

Le type point en C++

Interface: point.hpp

```
1 #include <ostream>
2 #ifndef POINT_H
3 #define POINT_H
4
5 class Point {
6     static int count_; // Total amount of points
7     const int rank_;
8     double x_, y_;
9
10    friend std::ostream&
11    operator<<(std::ostream & out, Point const& p);
12
13    public:
14    Point(double x, double y);
15    ~Point() = default;
16
17    double x() const;
18    double y() const;
19    void move(double dx, double dy);
20
21    Point operator +(Point const& p2) const;
22    bool operator == (Point const& other) const;
23 };
24
25 std::ostream&
26 operator<<(std::ostream & out, Point const& p);
27
28 #endif /* POINT_H */
```

Implémentation: point.cpp

```
1 #include "point.hpp"
2
3 int Point::count_ = 0;
4
5 Point::Point(double x, double y)
6     : x_(x), y_(y), rank_(++count_) {
7 }
8 double Point::x() const {
9     return this->x_;
10 }
11 double Point::y() const {
12     return this->y_;
13 }
14 void Point::move(double dx, double dy) {
15     x_ += dx;
16     y_ += dy;
17 }
18 Point Point::operator +(Point const& p2) const{
19     return Point(x_ + p2.x_, y_+p2.y_);
20 }
21 bool Point::operator== (Point const& p2) const{
22     return x_==p2.x_ && y_==p2.y_;
23 }
24
25 std::ostream&
26 operator<<(std::ostream & out, Point const& p) {
27     out << p.rank_<<": (" << p.x_ << ", "
28         << p.y_ << ")";
29     return out;
30 }
```

Usage: main.cpp

```
1 #include "point.hpp"
2 #include <iostream>
3
4 int main(int argc, char** argv) {
5     Point p1(10, 20);
6     std::cout << "    p1: " << p1 << std::endl;
7     p1.move(42, 42);
8     std::cout << "new p1: " << p1 << std::endl;
9     Point p2 = Point(20, 40);
10    std::cout << "    p2: " << p2 << std::endl;
11    Point total = p1 + p2;
12    std::cout << " total: " << total << std::endl;
13    Point* courbe =
14        new Point [3] {{10,20}, {11,21}, {12,22}};
15    delete[] courbe;
16 }
```

Test unitaire: pointTest.cpp

```
1 #define CATCH_CONFIG_MAIN
2 #include "catch.hpp"
3
4 #include "point.hpp"
5
6 TEST_CASE( "ctor et getters", "[point]" ) {
7     Point c1(1,4);
8     REQUIRE( c1.x() == 1);
9     REQUIRE( c1.y() == 4);
10 }
```

Pour compiler: CMakeLists.txt

```
1 project(MyProject)
2 set(CMAKE_CXX_FLAGS "${CMAKE_C_FLAGS} -Wall")
3 add_executable(prog main.cpp point.cpp)
4 add_executable(PointTest point.cpp pointTest.cpp)
5
6 enable_testing()
7 add_test(NAME point COMMAND PointTest)
```

Exemple de session

```
1 $ cmake . && make
2 (tests automatique du système + compilation)
3 $ ./prog
4     p1: 1:(10, 20)
5     new p1: 1:(52, 62)
6     p2: 2:(20, 40)
7     total: 3:(72, 102)
8 $ ./PointTest
9 =====
10 All tests passed (2 assertions in 1 test case)
11 $ ctest
12 Test project
13     Start 1: point
14 1/1 Test #1: point ..... Passed    0.00 sec
15
16 100% tests passed, 0 tests failed out of 1
17 $
```

Objet comme champ d'une classe

```
1 class PointCouleur {
2     Point p;
3     int couleur;
4     PointCouleur(double x, double y, int c)
5         : p(x,y), couleur(c) {}
6 };
```