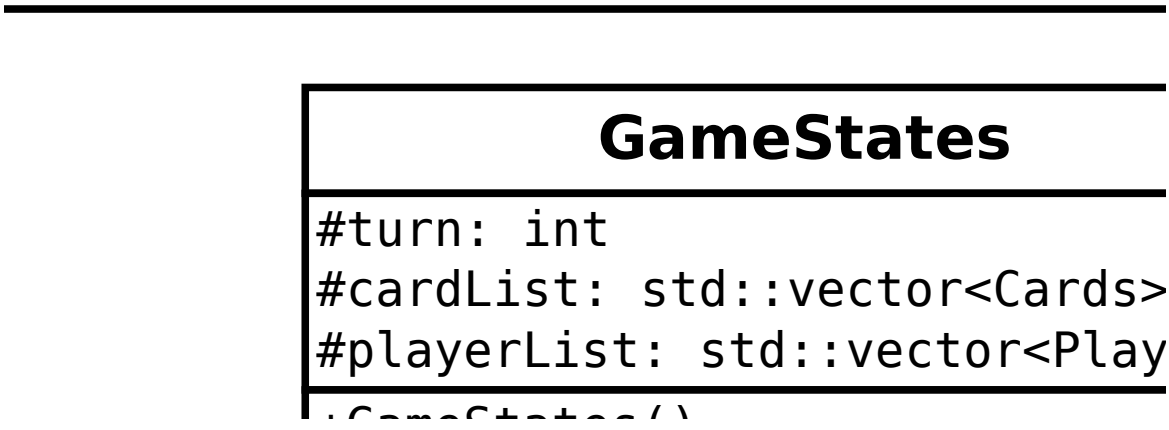
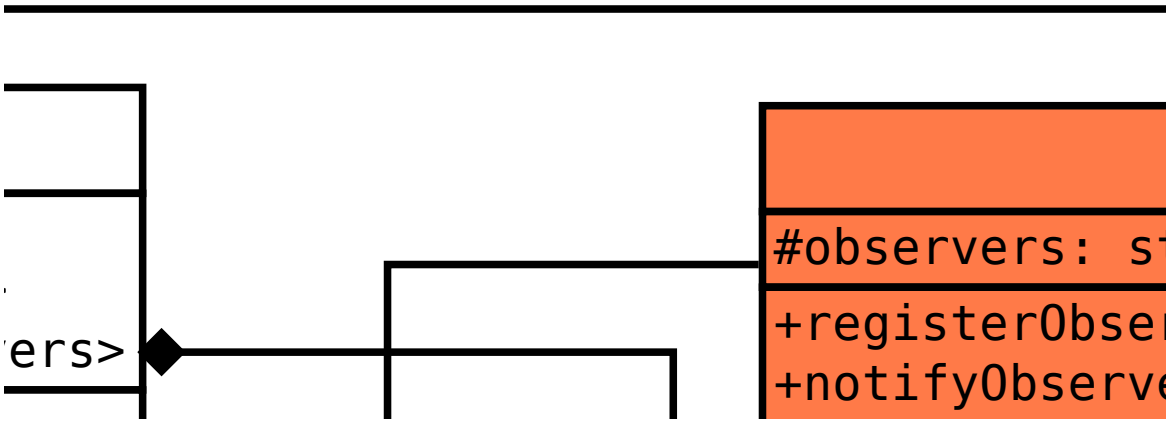
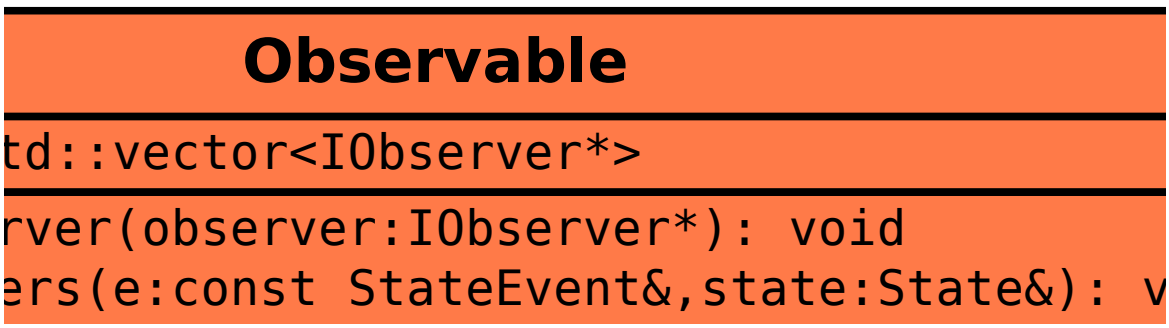
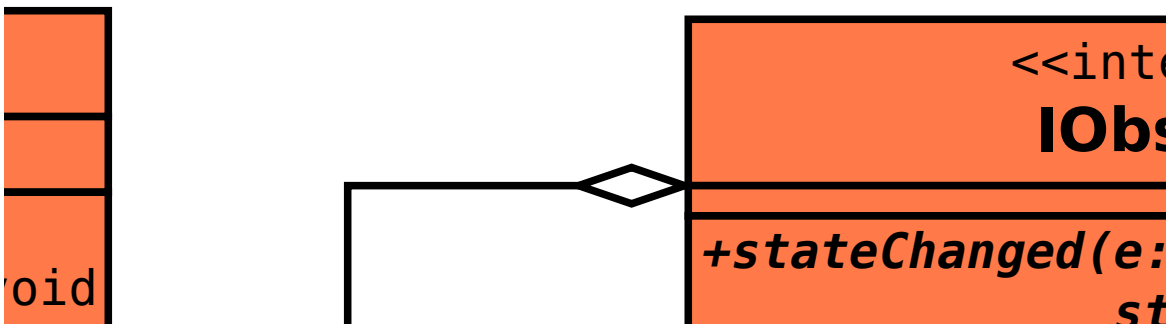
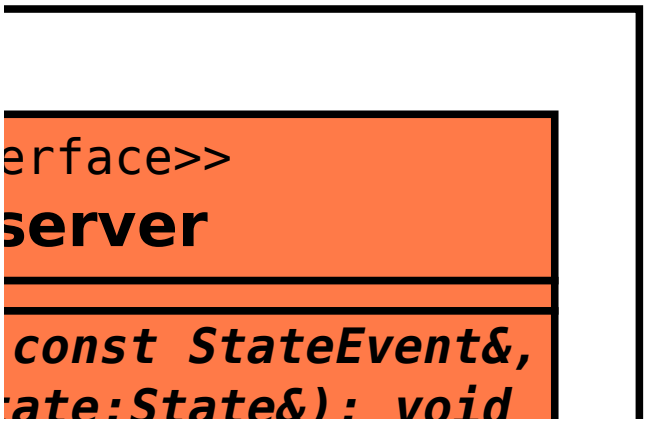

