
Subject: again... use if deleted function :?

Posted by [idkfa46](#) on Sun, 14 Jan 2018 11:10:53 GMT

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

Hi guys,

I'm here asking for your help again because I have one more error compiling my code.

Error:

```
C:\Users\Matteo\Dropbox\2_Sviluppo++\Workspace_upp\LibVerifiche\Tensoflessione.cpp (339):
error: use of deleted function 'QTFStr::QTFStr(const QTFStr&)'
    (): return q tf ;
C:\Users\Matteo\Dropbox\2_Sviluppo++\Workspace_upp\LibQTF/QTFStr.h (13): note:
'QTFStr::QTFStr(const QTFStr&)' is implicitly deleted because the default definition would be
ill-formed:
    (): class Q TFStr
C:\Users\Matteo\Dropbox\2_Sviluppo++\Workspace_upp\LibQTF/QTFStr.h (13): error: use of
deleted function 'constexpr Upp::Vector<QTFStr::QTFFormat>::Vector(const
Upp::Vector<QTFStr::QTFFormat>&)'
C:\Users\Matteo\Dropbox\2_Sviluppo++\Workspace_upp\LibQTF/QTFStr.h (13): error: use of
deleted function 'constexpr Upp::Vector<int>::Vector(const Upp::Vector<int>&)'
C:\Users\Matteo\Dropbox\2_Sviluppo++\Workspace_upp\LibQTF/QTFStr.h (13): error: use of
deleted function 'constexpr Upp::Vector<int>::Vector(const Upp::Vector<int>&)'
C:\Users\Matteo\Dropbox\2_Sviluppo++\Workspace_upp\LibQTF/QTFStr.h (13): error: use of
deleted function 'constexpr Upp::Vector<Upp::Vector<int> >::Vector(const
Upp::Vector<Upp::Vector<int> >&)'
C:\Users\Matteo\Dropbox\2_Sviluppo++\Workspace_upp\LibVerifiche\Tensoflessione.cpp (423):
error: use of deleted function 'QTFStr::QTFStr(const QTFStr&)'
C:\Users\Matteo\Dropbox\2_Sviluppo++\Workspace_upp\LibVerifiche\Flessione.cpp (409): error:
use of deleted function 'QTFStr::QTFStr(const QTFStr&)'
C:\Users\Matteo\Dropbox\2_Sviluppo++\Workspace_upp\LibQTF/QTFStr.h (13): error: use of
deleted function 'constexpr Upp::Vector<QTFStr::QTFFormat>::Vector(const
Upp::Vector<QTFStr::QTFFormat>&)'
C:\Users\Matteo\Dropbox\2_Sviluppo++\Workspace_upp\LibQTF/QTFStr.h (13): error: use of
deleted function 'constexpr Upp::Vector<int>::Vector(const Upp::Vector<int>&)'
C:\Users\Matteo\Dropbox\2_Sviluppo++\Workspace_upp\LibQTF/QTFStr.h (13): error: use of
deleted function 'constexpr Upp::Vector<int>::Vector(const Upp::Vector<int>&)'
C:\Users\Matteo\Dropbox\2_Sviluppo++\Workspace_upp\LibQTF/QTFStr.h (13): error: use of
deleted function 'constexpr Upp::Vector<Upp::Vector<int> >::Vector(const
Upp::Vector<Upp::Vector<int> >&)'
C:\Users\Matteo\Dropbox\2_Sviluppo++\Workspace_upp\LibVerifiche\Flessione.cpp (508): error:
use of deleted function 'QTFStr::QTFStr(const QTFStr&)'
```

function:

```
QTFStr Tensoflessione::Instabilitaqtf(QTFStr &qtf)
{
```

```

String risultato;

<= 1 &";

<= 1 ";
risultato << GetVerificaInst();

qtf.StartTable(3,2,2,10).CharSize(12);
qtf.TableSubTitle("Instabilità flesso-torsionale");
qtf.LeftCellMargin(75)("I[, eff]").LeftCellMargin(15).AlignRight()
  (Format("%.2f", GetInstab().leff)).AlignLeft("mm")("Lunghezza libera di inflessione");

  (Format("%.2f", GetInstab().sigmam_critico)).AlignLeft("")("Tensione critica di svergolamento");

  (Format("%.2f", GetInstab().lambda_relm)).AlignLeft("")("Snellezza relativa di svergolamento");
qtf.LeftCellMargin(75)("k[, crit]").LeftCellMargin(15).AlignRight()
  (Format("%.2f", GetInstab().kcrit)).AlignLeft("N/mm[2"]("Coefficiente di sbandamento
laterale");
qtf.TableSubTitle(risultato);
qtf.EndTable();

qtf.TableSubTitle(risultato);
qtf.EndTable();

return qtf;
}

```

QTFStr class:

```

class QTFStr
{
public:
  class QTFFormat : public Moveable<QTFFormat>
  {
  public:
    int tableKeep;
    bool tableBorder;
    bool bold;
    bool strike;
    bool underline;
    bool subscript;
    bool superscript;
    bool italic;
    int align;
    int charSize;
    char font;
    int bottomCellMargin;
  };
};

```

```

int topCellMargin;
int leftCellMargin;
int rightCellMargin;
String foreColor;
String backColor;
String cellbackground;

QTFFormat();
void Clear(void);
QTFFormat(const QTFFormat &fmt);
QTFFormat const &operator=(QTFFormat const &fmt);
};

