Cours4 - C++

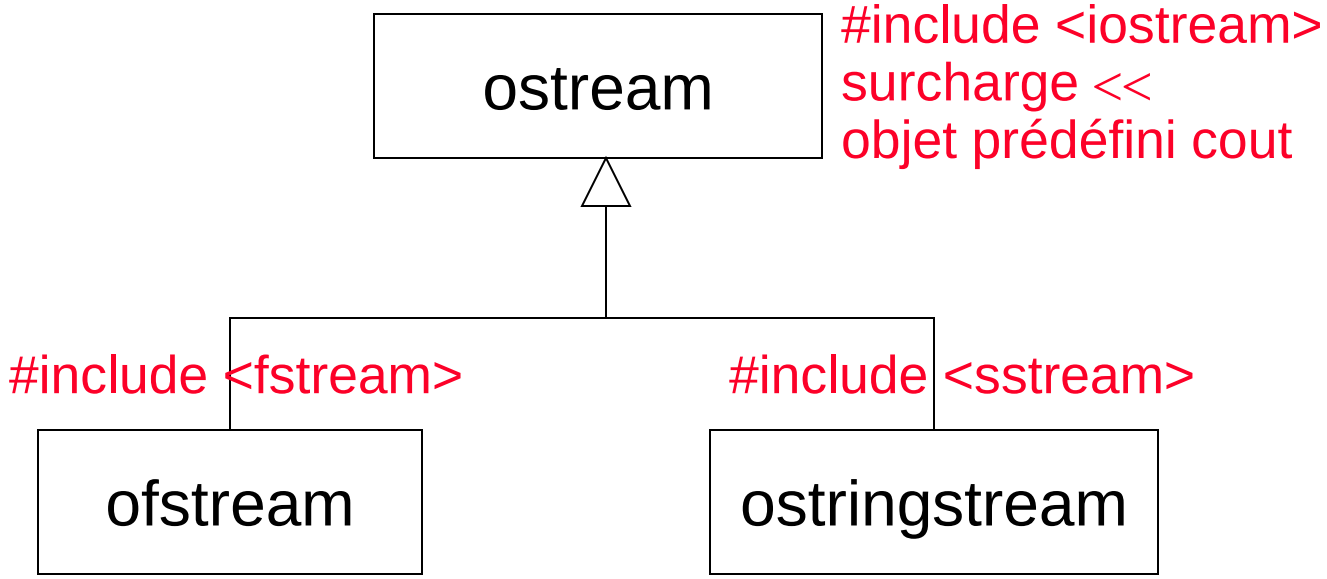
classe string / classes mères ostream et istream / I/O fichier / formatage en mémoire

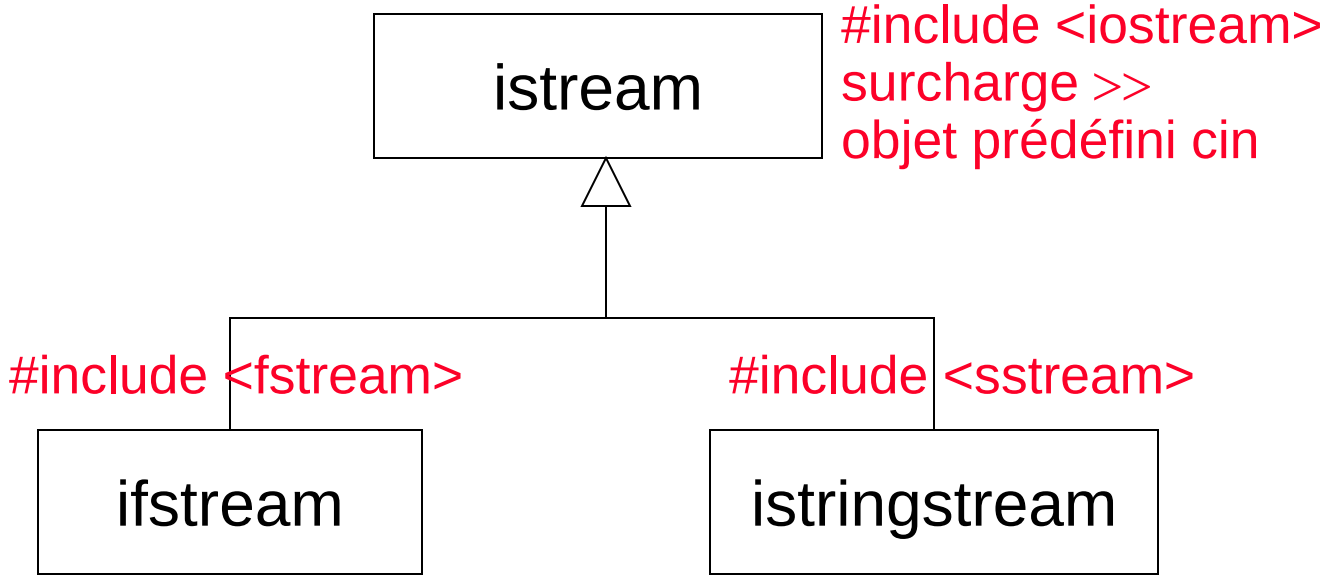
J-F. Kamp

Janvier 2025

Introduction

Flots entrée et sortie : arborescence d'héritage





Classe ostream

Surcharge de <<

- Dans classe ostream, fonction membre :

  ostream& operator<< (type var)

  Exemple :
  cout << n; // 2 opérandes : cout et n

  qui est modifié ? cout
  qui reste inchangé ? n

  => fct mbre operator<< s'applique à cout :

  - prend en paramètre n (type bool, int, char, double, char*, string, ...)

