Classe LogParser

LogParser.h

```
/*************************************************************************
LogParser  -  fichier d'en-tête de la classe <LogParser>
-------------------
debut      : 27/11/2015 22:28:31
copyright  : (C) 2015 par Quentin SCHROTER, Nicolas GRIPONT
e-mail     : quentin.schroter@insa-lyon.fr , nicolas.gripont@insa-lyon.fr
*************************************************************************/

//---------- Interface de la classe <LogParser> (fichier LogParser.h) ----
#if ! defined ( LogParser_H )
#define LogParser_H

//--------------------------------------------------- Interfaces utilisees

#include "LogLine.h"
#include "Date.h"
//------------------------------------------------------------- Constantes

//------------------------------------------------------------------ Types

//--------------------------------------------------------------------------
// Rôle de la classe <LogParser> : Classe permettant de creer des objets
// de type LogLine. Cette classe ne contient qu?une methode de classe
// (static), nous avons donc interdit de construire un objet de ce type.
//--------------------------------------------------------------------------

class LogParser
{
//----------------------------------------------------------------- PUBLIC

public:
//---------------------------------------------------- Methodes publiques
   static LogLine Parser (string line , string domainName);
         // Mode d'emploi :
   // Methode permettant de creer un  objet de type LogLine à partir d'une
   // string contenant une ligne de log d'un fichier de log Apache et
   // à partir du nom de domaine du serveur apache générant le fichier
   // de logs.
         //
         // Contrat :
   // Il faut que la ligne donnée en paramètre soit
   // valide. Sinon l'objet LogLine retourné ne sera pas valide.
         //


   static string Extension ( string cheminAcces );
   // Mode d'emploi :
   // Methode de classe permettant de recuperer l'extension d'un fichier
   // a partir d?une string contenant son chemin d?acces.
   // Contrat : aucun.
```

```
    //
    //

//------------------------------------------------- Surcharge d'operateurs
    LogParser & operator = ( const LogParser & unLogParser );
    // Mode d'emploi :
    // Operateur "=" interdit.
            //
    // Contrat :
    // Aucun.
            //


//-------------------------------------------- Constructeurs - destructeur
    LogParser ( const LogParser & unLogParser );
            // Mode d'emploi (constructeur de copie) :
    // Constructeur de copie interdit.
    //
    // Contrat :
    // Aucun.
            //

    LogParser ( );
            // Mode d'emploi :
    // Constructeur par defaut interdit.
    //
    // Contrat :
    // Aucun.
            //

    virtual ~LogParser ( );
            // Mode d'emploi :
    // Destructeur interdit.
    //
            // Contrat :
            //

//------------------------------------------------------------------ PRIVE

protected:
//----------------------------------------------------- Methodes protegees

    static Date ParserDate ( string date );
    // Mode d'emploi :
    // Methode permettant de creer un objet de type Date à partir d'une
    // string contenant le champs date d'une ligne de log d'un
    // fichier de log Apache.
    //
    // Contrat : aucun.
    //
    //

    static int Mois ( string mois );
    // Mode d'emploi :
    // Methode permettant de convertir un mois anglais
    // (sur 3 lettres, ex :Sep) en int
```

```
    //
    // Contrat : aucun.
    //
    //

    static string SplitReferer ( string referer, string domainName );
    // Mode d'emploi :
    // Methode permettant de recuperer le fichier d'une url contenant
    // le nom de domaine donne en parametre. Retourne "-" si l'url
    // (referer) ne contient pas le nom de domain.
    //
    // Contrat : aucun.