state

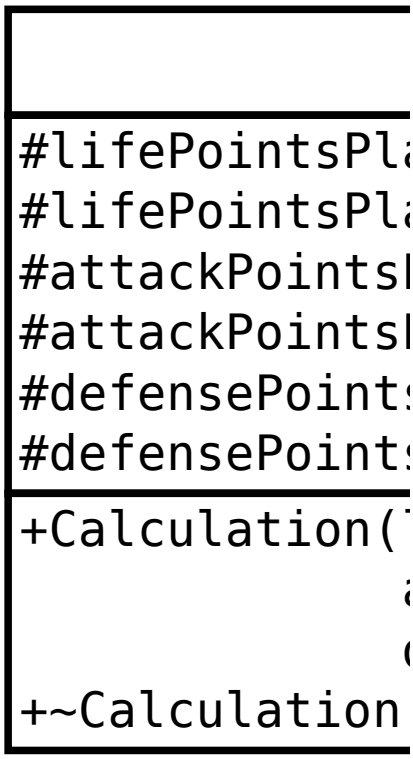

GameStates
#turn: int
#cardList: std::vector<Cards>
#playerList: std::vector<Play


ers>
#observers: st
+registerObser
+notifyObserve


Observable
td::vector<IObserver*>
rver(observer:IObserver*): void
ers(e:const StateEvent&,state:State&): v


void
<<inte
IObs
+stateChanged(e:
st


erface>>
server
const StateEvent&,
ate:State&): void


#lifePointsPla
#lifePointsPla
#attackPointsI
#attackPointsI
#defensePoints
#defensePoints
+Calculation(
+~Calculation

