
Công cụ tìm kiếm nhỏ của chúng ta được thực hiện ngay bây giờ. Bây giờ là lúc thêm các phương thức cuối cùng sẽ giao tiếp với trang web hoặc ứng dụng của chúng tôi. Việc đầu tiên chỉ đơn giản là định dạng truy vấn tìm kiếm Lucene và gọi phương thức tìm kiếm riêng tư chính **\_search ():**

**[20]**

Nó thay thế tất cả dấu gạch ngang "-" trong yêu cầu tìm kiếm của bạn và thêm "\*" (dấu sao) sau mỗi từ để bạn có thể tìm kiếm theo từng từ. Vì vậy, về cơ bản, phương pháp Search () ở trên là khá nhiều sân chơi chính của bạn, nơi bạn tùy chỉnh querries của bạn để có được kết quả tìm kiếm hoàn hảo. Ngoài ra, để thử các hạt tìm kiếm gốc Lucene, chúng ta có thể thêm phương pháp tìm kiếm mặc định **SearchDefault ()**, không định dạng truy vấn của bạn theo bất kỳ cách nào:

**[21]**

Tìm kiếm trên internet về cách viết các cụm tìm kiếm Lucene nâng cao và chính xác, hãy thử sử dụng hàm **SearchDefault ()** , sau đó bạn có thể sửa đổi phương thức Search () để phù hợp hơn với nhu cầu của bạn. Như đã đề cập trong **Bước 5**, bây giờ chúng ta cũng đang thêm một phương thức để trả về toàn bộ chỉ mục tìm kiếm Lucene:

**[22]**

Hãy nhớ, sử dụng nó một cách hợp lý, bởi vì chỉ số càng lớn thì càng mất nhiều thời gian để tải hết.

Cuối cùng thì chúng ta đã thêm hai phương pháp cho đến nay hoàn thành lớp LuceneSearch cơ bản - phương thức đầu tiên là Search (), định dạng truy vấn tìm kiếm Lucene và tìm kiếm theo tất cả các trường hoặc bởi một trường đơn lẻ. Phương thức thứ hai là **GetAllIndexRecords ()** chỉ trả về tất cả các bản ghi trong chỉ mục tìm kiếm.

**THAM KHẢO:**

Hình [1] – [22]: Tham khảo từ trên Intermet.

Link : *https://www.codeproject.com/Articles/320219/Lucene-Net-ultra-fast-search-for-MVC-or-WebForms*