    //
    //


//------------------------------------------------------ Attributs proteges

};

//---------------------------- Autres definitions dependantes de <LogParser>

#endif // LogParser_H
```

LogParser.cpp

```
/*************************************************************************
LogParser  -  fichier de realisation de la classe <LogParser>
-------------------
debut      : 27/11/2015 22:28:52
copyright  : (C) 2015 par Quentin SCHROTER, Nicolas GRIPONT
e-mail     : quentin.schroter@insa-lyon.fr , nicolas.gripont@insa-lyon.fr
*************************************************************************/

//-------- Realisation de la classe <LogParser> (fichier LogParser.cpp) --

//---------------------------------------------------------------- INCLUDE

//-------------------------------------------------------- Include système

#include <sstream>
#include <iostream>
#include <string>
#include <regex>

using namespace std;
//------------------------------------------------------ Include personnel
#include "LogParser.h"

//------------------------------------------------------------ Constantes
```

```
//---------------------------------------------------------------- PUBLIC

//---------------------------------------------------- Methodes publiques
LogLine LogParser::Parser( string line, string domainName)
// Algorithme :
//
{
    LogLine logLine;
    string tmp;
    char separateur;

    istringstream iss(line);

    separateur = ' ';
    getline(iss,tmp,separateur);
    logLine.clientIP = tmp;

    getline(iss,tmp,separateur);
    logLine.userLogName = tmp;

    getline(iss,tmp,separateur);
    logLine.authenticatedUser = tmp;

    separateur = '[';
    getline(iss,tmp,separateur);
    separateur = ']';
    getline(iss,tmp,separateur);
    logLine.date = ParserDate(tmp);

    separateur = '"';
    getline(iss,tmp,separateur);
    separateur = ' ';
    getline(iss,tmp,separateur);
    logLine.actionType = tmp;

    getline(iss,tmp,separateur);
    if ( tmp[tmp.length()-1] == '/' && tmp.length() > 1 )
    {
        tmp = tmp.substr(0,tmp.length()-1);
    }
    logLine.requestedURL = tmp;

    separateur = '"';
    getline(iss,tmp,separateur);
    logLine.httpVersion = tmp;

    separateur = ' ';
    getline(iss,tmp,separateur);
    getline(iss,tmp,separateur);
    logLine.status = stoi(tmp);

    getline(iss,tmp,separateur);
    if (tmp != "-")
    {
        logLine.bytesNumber = stoi(tmp);
    }
```

```
    else
    {
        logLine.bytesNumber = 0;
    }

    separateur = "";
    getline(iss,tmp,separateur);
    getline(iss,tmp,separateur);
    logLine.domainName = domainName;
    tmp = SplitReferer (tmp,  domainName);
    if ( tmp[tmp.length()-1] == '/' && tmp.length() > 1 )
    {
        tmp = tmp.substr(0,tmp.length()-1);
    }
    logLine.sourceFile = tmp;

    getline(iss,tmp,separateur);
    getline(iss,tmp,separateur);
    logLine.navigator = tmp;

    return logLine;
} //----- Fin de Parser


string LogParser::Extension ( string cheminAcces )
{
    string s = "";
    int i = cheminAcces.length() - 1;
    while (i > 0 && cheminAcces[i] != '.')
    {
        s += cheminAcces[i];
        i--;
    }
    s += '.';
    reverse(s.begin(), s.end());

    if ( s == cheminAcces )
    {
        s = "";
    }

    return s;
}

//---------------------------------------------------- Surcharge d'operateurs


//-------------------------------------------- Constructeurs - destructeur



//------------------------------------------------------------------ PRIVE

//----------------------------------------------------- Methodes protegees

Date LogParser::ParserDate ( string date )
```

```
// Algorithme :
//
{
    int annee = 0;
    int mois = 0;
    int jour = 0;
    int heure = 0;
    int minute = 0;
    int seconde = 0;
    int gmt = 0;

    regex rgx("([0-9]{2})/([A-Z][a-z][a-z])/([0-9]{4})[ ]?:([0-9]{2}):([0-9]{2}):([0-9]{2})[ ]([+-][0-9]{4})");
    smatch match;
    if (std::regex_search(date, match, rgx))
    {
        jour = stoi(match[1]);
        mois = Mois(match[2]);
        annee = stoi(match[3]);
        heure = stoi(match[4]);
        minute = stoi(match[5]);
        seconde = stoi(match[6]);
        gmt = stoi(match[7]);
    }

    return Date(annee,mois,jour,heure,minute,seconde,gmt);
} //----- Fin de ParserDate



int LogParser::Mois ( string mois )
// Algorithme :
//
{
    int m = -1;


    if ( mois == "Jan" )
    {
        m = 1;
    }
    else if ( mois == "Feb" )
    {
        m = 2;
    }
    else if ( mois == "Mar" )
    {
        m = 3;
    }
    else if ( mois == "Apr" )
    {
        m = 4;
    }
    else if ( mois == "May" )
    {
        m = 5;
    }
```

```
    else if ( mois == "Jui" )
    {
        m = 6;
    }
    else if ( mois == "Jul" )
    {
        m = 7;
    }
    else if ( mois == "Aug" )
    {
        m = 8;
    }
    else if ( mois == "Sep" )
    {
        m = 9;
    }
    else if ( mois == "Oct" )
    {
        m = 10;
    }
    else if ( mois == "Nov" )
    {
        m = 11;
    }
    else if ( mois == "Dec" )
    {
        m = 12;
    }

    return m;
} //----- Fin de Mois


string LogParser::SplitReferer ( string referer, string domainName )
// Algorithme :
//
{
    string result;

//    regex r("([a-zA-Z]+://)([a-zA-Z0-9.]+)(/[a-zA-Z0-9./]+)")
//    match[1] = http://
//    match[2]= nomDomaine
//    match[3]=cheminAcces

//    string tmpRgx = "https?://";
//    tmpRgx += domainName;
//    tmpRgx += ("(.*)");

//    regex rgx(tmpRgx);
//    smatch match;

//    if (regex_search(referer, match, rgx))
//    {
//        result = match[1];
//    }
//    else
```

```
//    {
//        result = "-";
//    }

    string tmp = referer.substr(0,domainName.length());
    if ( tmp == domainName)
    {
        result = referer.substr(domainName.length(),referer.length());
    }
    else
    {
        result = "-";
    }

    return result;
} //----- Fin de SplitReferer
```