  - renvoie type ostream&

Classe istream

Surcharge de >>

- Dans classe istream, fonction membre :

  istream& operator>> (type& var)

  Exemple :
  cin >> n; // 2 opérandes : cin et n

  => fct mbre operator>> s'applique à cin :

  - prend en paramètre var (type bool, int, char, double, char*, string, ...) passé par référence
  - renvoie type istream&
  - délimiteurs : espace, '\t', '\v', '\n', '\f'

- Un délimiteur n'est pas un caractère lu
  Cas char* : char* pChar;
  cin >> pChar;
  => impossible de stocker une phrase avec délimiteurs "bonzour info2" devient "bonzour"

Classe string

Classe string

- En C++, la classe string permet aussi de manipuler une chaîne de caractères (nécessite <string>)

- Similitudes avec String de Java : une classe complète pour la manipulation de chaînes de caractères
  www.cppreference.com/cppstring/

- En C++, string est une structure de type vector<char> où la notion de caractère de fin '\0' n'existe plus. Dès lors :

  - accès à un caractère : char& at(int ind) ou [ind]
  - taille de la chaine : int size() ou int length()
  - parcours possible avec iterator

Classe string : constructeurs

- string st1; // chaîne vide, size = 0
- string st2("Bonjour"); // Init. avec chaîne const.
- string st3 (st2); // Init. avec autre chaîne
- Conversion char* => string :
  - const char* pTab = "Hello";
  - string st4 (pTab);
- string st5 (10, '*'); // Chaîne de taille 10 fois *
- !! Attention, contrairement à Java :

```
int a = 102;
double b = 5.56;
string st ( "blabla" + a + b + "fin" ); // NON !!
Java : String st = new String ( "blabla" + a );
```

Classe string : opérateurs

Un certain nombre d'opérateurs sont *surchargés* par la classe string :

- st1 = st2 // affectation entre 2 string
- st1 = pChar // affectat. char* => string
- st1 == st2 // comp. entre 2 string
- idem !=, >, >=, <, <=
- st3 = st1 + st2 // concaténation de string
- st3 = st1 + pChar // concat. string + char*
- st3 = st1 + unCar // concat. string + char
- cout << st1 << endl;
- cin >> st2;

Classe string : méthodes utiles

- Retourne le caractère à l'indice indiqué :
  char& at (int ind)

- Conversion string => char* :
  const char* c_str()

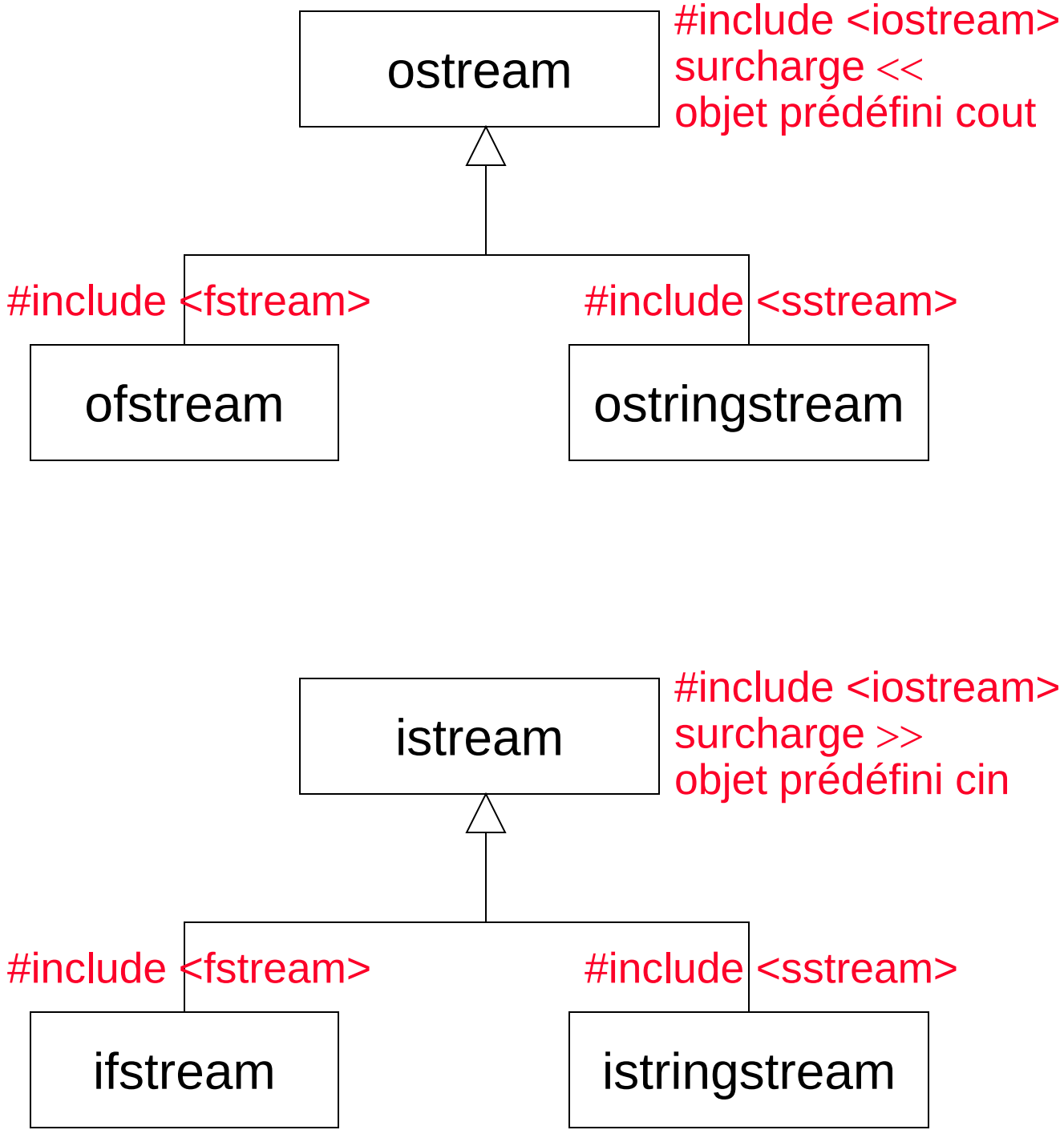
- Lecture à partir d'un flux en entrée (!! fonction qui ne fait pas partie de la classe string) :
  istream& getline (istream& in, string& st, char delim = '\n')