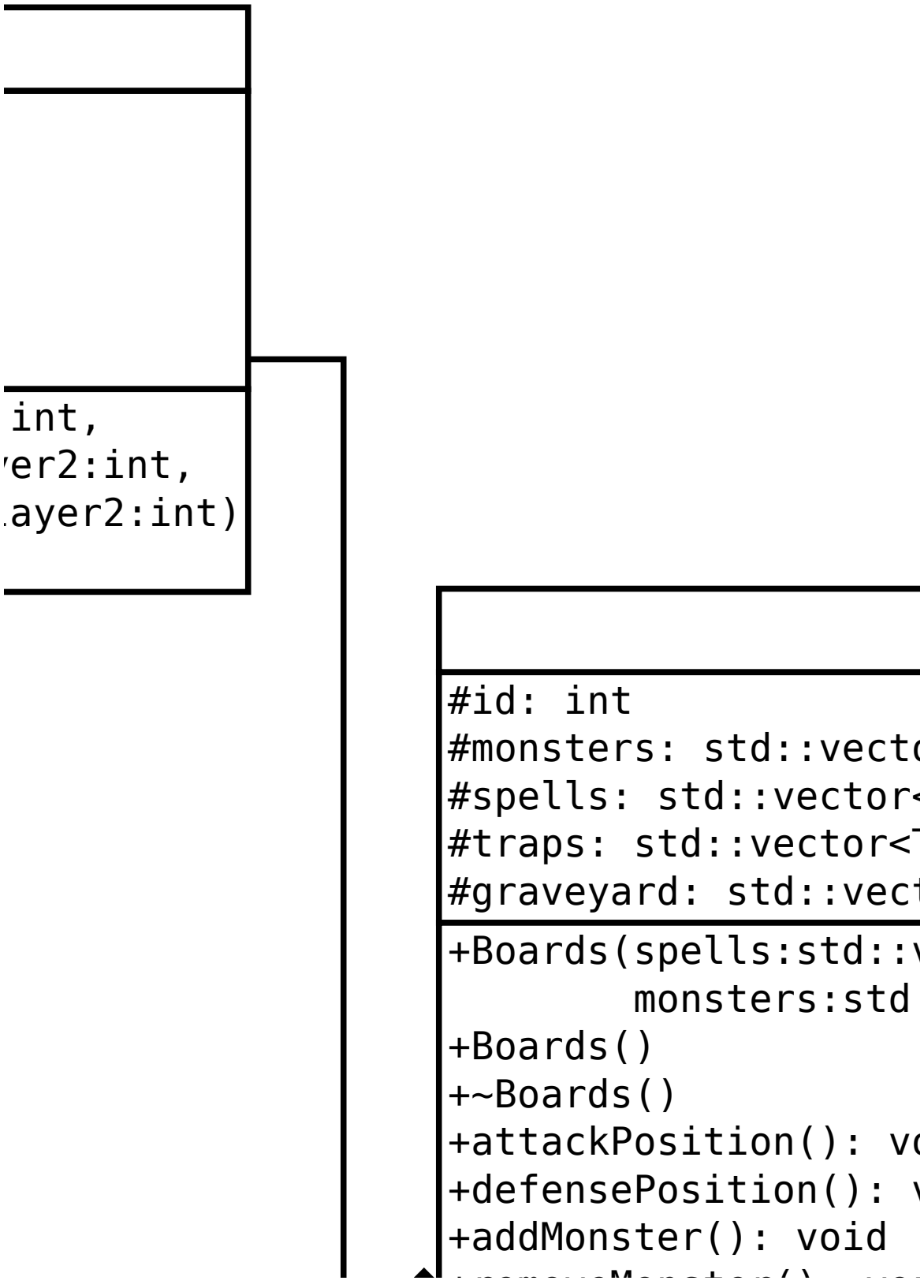

enumeration>>

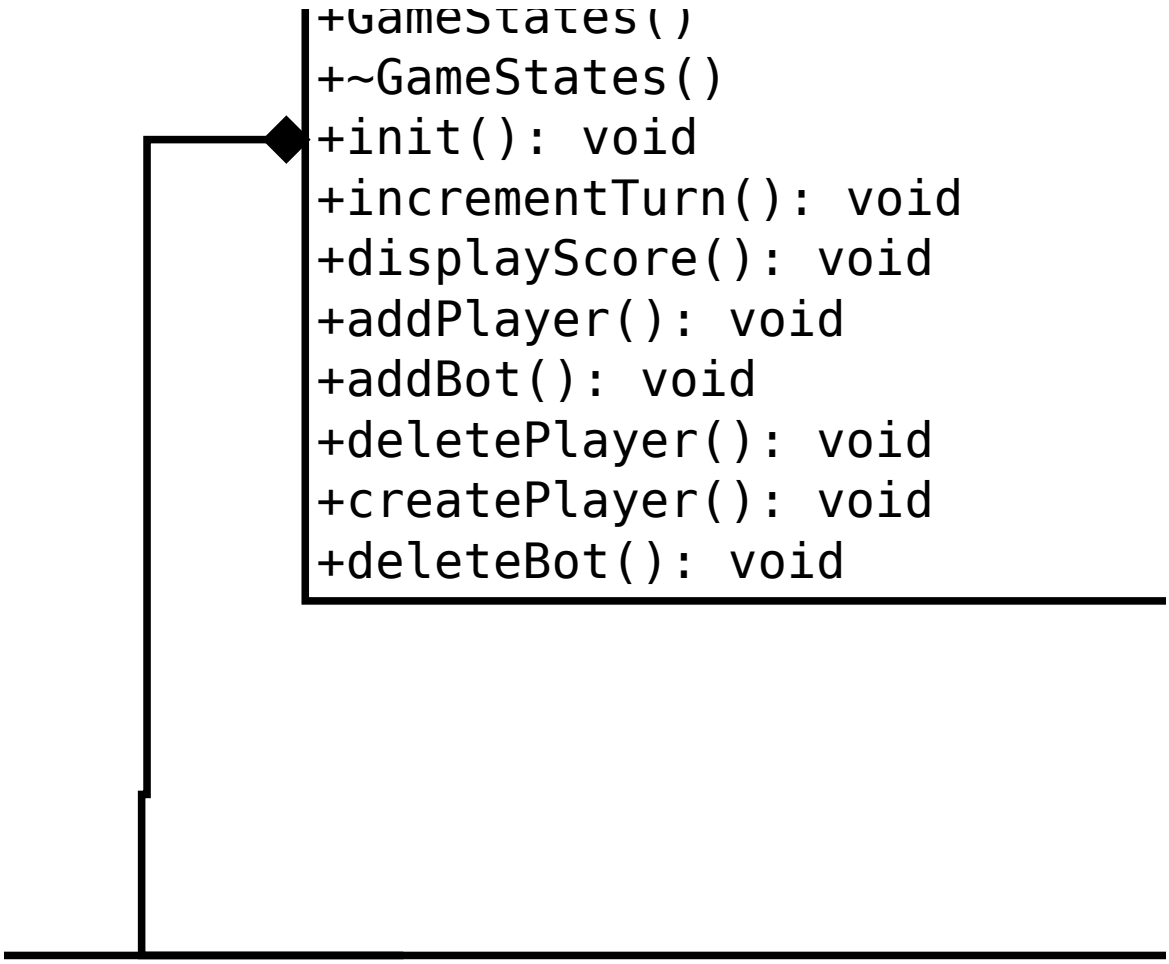
Calculation

ayer1: int
ayer2: int
Player1: int
Player2: int
sPlayer1: int
sPlayer2: int

---

lifePointsPlayer1:int,lifePointsPlayer2:
attackPointsPlayer1:int,attackPointsPlay
defensePointsPlayer1:int,defensePointsPl
()


