
Subject: Re: How to add/enable spell-CORRECTING with richedit
(not-quite-solved-sort-of)

Posted by [koldo](#) on Thu, 03 Jan 2019 09:23:52 GMT

[View Forum Message](#) <> [Reply to Message](#)

Hello Mirek

To measure the "difference" between words to see what are the best matches in dictionary, I use the Levenshtein distance and the DamerauLevenshtein distance. One is faster and the other, better.

As the implementation considers all characters as char, prior to this I normalize accented and special characters.

This is the implementation:

```
int LevenshteinDistance(const char *s, const char *t) {  
    int lens = int(strlen(s));  
    int lent = int(strlen(t));
```

```
    Buffer<int> v0(lent + 1);  
    Buffer<int> v1(lent + 1);
```

```
    for (int i = 0; i <= lent; ++i)  
        v0[i] = i;
```

```
    for (int i = 0; i < lens; ++i) {  
        v1[0] = i + 1;
```

```
        for (int j = 0; j < lent; ++j) {  
            int deletionCost = v0[j + 1] + 1;  
            int insertionCost = v1[j] + 1;  
            int substitutionCost;  
            if (s[i] == t[j])  
                substitutionCost = v0[j];  
            else  
                substitutionCost = v0[j] + 1;
```

```
            v1[j + 1] = min(deletionCost, insertionCost, substitutionCost);  
        }
```

```
        Swap(v0, v1);  
    }
```

```
    return v0[lent];  
}
```

```
int DamerauLevenshteinDistance(const char *s, const char *t, int alphabetLength) {  
    int lens = int(strlen(s));  
    int lent = int(strlen(t));  
    int lent2 = lent + 2;  
    Buffer<int> H((lens+2)*lent2);
```

```

int infinity = lens + lent;
H[0] = infinity;
for(int i = 0; i <= lens; i++) {
H[lent2*(i+1)+1] = i;
H[lent2*(i+1)+0] = infinity;
}
for(int j = 0; j <= lent; j++) {
H[lent2*1+(j+1)] = j;
H[lent2*0+(j+1)] = infinity;
}
Buffer<int> DA(alphabetLength, 0);

for(int i = 1; i <= lens; i++) {
    int DB = 0;
    for(int j = 1; j <= lent; j++) {
        int i1 = DA[t[j]-1];
        int j1 = DB;
        int cost = (s[i-1] == t[j-1]) ? 0 : 1;
        if(cost == 0)
            DB = j;
        H[lent2*(i+1)+j+1] =
            min(H[lent2*i    + j] + cost,
                H[lent2*(i+1) + j] + 1,
                H[lent2*i    + j+1] + 1,
                H[lent2*i1   + j1] + (i-i1-1) + 1 + (j-j1-1));
    }
    DA[s[i-1]] = i;
}
return H[lent2*(lens+1)+lent+1];
}I hope this helps.

```
