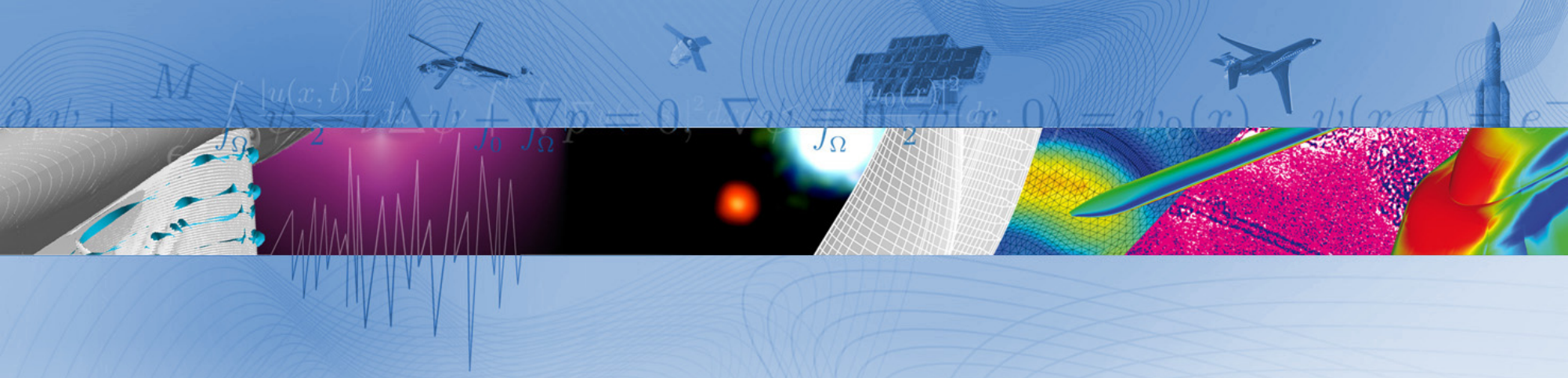

Introduction à la programmation C++

Divers

Alexandre Boulch

aboulch.github.io

ONERA
THE FRENCH AEROSPACE LAB

Plan de la séance

STL : vector

La STL

Standard Template Library

- Librairie disponible par défaut en C++
- De nombreux modules :
  - classes : chînes de caractères, tableaux, piles ...
  - algorithmes : tri, $n^{ième}$ éléments ...
  - lecture / écriture

Des modules connus :

- `iostream`
- `string`

vector

Une classe particulièrement intéressante de la STL est la classe `vector`.

- tableau dynamique
- pas de gestion de la mémoire
- interface type tableau

```
#include <vector>

using namespace std;
```

Classe template

La classe `vector` est une classe template, elle peut s'adapter à tous les types de données. (les templates seront évoqués dans les cours suivants).

```
// creation d'un vector
vector<T> tab;

// Exemple :
vector<int> t_int;
vector<double> t_double;
vector<Matrix> t_mat;
vector<float*> t_point;
```

vector : utilisation

Comme un tableau classique :

```
// creation d'un vector d'entier 100 cases
vector<int> t(100);

// acces au cases
for(int i=0; i<100; i++){
    t[i] = i*i;
}
cout << t[5] << endl;

// recuperer la taille
cout << t.size() << endl;
```

où l'on peut récupérer la taille.

vector : utilisation

- Redimensionner :

```
// creation d'un vector d'entier 100 cases
t.resize(1000);
```

Les éléments présents sont gardés.

- Création et remplissage :

```
// creation d'un vector d'entier 100 cases rempli avec 5.6
vector<double> t2(1000, 5.6);
```

vector : utilisation

- Premier élément :

```
cout << *t.begin() << endl;
```

C'est un iterateur (fonctionne un peu comme un pointeur)

- Fin de vector :

```
t.end(); // Attention pointe juste derriere la derniere case
```

vector : utilisation

- Concaténer :

```
vector<int> t1 (10,2);
vector<int> t2 (30, 100);
t2.insert(t2.end(), t1.begin(), t1.end());
```

C'est un iterateur (fonctionne un peu comme un pointeur)

- Trier :

```
#include <algorithm>

...

std::sort(t.begin(), t.end());
```