int,
er2:int,
ayer2:int)
#id: int
#monsters: std::vecto
#spells: std::vector<
#traps: std::vector<T
#graveyard: std::vect
+Boards(spells:std::v
monsters:std
+Boards()
+~Boards()
+attackPosition(): vo
+defensePosition(): v
+addMonster(): void

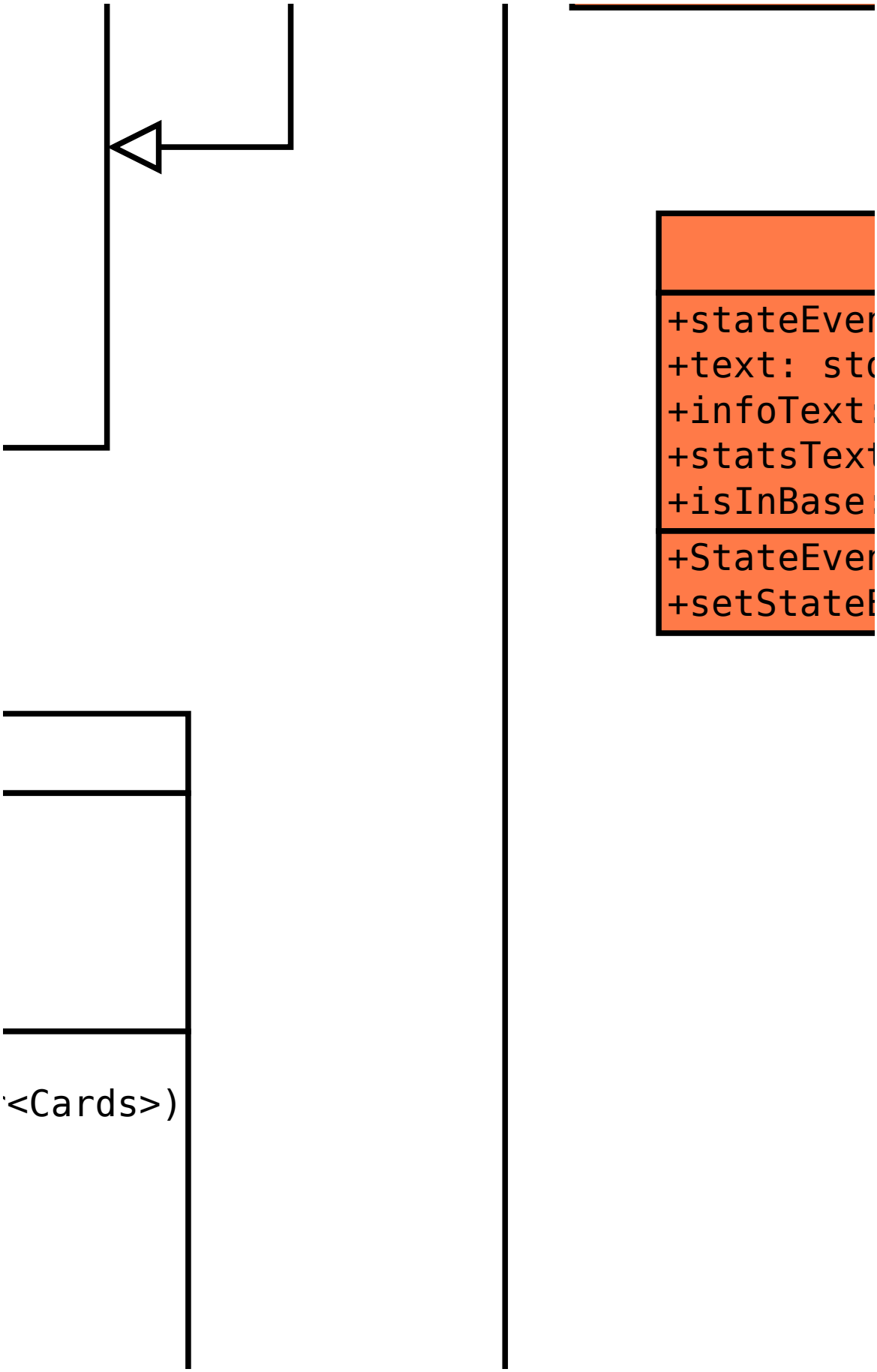


