

```
1> lucene_server:start().  
ok
```

```
2> User1 =  
    [{name, "Fernando"}  
     ,{nick, "elbrujoalcon"}].  
[{name, "Fernando"},{nick, "elbrujoalcon"}]  
  
3> User2 =  
    [{name, "Ariel Ortega"}  
     ,{nick, "burrito"}].  
[{name, "Ariel Ortega"},{nick, "burrito"}]  
  
4> lucene:add([User1,User2]).  
ok
```

```
5> lucene:match("nick: b*", 10).  
{[[{name,"Fernando Benavides"},  
    {nick,"elbrujoalcon"},  
    {'score',1.0}]],  
 [{total_hits,1},  
  {first_hit,1},  
  {query_time,783},  
  {search_time,457}]}
```

```
6> lucene:add([[{code, X}] || X <- lists:seq(1,10)]).  
ok
```

```
7> {_, M} = lucene:match("code:[0 TO 10]", 5).  
{[[{code,1},{ 'score',1.0}],  
  [{code,2},{ 'score',1.0}],  
  [{code,3},{ 'score',1.0}],  
  [{code,4},{ 'score',1.0}],  
  [{code,5},{ 'score',1.0}]],  
  [{total_hits,10},  
   {first_hit,1},  
   {query_time,14335},  
   {search_time,3811},  
   {next_page,<<172,237,0,5,115,114,0,36,99,111,109,46,  
                116,105,103,101,114,116,101,120,...>>}]}
```

```
8> lucene:continue(proplists:get_value(next_page, M), 5).  
{[[{code,6},{ 'score',1.0}],  
  [{code,7},{ 'score',1.0}],  
  [{code,8},{ 'score',1.0}],  
  [{code,9},{ 'score',1.0}],  
  [{code,10},{ 'score',1.0}]],  
  [{total_hits,10},  
   {first_hit,6},  
   {query_time,2368},  
   {search_time,1731}]}
```

```
{pre_hooks, [{compile, "mkdir -p bin"},
               {compile, "./copy-jinterface.sh"}]}.

{post_hooks,
 [{clean,
   "rm -rf bin priv/lucene_server.jar priv/OtpErlang.jar"},
  {compile,
   "javac -g -deprecation -sourcepath java_src
        -classpath ./bin:./priv/* -d bin
        java_src/**/*.java"},
  {compile, "jar cf priv/lucene-server.jar -C bin ."}]}.

```

```

init([]) ->
...
    Port =
        erlang:open_port(
            {spawn_executable, Java},
            [{line,1000}, stderr_to_stdout,
            {args, JavaArgs ++
                ["-classpath", Classpath,
                "com.tigertext.lucene.LuceneNode",
                ThisNode, JavaNode, erlang:get_cookie(),
                integer_to_list(AllowedThreads)]}],
        wait_for_ready(
            #state{java_port = Port, java_node = JavaNode})
end.
...
wait_for_ready(State = #state{java_port = Port}) ->
    receive
        {Port, {data, {eol, "READY"}}} ->
            true = link(process()),
            true = erlang:monitor_node(State#state.java_node, true),
            {ok, State};
        Info ->
            ...
    end.

```

```
public static void main(String[] args) {
    String peerName = args.length >= 1 ? args[0]
        : "lucene_server@localhost";
    String nodeName = args.length >= 2 ? args[1]
        : "lucene_server_java@localhost";

    try {
        NODE = args.length >= 3 ? new OtpNode(nodeName, args[2])
            : new OtpNode(nodeName);
        PEER = peerName;

        final OtpGenServer server = new LuceneServer(NODE);

        server.start();

        System.out.println("READY");

    } catch (IOException e1) {
        jlog.severe("Couldn't create node: " + e1);
        e1.printStackTrace();
        System.exit(1);
    }
}
```