Subject: Re: Adding new .scd spelling dictionary
Posted by mirek on Fri, 30 Oct 2009 15:48:00 GMT
View Forum Message <> Reply to Message

koldo wrote on Thu, 29 October 2009 06:28Hello all

I would like to add a new .scd spelling dictionary and begin to use it to detect spelling errors when writing .tpp help files.

How can I create a new .scd file and add new words to it ?

Best regards
Koldo

Ha, good idea.

The problem is that existing .scd files were created way back in 2002 year. So the code can be outdated today.

And... it took me more than hour to find it. But I am glad we are about to refresh this one - and potentionally adding more .scd files.

Anyway, here we go:

```
#include <Speller/Speller.h>

byte                charset;
int                 vocn[256];
Vector<String>      voc;
VectorMap<int, String> line;

int LineCode(const String& s)
{
 return ToLower(s[0], CHARSET_DEFAULT) +
        (ToLower(s[1], CHARSET_DEFAULT) << 8) +
        (ToLower(s[2], CHARSET_DEFAULT) << 16);
}

bool Contains(const String& a, const String& b)
{
 for(int i = 0; i + b.GetLength() <= a.GetLength(); i++)
  if(memcmp(~a + i, ~b, b.GetLength()) == 0) return true;
 return false;
}

struct LengthOrder
{
```

```
 bool operator()(const String& a, const String& b) const
 {
  return a.GetLength() > b.GetLength();
 }
};

struct NoCaseOrder
{
 bool operator()(const String& a, const String& b) const
 {
  String la = ToLower(a);
  String lb = ToLower(b);
  return la != lb ? la < lb : a > b;
 }
};

void Make()
{
 FileIn in("f:/dict/cs_cz.txt");
 SetDefaultCharset(CHARSET_WIN1250);
 Vector<String> w;
 Index<int> alphabet;
 String maxl;
 int    maxlen = 0;
 while(!in.IsEof()) {
  String l = in.GetLine();
  if(l.GetLength() > maxlen) {
   maxlen = l.GetLength();
   maxl = l;
  }
  if(l.GetLength() > 1) {
   if(l.GetLength() == 2)
    l.Cat(127);
   w.Add(l);
   for(const char *s = l; s < l.End(); s++)
    alphabet.FindAdd((byte)*s);
  }
 }

 printf("Words loaded, now sorting\n");

 ASSERT(maxlen < 64);

 LOG("Maximal length:" << maxlen << "  " << maxl);

 Sort(w, NoCaseOrder());

 printf("Sorted, now gathering voc candidates\n");
```

```
// ------------------

 VectorMap<String, int> part;
 int dict = 0;
 int i = 0;
 while(i < w.GetCount()) {
  int linecode = LineCode(w[i]);
  String prevw;
  printf("line %s\n", ~ToLower(w[i].Mid(0, 3)));
  while(i < w.GetCount() && LineCode(w[i]) == linecode) {
   String ww = w[i];
   for(int j = 0; j < prevw.GetLength(); j++)
    if(ww[j] != prevw[j]) break;
   if(j >= dict)
    dict = j + 1;
   for(int l = 2; l < ww.GetLength() - 1; l++)
    for(int q = j; q + l <= ww.GetLength(); q++)
     part.GetAdd(ww.Mid(q, l), 0)++;
   prevw = ww;
   i++;
  }
 }

 printf("Creating voc\n");

 int dcount = 256 - dict;
 RLOG("dict: " << dict);
 RLOG("dict size: " << dcount);
 RLOG(" alphabet:" << alphabet.GetCount());
 RLOG(" combinations: " << dcount - alphabet.GetCount());

 for(i = 0; i < alphabet.GetCount(); i++)
  voc.Add(String(alphabet[i], 1));

 Vector<int> value;

 for(i = 0; i < part.GetCount(); i++)
  value.Add() = part[i] * (part.GetKey(i).GetLength() - 1);

 while(voc.GetCount() + dict < 256) {
  int m = 0;
  int mi = 0;
  int i;
  for(i = 0; i < part.GetCount(); i++)
   if(value[i] > m) {
    m = value[i];
    mi = i;
```

```
   }
  if(m <= 0) break;
  String v = part.GetKey(mi);
  vocn[voc.GetCount()] = value[mi];
  voc.Add(v);
  RLOG("Adding " << v << " value:" << value[mi] << " count:" << part[mi]);
  printf("Adding %s value %d\n", ~v, value[mi]);
  for(i = 0; i < part.GetCount(); i++) {
   if(Contains(part.GetKey(i), v))
    value[i] -= v.GetLength() * part[i];
   if(Contains(v, part.GetKey(i)))
    value[i] -= part.GetKey(i).GetLength() * part[i];
  }
  value[mi] = 0;
 }

 int sum = 0;
 for(i = 0; i < voc.GetCount(); i++) {
  sum += vocn[i];
  RLOG(vocn[i] << "  " << voc[i]);
 }
 RLOG("Total " << sum);

// ------------------
 Sort(voc, LengthOrder());

 i = 0;
 while(i < w.GetCount()) {
  int linecode = LineCode(w[i]);
  String& ln = line.GetAdd(linecode);
  printf("LINE %s\n", ToLower(~w[i].Mid(0, 3)));
  RLOG("---- Line " << ToLower(~w[i].Mid(0, 3)));
  String prevw;
  bool next = false;
  while(i < w.GetCount() && LineCode(w[i]) == linecode) {
   String ww = w[i];
   for(int j = 0; j < prevw.GetLength(); j++)
    if(ww[j] != prevw[j]) break;
   if(next)
    ln.Cat(j);
   RLOG(j << "\t" << w[i]);
   next = true;
   const char *s = ~ww + j;
   while(*s) {
    for(int i = 0; i < voc.GetCount(); i++) {
     if(memcmp(s, voc[i], voc[i].GetLength()) == 0) {
      RLOG("  " << s << " " << voc[i]);
      ln.Cat(i + dict);
```

```
      s += voc[i].GetLength();
      break;
     }
    }
    ASSERT(i < voc.GetCount());
   }
   prevw = ww;
   i++;
  }
  RLOGHEXDUMP(ln, ln.GetLength());
 }
 int l = 0;
 for(i = 0; i < line.GetCount(); i++) {
  line[i].Cat(0);
  l += line[i].GetLength();
 }
 FileOut out("F:/dict/x.spell");
 out.Put(GetDefaultCharset());
 out.Put(0);
 out.Put(dict);
 for(i = 0; i < voc.GetCount(); i++) {
  out.Put(voc[i]);
  out.Put(0);
 }
 for(i = 0; i < line.GetCount(); i++) {
  out.PutL(line.GetKey(i));
  out.PutL(line[i].GetLength());
  out.Put(line[i]);
 }
}

void Main()
{
 Make();
}
```

(I have not even tried to compile that yet).

The input file is specified here:

```
void Make()
{
 FileIn in("f:/dict/cs_cz.txt");
```

and it should be one word per line, all possible variants. The code compresses it to .scd format.