```
using System;
using System.Text;
using System.Text.RegularExpressions;
using System.Collections.Generic;
using System.Diagnostics;

Stopwatch stopwatch = Stopwatch.StartNew();

IDocument Document = Context.Document;

IField diary = Context.Field("Diary");
if (diary != null) diary.Value = string.Format("{0:F2} Searcher
started\n", stopwatch.Elapsed.TotalSeconds);

// Clear any existing "Keyword Search" hits.
if (diary != null) diary.Text += string.Format("{0:F2} Clearing previous
search results\n", stopwatch.Elapsed.TotalSeconds);
for (int targetIndex = 0; targetIndex < 100; targetIndex++)
{
    if ( ! Context.HasField("Target" + targetIndex) ) break;

    IField fldTarget = Context.Field("Target" + targetIndex);
    foreach (IFieldRegion reg in fldTarget.Regions)
    {
        if (reg.IsMatched) reg.Delete();
    }

    fldTarget.Value = DBNull.Value;
    if (diary != null) diary.Text += string.Format("{0:F2} Cleared {1}\n",
stopwatch.Elapsed.TotalSeconds, fldTarget.FullName);
}

if (diary != null) diary.Text += string.Format("{0:F2} Gathering search
targets\n", stopwatch.Elapsed.TotalSeconds);

IField targetsField = Context.Field("Targets");

if (string.IsNullOrEmpty(targetsField.Text))
{
    if (diary != null) diary.Text += string.Format("{0:F2} No \"Targets\"
specified (dataset is empty?)\n", stopwatch.Elapsed.TotalSeconds);
    else FCTools.ShowMessage("No \"Targets\" specified (dataset is
empty?)");
    return;
}

if (diary != null) diary.Text += string.Format("{0:F2} Splitting search
target into array\n", stopwatch.Elapsed.TotalSeconds);
string[] targetArray = targetsField.Text.Split('\n');

Dictionary<string, Regex> targets = new Dictionary<string,
Regex>(targetArray.Length); // user-specified target, computed RegEx
```

```
if (diary != null) diary.Text += string.Format("{0:F2} Constructing search
target Regular Expressions\n", stopwatch.Elapsed.TotalSeconds);
for (int recIndex = 0; recIndex < targetArray.Length; recIndex++)
{
    string key = targetArray[recIndex];

    if ( ! string.IsNullOrEmpty(key) ) key = key.Trim();
    if (string.IsNullOrEmpty(key)) continue;

    if (diary != null) diary.Text += string.Format("{0:F2} Preparing
search target: {1}\n", stopwatch.Elapsed.TotalSeconds, key);

    // create a RegEx-safe sanitized version of the target
    string re = key.Replace("[", @"\[").Replace("]", @"\]")    // protect
user-specified square brackets
                   .Replace("(", @"\(").Replace(")", @"\)")    // protect
user-specified parentheses
                   .Replace("-", @"\p{Pd}")                    // make "-"
match any kind of dash
                   .Replace(" ", @"\s*");                      // make " "
match any amount of whitespace

    Regex value = new Regex(re, RegexOptions.IgnoreCase |
RegexOptions.Compiled);

    targets.Add(key, value);
    if (diary != null) diary.Text += string.Format("{0:F2} Added search
target\n", stopwatch.Elapsed.TotalSeconds);
}

if (targets.Count == 0)
{
    if (diary != null) diary.Text += string.Format("{0:F2} No non-null
\"Targets\" specified (dataset is empty?)\n",
stopwatch.Elapsed.TotalSeconds);
    if (diary == null) FCTools.ShowMessage("No non-null \"Targets\"
specified (dataset is empty?)");
    return;
}
else
{
    if (diary != null) diary.Text += string.Format("{0:F2}
targets.Count={1}\n", stopwatch.Elapsed.TotalSeconds, targets.Count);
}

IPages  Pages = Document.Pages;
int hitCount = 0;
IField fldFound = null;

//int MAX_SECONDS = 10 * (targets.Count+1);

foreach (KeyValuePair<string, Regex> target in targets)
{
    //if (stopwatch.Elapsed.TotalSeconds > MAX_SECONDS)
```

```
    //{
    //    throw new ApplicationException("Loop aborted: exceeded time
limit of " + MAX_SECONDS + " seconds.");
    //}

    if (diary != null) diary.Text += string.Format("{0:F2} Searching for
target: {1}\n", stopwatch.Elapsed.TotalSeconds, target.Key);

    bool firstHit = true;
    for (int pageIndex = 0; pageIndex < Pages.Count; pageIndex++)
    {
        //if (stopwatch.Elapsed.TotalSeconds > MAX_SECONDS)
        //{
        //    throw new ApplicationException("Loop aborted: exceeded time
limit of " + MAX_SECONDS + " seconds.");
        //}

        if (diary != null) diary.Text += string.Format("{0:F2} Examining
page {1}\n", stopwatch.Elapsed.TotalSeconds, (pageIndex + 1));

        IPage Page = Pages[pageIndex];
        string fullText = Page.FullText;
        IRects fullTextCharRects = Page.FullTextCharRects;

        if (diary != null) diary.Text += string.Format("{0:F2} Page {1}
char count={2}\n", stopwatch.Elapsed.TotalSeconds, (pageIndex + 1),
fullText.Length);

        MatchCollection matches = target.Value.Matches(fullText);

