```
/*
 * Classe spécifique à la gestion de l'affichage de la barrière
 */

#ifndef _AFFICHEUR_H
#define _AFFICHEUR_H

#include <Socket.h>
#include <string.h>

//-------------------------------------------------------------
class Afficheur
{
  Socket & barriere;
  char ligneUn[17];
  char ligneDeux[17];
  struct { int x; int y; } curseur;
public:
  Afficheur(Socket &service);
  void clear();
  void home();
  void setCursor(bool visible);
  void positionCursor(int x, int y);
  void affiche(int x, int y, char *message);
  void afficheCaractere(char caractere);
  void afficheMessages(char *premier, char *deuxieme);
  void affiche(char *premier, char *deuxieme);
  void centre(char *premier, char *deuxieme);
  char* getLigneUn();
  char* getLigneDeux();
private:
  void effacerLignes();
};
//-------------------------------------------------------------

// Constructeur : connexion avec le service et mise à zéro
// des deux lignes d'affichage;
inline
Afficheur::Afficheur(Socket &service) : barriere(service)
{
  effacerLignes();
  char invite[] = "EN SERVICE";
  for (int i=0; invite[i]!='\0'; i++) ligneUn[i] = invite[i];
}

// Efface le contenu des deux lignes de l'afficheur
inline
void Afficheur::clear()
{
  barriere.envoyer("Afficheur:clear");
  barriere.recevoir();
  effacerLignes();
  curseur.x = 0;
  curseur.y = 0;
}

// Place le curseur en haut à gauche de l'afficheur
inline
void Afficheur::home()
{
  barriere.envoyer("Afficheur:home");
```

```
  barriere.recevoir();
  curseur.x = 0;
  curseur.y = 0;
}

// Rendre visible ou pas le curseur de l'afficheur
inline
void Afficheur::setCursor(bool visible)
{
  char chaine[30];
  strcpy(chaine, "Afficheur:setCursor:");
  strcat(chaine, (visible ? "true" : "false"));
  barriere.envoyer(chaine);
  barriere.recevoir();
}

// Place le curseur de l'afficheur à l'endroit spécifié
inline
void Afficheur::positionCursor(int x, int y)
{
  char chaine[40];
  sprintf(chaine, "Afficheur:positionCursor:%d:%d", x, y);
  barriere.envoyer(chaine);
  barriere.recevoir();
  curseur.x = x;
  curseur.y = y;
}

// Affiche un message aux coordonnées spécifiées
inline
void Afficheur::affiche(int x, int y, char *message)
{
  char chaine[40];
  sprintf(chaine, "Afficheur:affiche:%d:%d:%s", x, y, message);
  barriere.envoyer(chaine);
  barriere.recevoir();
  int longueur = strlen(message);
  if (y==0) for (int i=0; i<longueur; i++) ligneUn[i+x] = message[i];
  if (y==1) for (int i=0; i<longueur; i++) ligneDeux[i+x] = message[i];
  curseur.x = x + longueur;
  curseur.y = y;
}

// Affiche un caractère à l'endroit où se trouve le curseur de l'afficheur
inline
void Afficheur::afficheCaractere(char caractere)
{
  char chaine[40];
  sprintf(chaine, "Afficheur:afficheCaractere:%c", caractere);
  barriere.envoyer(chaine);
  barriere.recevoir();
  if (curseur.y==0) ligneUn[curseur.x++] = caractere;
  if (curseur.y==1) ligneDeux[curseur.x++] = caractere;
}

// Affiche deux messages sur chacune des lignes de l'afficheur
inline
void Afficheur::afficheMessages(char *premier, char *deuxieme)
{
  char chaine[80];
  sprintf(chaine, "Afficheur:afficheMessages:%s:%s", premier, deuxieme);
```

```
  barriere.envoyer(chaine);
  barriere.recevoir();
  effacerLignes();
  for (int i=0; premier[i]!='\0'; i++) ligneUn[i] = premier[i];
  for (int i=0; deuxieme[i]!='\0'; i++) ligneDeux[i] = deuxieme[i];
  curseur.y = 1;
  curseur.x = strlen(deuxieme);
}

// Affiche deux messages avec un effacement préliminaire
inline
void Afficheur::affiche(char *premier, char *deuxieme)
{
  clear();
  afficheMessages(premier, deuxieme);
}

// Affiche deux messages centrés avec un effacement préliminaire
inline
void Afficheur::centre(char *premier, char *deuxieme)
{
  clear();
  int position = (16 - strlen(premier)) / 2;
  affiche(position, 0, premier);
  position = (16 - strlen(deuxieme)) / 2;
  affiche(position, 1, deuxieme);
}


// Retourne le contenu de la première ligne de l'afficheur
inline
char* Afficheur::getLigneUn()
{
  return ligneUn;
}

// Retourne le contenu de la deuxième ligne de l'afficheur
inline
char* Afficheur::getLigneDeux()
{
  return ligneDeux;
}

// Effacer le contenu des deux lignes d'affichage
inline
void Afficheur::effacerLignes()
{
  for (int i=0; i<16; i++)
  {
    ligneUn[i] = '*';
    ligneDeux[i] = '*';
  }
  ligneUn[16] = '\0';
  ligneDeux[16] = '\0';
}

#endif
```