Subject: CsvComparator application
Posted by Didier on Fri, 06 Aug 2010 09:10:57 GMT
View Forum Message <> Reply to Message

Hi all,

I just made a teaser that uses the TCC lib to compare to cvs files by using a dedicated 'C' comparison code
==> This enables all comparison techniques imaginable and not only the ones available in exel or other.

To launch the app, just compile it and run from console:
CsvComparator ref.csv csvComparaisonCode.h result.csv

The files ref.csv and result.csv are compared using 'csvComparaisonCode.h'.

In the display you can see the columns beeing compared are in black while others , which are not being compared are in light gray.
When a cell is in default ==> the difference is displayed in red

How the comparison file is build:

```
// header data: must always be there
typedef enum {
 CDTE_NOT_PROCESSED= 0,
 CDTE_UINT        = 1,
 CDTE_INT         = 2,
 CDTE_FLOAT       = 3,
 CDTE_STRING      = 4,
 CDTE_BOOL        = 5
} ColumnDataTypeEnum;


typedef unsigned long long CDT_UINT;
typedef signed long long   CDT_INT;
typedef double             CDT_FLOAT;
typedef const char*        CDT_STRING;
typedef bool               CDT_BOOL;


// enables to set the line and column separators used in the csv files
// if this function is not present, the default values are 'space' and 'tab'
void getParsingSeparators(char* lineSeparator,  char* columnSeparator)
{
 *lineSeparator='\n';
 *columnSeparator=' ';
}
```

```
// this function enables to set the columns beeing processed (== compared)
// by setting their data type
ColumnDataTypeEnum getDataType(int colNbr)
{
 ColumnDataTypeEnum res = CDTE_NOT_PROCESSED;
 switch(colNbr)
 {
  case 0: res = CDTE_INT; break;
  case 1: res = CDTE_STRING; break;
  case 2: res = CDTE_FLOAT; break;
  case 3: res = CDTE_FLOAT; break;
  case 7: res = CDTE_STRING; break;
 }
 return res;
}



// For each column that needs to be processed, there is a corresponding testcol_xxx  function
// * The 'xxx' value is the same as in the 'switch case' of the 'getDataType' function
// * The INPUT DATA TYPES of the comparison functions corresponds to the type defined by
'getDataType'
// * The other parameters 'reslutStrings' and 'refStrings' allow a comparison function to access
data on other
//   columns in text form
bool testCol_0(int curRow, CDT_INT* col, CDT_INT* refCol, const char*** reslutStrings, const
char*** refStrings)
{
 return (col[curRow] == refCol[curRow]);
}

bool testCol_1(int curRow, CDT_STRING* col, CDT_STRING* refCol, const char*** reslutStrings,
const char*** refStrings)
{
 return ( strcmp(col[curRow], refCol[curRow]) == 0 );
}

bool testCol_2(int curRow, CDT_FLOAT* col, CDT_FLOAT* refCol, const char*** reslutStrings,
const char*** refStrings)
{
 return ((col[curRow] == refCol[curRow]) || (col[curRow] == -refCol[curRow]));
}

bool testCol_3(int curRow, CDT_FLOAT* col, CDT_FLOAT* refCol, const char*** reslutStrings,
const char*** refStrings)
{
```

```
 return (col[curRow] == refCol[curRow]);
}

bool testCol_7(int curRow, CDT_STRING* col, CDT_STRING* refCol, const char*** reslutStrings,
const char*** refStrings)
{
 return ( strcmp(col[curRow], refCol[curRow]) == 0 );
}
```

This is just a programm I made for fun(and therefore is far from beeing perfect), but the use of Tcc is quite interesting.
Maybe some of you might find this useful.

NB: it compiles under windows and linux, but works only on linux du to include/Tcc problems. I didn't take time to look into this problem.

File Attachments

1) CsvComparator.tar.gz, downloaded 491 times