- Nombre de caractères dans la chaîne :
  unsigned int length() ou unsigned int size()

- Autres : string substr(...), string& erase(...)

Formatage d’une chaîne de caractères

Flots entrée et sortie : arborescence d'héritage



Classe ostringstream

- Rôle : manipuler une chaîne de caractères (string) comme un flux de caractères et formaté ce flux de caractères grâce à l'opérateur <<

- Exemple :

```
ostringstream tmp;
string maCh;
tmp << type primitif << délimiteur ('\n') << ...
maCh = tmp.str();
```

Classe istringstream

- Rôle : à partir d'un tableau de caractères ou d'une chaîne (string), cette classe permet d'éclater le contenu dans différents types primitifs (bool, int, float, char, char*) et type string grâce à l'opérateur >> (cf. StringTokenizer en Java)

- Exemple :

```
string ch = "123      45.2          salut";
istringstream tmp ( ch );
int n;
float x;
string s;
tmp >> n >> x >> s;
// Après cette instruction :
// n contient 123
// x contient 45.2
// s contient "salut"
```

I/O fichier

La classe ofstream

- La classe ofstream dérive de la classe ostream :

```
#include <iostream>
#include <fstream>
#include <string>        // éventuellement
```

rôle : permet d'écrire dans un flot de sortie connecté à un fichier.

- Constructeur de la classe :

```
ofstream ( const char* pChar, ios::openmode )
```

2 paramètres :

- pChar : nom du fichier
- ios::openmode : constante mode d'ouverture (ios::out par défaut)

La classe ofstream

- Une instance de ofstream a forcément accès aux méthodes décritent dans ostream :
  - la surcharge de <<
- Exemple d'utilisation de l'opérateur << pour l'écriture dans un fichier **texte**

```
const char* nomFich = "fichierTxt.txt";
ofstream fluxO ( nomFich, ios::out );
string maChaine;

maChaine = "Un texte bien long…";

// écriture dans le fichier texte "fichierTxt.txt"
fluxO << maChaine << endl;

fluxO.close();
```

La classe ifstream

- La classe ifstream dérive de la classe istream :

```
#include <iostream>
#include <fstream>
#include <string>        // éventuellement
```

rôle : permet de lire à partir d'un flot d'entrée connecté à un fichier.

- Constructeur de la classe :

```
ifstream ( const char* pChar, ios::openmode )
```

2 paramètres :

- pChar : nom du fichier
- ios::openmode : constante mode d'ouverture (ios::in par défaut)

La classe ifstream

- Une instance de ifstream a forcément accès aux méthodes décrites dans istream :
  - la surcharge de >>
- La fonction istream& getline (istream& in, string& st, char delim = '\n')

  permet de lire un fichier **texte**, exemple :

```
const char* nomFich = "fichierTxt.txt";
ifstream fluxIn ( nomFich, ios::in );
string maChaine;

// lecture du fichier ligne par ligne
while ( ! fluxIn.eof() ) {
      getline ( fluxIn, maChaine, ' \n ' );
}

fluxIn.close();
```