        if (matches != null && matches.Count > 0)
        {
            if (diary != null) diary.Text += string.Format("{0:F2} Page
{1} matches.Count={2}\n", stopwatch.Elapsed.TotalSeconds, (pageIndex + 1),
matches.Count);
            foreach (Match match in matches)
            {
                //if (stopwatch.Elapsed.TotalSeconds > MAX_SECONDS)
                //{
                //    throw new ApplicationException("Loop aborted:
exceeded time limit of " + MAX_SECONDS + " seconds.");
                //}

                // there should be only one group (the entire match)
                if (match.Groups.Count > 0 && match.Groups[0].Success)
                {
                    int startPos = match.Groups[0].Index;
                    int stopPos  = match.Groups[0].Index +
match.Groups[0].Length - 1;

                    if ( diary != null) diary.Text +=
string.Format("{0:F2} Found at [{1}..{2}]: {3}\n",
stopwatch.Elapsed.TotalSeconds, startPos, stopPos, match.Groups[0].Value);
                    if (firstHit)
```

```
                    {
                        string fieldName = "Target" + hitCount++;
                        if ( ! Context.HasField(fieldName) ) return;
                        fldFound = Context.Field(fieldName);
                        firstHit = false;
                    }

                    // Calculate the surrounding rect for the character
regions
                    // This is complicated because there can be more than
one box.
                    int left   = -1;
                    int top    = -1;
                    int right  = -1;
                    int bottom = -1;
                    string strRect = string.Empty;
                    for (int pos = startPos; pos <= stopPos; pos++)
                    {
                        //if (stopwatch.Elapsed.TotalSeconds >
MAX_SECONDS)
                        //{
                        //    throw new ApplicationException("Loop
aborted: exceeded time limit of " + MAX_SECONDS + " seconds.");
                        //}

                        IRect rect = fullTextCharRects[pos];

                        if (rect.Right == 0 || rect.Bottom == 0) continue;

                        if (left == -1)
                        {
                            // first rect
                            left   = rect.Left;
                            top    = rect.Top;
                            right  = rect.Right;
                            bottom = rect.Bottom;
                        }
                        else
                        {
                            int horzDiff = rect.Left - right;
                            bool vertOverlap = (rect.Bottom >= top) &&
(rect.Top <= bottom);

                            if ( vertOverlap && horzDiff < 8 )
                            {
                                // this rect's left edge corresponds with
the last rect's right edge
                                top    = Math.Min(top   , rect.Top   );
                                bottom = Math.Max(bottom, rect.Bottom);
                                right  = Math.Max(right, rect.Right);
                            }
                            else
                            {
```

```
                                // Doesn't align; record it and start a
new rect
                                strRect = "[" + left + "," + top + "," +
right + "," + bottom + "]";
                                if (diary != null) diary.Text +=
string.Format("{0:F2} Adding region: {1}\n",
stopwatch.Elapsed.TotalSeconds, strRect);

                                fldFound.AddRegion(Page, strRect);

                                left   = rect.Left;
                                top    = rect.Top;
                                right  = rect.Right;
                                bottom = rect.Bottom;
                            }
                        }
                    }

                    if (left != -1)
                    {
                        // Add the last surrounding rect
                        strRect = "[" + left + "," + top + "," + right +
"," + bottom + "]";
                        if (diary != null) diary.Text +=
string.Format("{0:F2} Adding final region: {1}\n",
stopwatch.Elapsed.TotalSeconds, strRect);
                        fldFound.AddRegion(Page, "[" + left + "," + top +
"," + right + "," + bottom + "]");
                    }

                    fldFound.Value = match.Groups[0].Value;
                }
            }
        }
        else
        {
            if (diary != null) diary.Text += string.Format("{0:F2} Page
{1} No matches\n", stopwatch.Elapsed.TotalSeconds, (pageIndex + 1));
        }
    }
}
```