

```

1  #include "event.h"
2
3  /* CONSTRUCTEURS */
4
5  // Constructeur simple
6  event::event(){
7      label="";
8      occurred=false;
9      nb_occurence=0;
10     date.clear();
11 }
12
13 // Constructeur évolué, créé l'event à partir des données
14 event::event(string nom){
15     label=nom;
16     occurred=false;
17     nb_occurence=0;
18     date.clear();
19 }
20
21 /* OPERATEURS */
22
23 event event::operator =(const event &e){
24     label=e.label;
25     occurred=e.occurred;
26     nb_occurence=e.nb_occurence;
27     date=date;
28     return *this;
29 }
30
31 /* ASSESSEURS */
32
33 // Assesseur, ressort le nom de l'event
34 string event::get_label(){
35     return label;
36 }
37
38 // Assesseur, ressort l'état de l'event (occurred ou non)
39 bool event::get_occured(){
40     return occurred;
41 }
42
43 // Assesseur, ressort le nombre d'occurence
44 int event::get_nb_occurence(){
45     return nb_occurence;
46 }
47
48 // Assesseur, ressort le tableau des heures d'occurence
49 std::vector<int> event::get_h_occure(){
50     return date;
51 }
52
53 // Assesseur, ressort l'heure de la dernière occurrence
54 int event::get_last_h_occure(){
55     return date.back();
56 }
57
58 // Mutateur, change le label de l'élément
59 void event::set_label(string name){
60     label = name;
61 }
62
63 // Mutateur, change la valeur du nombre de validation de la chronique
64 void event::set_occurence(int nb){
65     nb_occurence = nb;
66 }

```

```

67
68 // Mutateur, change la valeur du booleen occured
69 void event::set_occured(bool occur){
70     occured = occur;
71 }
72
73 // Mutateur, ajoute une valeur dans les heures d'apparition de l'event
74 void event::set_h_event(int heure){
75     date.push_back(heure);
76 }
77
78 // Reinitialisateur, remet à 0 le tableau des heures de validation
79 void event::reset_h_event(){
80     date.clear();
81 }
82
83 // Reinitialisateur, réinitialise tous les attributs
84 void event::reinit_event_all(){
85     label="";
86     occured=false;
87     nb_occurence=0;
88     date.clear();
89 }
90
91 /* METHODES */
92
93 // Affiche certains éléments dans la console
94 void event::afficher(){
95     cout<<"nom = "<<label<<endl;
96     cout<<"event occured = "<<occured<<endl;
97     if (occured) {
98         cout<<"nombre d'occurence = "<<nb_occurence<<endl;
99         cout<<"derniere validation = "<<date.back()<<endl;
100     }
101 }
102
103 // Affiche les heures d'occurence de l'event dans le terminal
104 void event::afficher_heures_occurences(){
105     vector<int>::iterator it; // itérateur
106     for (it = date.begin(); it != date.end() ; it++)
107     {
108         cout<<"occurence à "<<(*it)<<endl;
109     }
110 }
111
112 // Affiche le nombre de fois où l'event est arrivé
113 void event::afficher_nb_occurence(){
114     cout<<"nombre d'occurence = "<<nb_occurence<<endl;
115 }

```