```

private:

```

// QTF string container
String str;

// settings stack
Vector<QTFFormat> formatStack;

// table depth level
int tableLevel;

// table running values
Vector<int>  tableColumns;
Vector<int>  tableColumn;
Vector<Vector<int> > columnWidths;;

// formatting running values
QTFFormat format;

// error flag -- stops any processing if set
bool err;

// outputs format codearound string
String FormatString(const String &s);

```

protected:

public:

```

// constructor
QTFStr();

// empty string and reset all formats to default
QTFStr &Clear(void);

```

```

// clears all formatting options -- reset them to defaults
QTFStr &ClearFormats(void);

// push/pop format values
QTFStr &PushFormat(void);
QTFStr &PopFormat(void);

// set/get format
QTFStr &GetFormat(QTFFormat &fmt) { fmt = format; return *this; }
QTFStr &SetFormat(QTFFormat const &fmt) { format = fmt; return *this; }

// checks condition, if true sets error value, logs the (first) error with its QTF
// and stops any further processing. Returns err flag
bool CheckError(bool cond, String const &msg);

// format change functions
QTFStr &TableOnPage(void) { format.tableKeep = KEEP_TABLE_ON_PAGE; return *this; }
QTFStr &CellOnPage(void) { format.tableKeep = KEEP_CELL_ON_PAGE; return *this; }
QTFStr &FreeTable(void) { format.tableKeep = FREE_TABLE; return *this; }
QTFStr &TableBorder(void) { format.tableBorder = true; return *this; }
QTFStr &NoTableBorder(void) { format.tableBorder = false; return *this; }
QTFStr &Bold(void) { format.bold = true; return *this; }
QTFStr &NoBold(void) { format.bold = false; return *this; }
QTFStr &Strike(void) { format.strike = true; return *this; }
QTFStr &NoStrike(void) { format.strike = false; return *this; }
QTFStr &Underline(void) { format.underline = true; return *this; }
QTFStr &NoUnderline(void) { format.underline = false; return *this; }
QTFStr &Subscript(void) { format.subscript = true; return *this; }
QTFStr &NoSubscript(void) { format.subscript = false; return *this; }
QTFStr &SuperScript(void) { format.superscript = true; return *this; }
QTFStr &NoSuperScript(void) { format.superscript = false; return *this; }
QTFStr &Italic(void) { format.italic = true; return *this; }
QTFStr &NoItalic(void) { format.italic = false; return *this; }
QTFStr &AlignLeft(void) { format.align = ALIGN_LEFT; return *this; }
QTFStr &AlignCenter(void) { format.align = ALIGN_CENTER; return *this; }
QTFStr &AlignRight(void) { format.align = ALIGN_RIGHT; return *this; }
QTFStr &AlignJustify(void) { format.align = ALIGN_JUSTIFY; return *this; }
QTFStr &CharSize(int siz) { format.charSize = siz; return *this; }
QTFStr &IncCharSize() { if (format.charSize < 9) format.charSize++; return *this; }
QTFStr &DecCharSize() { if (format.charSize > 0) format.charSize--; return *this; }
QTFStr &Font(char fnt) { format.font = fnt; return *this; }
QTFStr &ForeColor(String col = "") { format.foreColor = (col == "" ? "0" : col); return *this; }
QTFStr &BackColor(String col = "") { format.backColor = (col == "" ? "2" : col); return *this; }
//
QTFStr &BackgroundColor(String col = ""){ format.cellbackground = (col == "" ? "2" : col); return
*this; }
//
QTFStr &LeftCellMargin(int m = 25) { format.leftCellMargin = m; return *this; }

```

```

QTFStr &RightCellMargin(int m = 25) { format.topCellMargin = m; return *this; }
QTFStr &TopCellMargin(int m = 15) { format.rightCellMargin = m; return *this; }
QTFStr &BottomCellMargin(int m = 15) { format.bottomCellMargin = m; return *this; }

```

```

// output text
QTFStr &Txt(const String &s, int merge = 0);
QTFStr &Txt(RichObject const &obj, int merge = 0);
QTFStr &operator()(const String &s, int merge = 0) { return Txt(s, merge); }
QTFStr &operator()(RichObject const &obj, int merge = 0) { return Txt(obj, merge); }

```

```

QTFStr &Iml(String const &Name, int cx, int cy, bool keepAspect = true, int merge = 0);
QTFStr &CenterIml(String const &Name, int cx, int cy, bool keepAspect = true, int merge = 0);

```

```

QTFStr &Object(RichObject const &obj, int merge = 0);
QTFStr &CenterObject(RichObject const &obj, int merge = 0);

```

```

// page title
QTFStr &PageTitle(String const &s1, String const &s2);

```

```

/////////////////////////////////////////////////////////////////
//                                TABLE HANDLING                                //
/////////////////////////////////////////////////////////////////
QTFStr &StartTable(Vector<int> const &colW);
QTFStr &StartTable(
    int w1 = Null, int w2 = Null, int w3 = Null, int w4 = Null,
    int w5 = Null, int w6 = Null, int w7 = Null, int w8 = Null);
QTFStr &TableRestart(void);
QTFStr &Cell(String const &Txt, int merge = 0);
QTFStr &TableTitle(String const &title);
QTFStr &TableSubTitle(String const &title);
QTFStr &TableNewLine(void);
QTFStr &EndTable(void);

operator const char *() { return str; }
operator const String&() { return str; }
String const &ToString(void) const { return str; }

}; // END Class QTFStr

```

The move constructor is implicitly deleted because the copy-constructor is explicitly deleted?

How can I easy fix it?

Regards,
Matteo