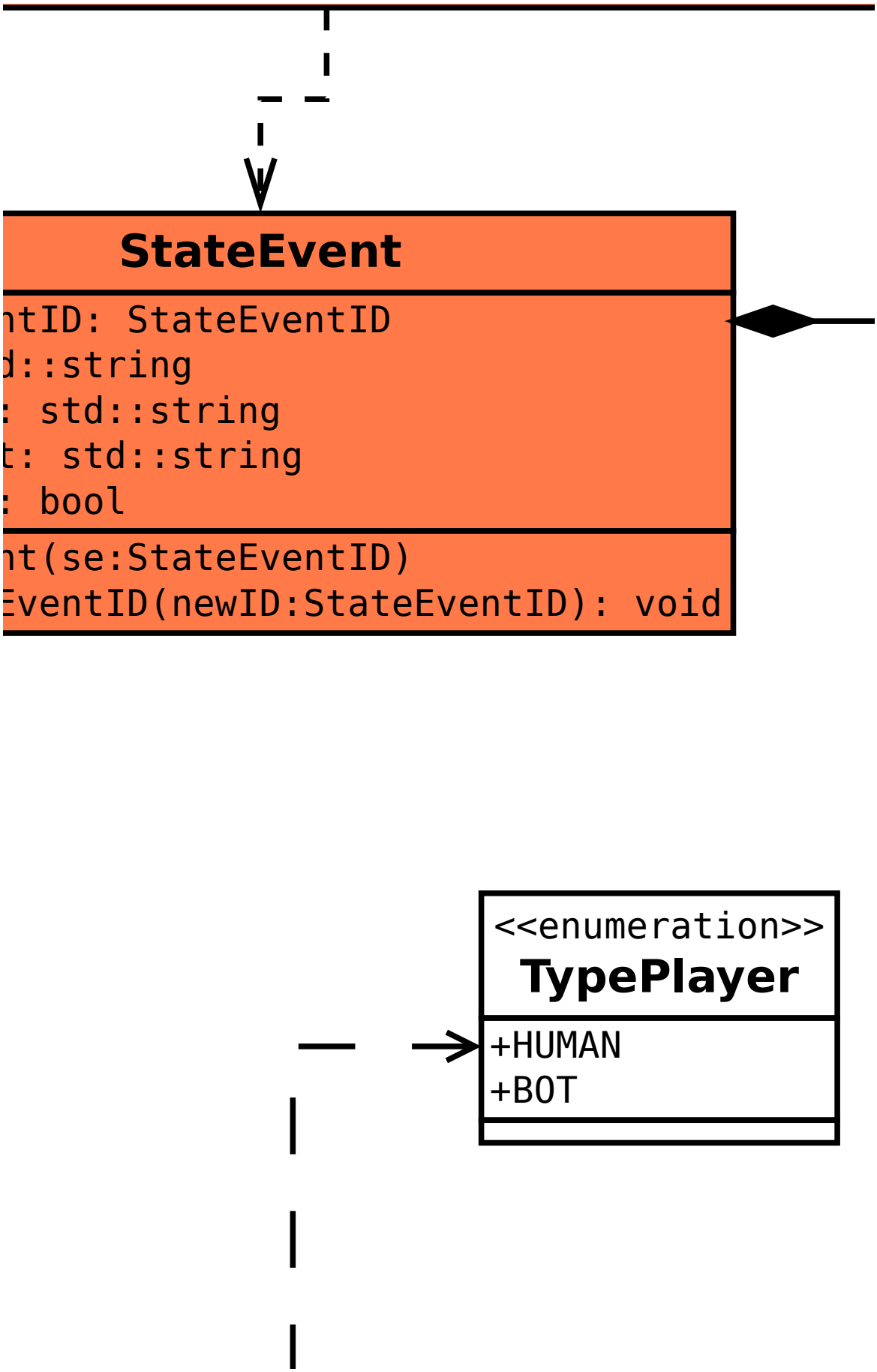
Boards

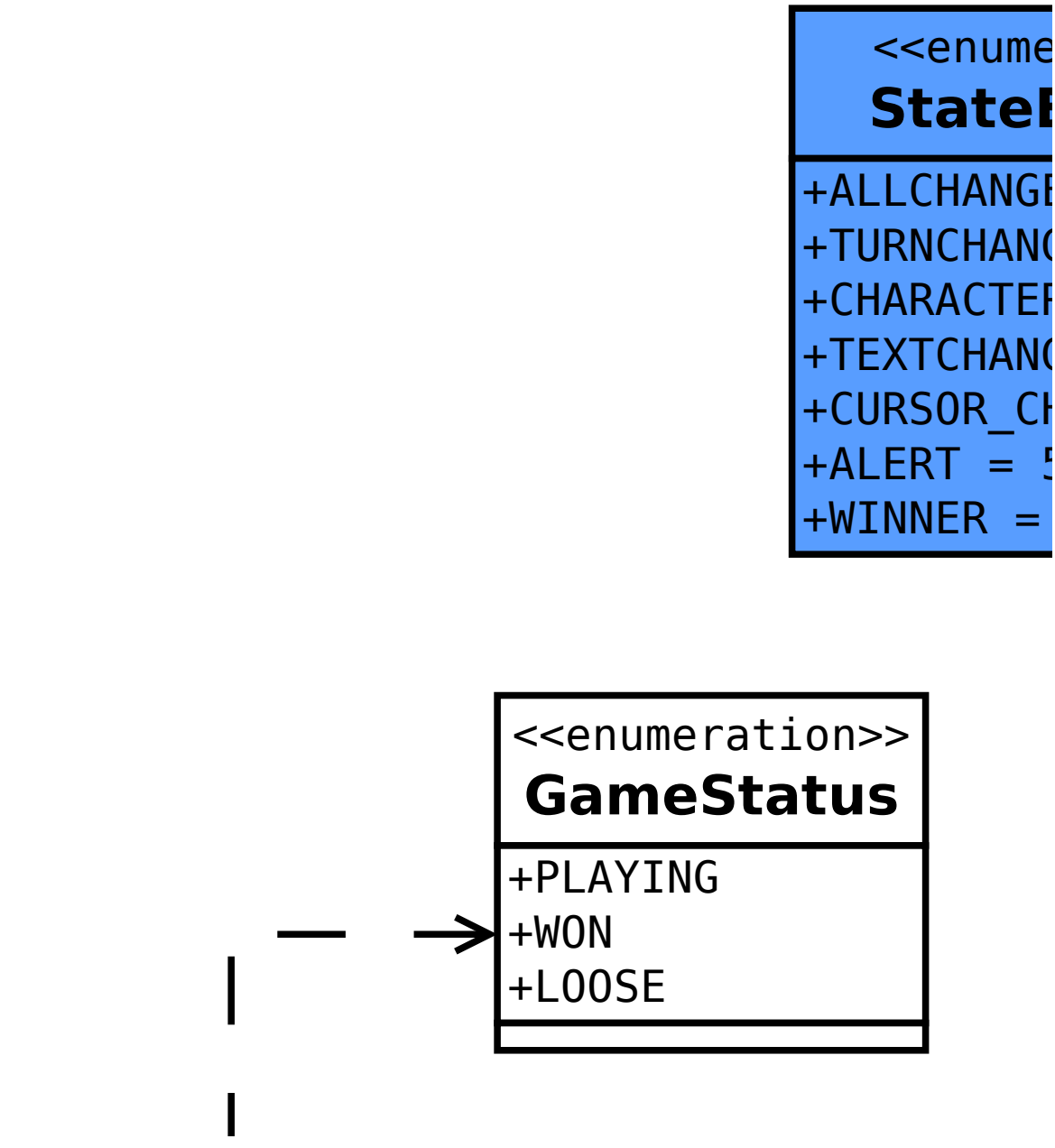
or<Monsters>
<Spells>
Traps>
tor<Cards>

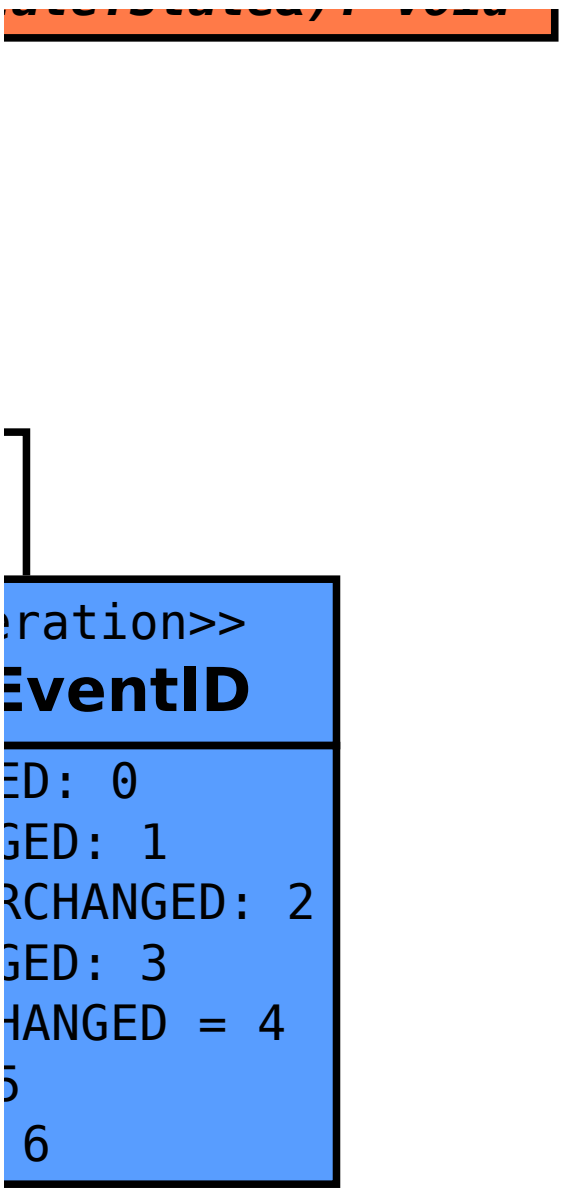
vector<Spells>,traps:std::vector<Traps>,
::vector<Monsters>,graveyard:std::vector

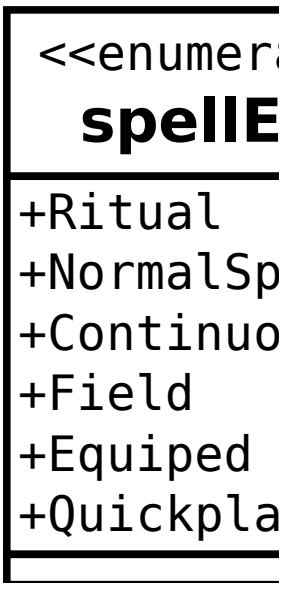
oid
void

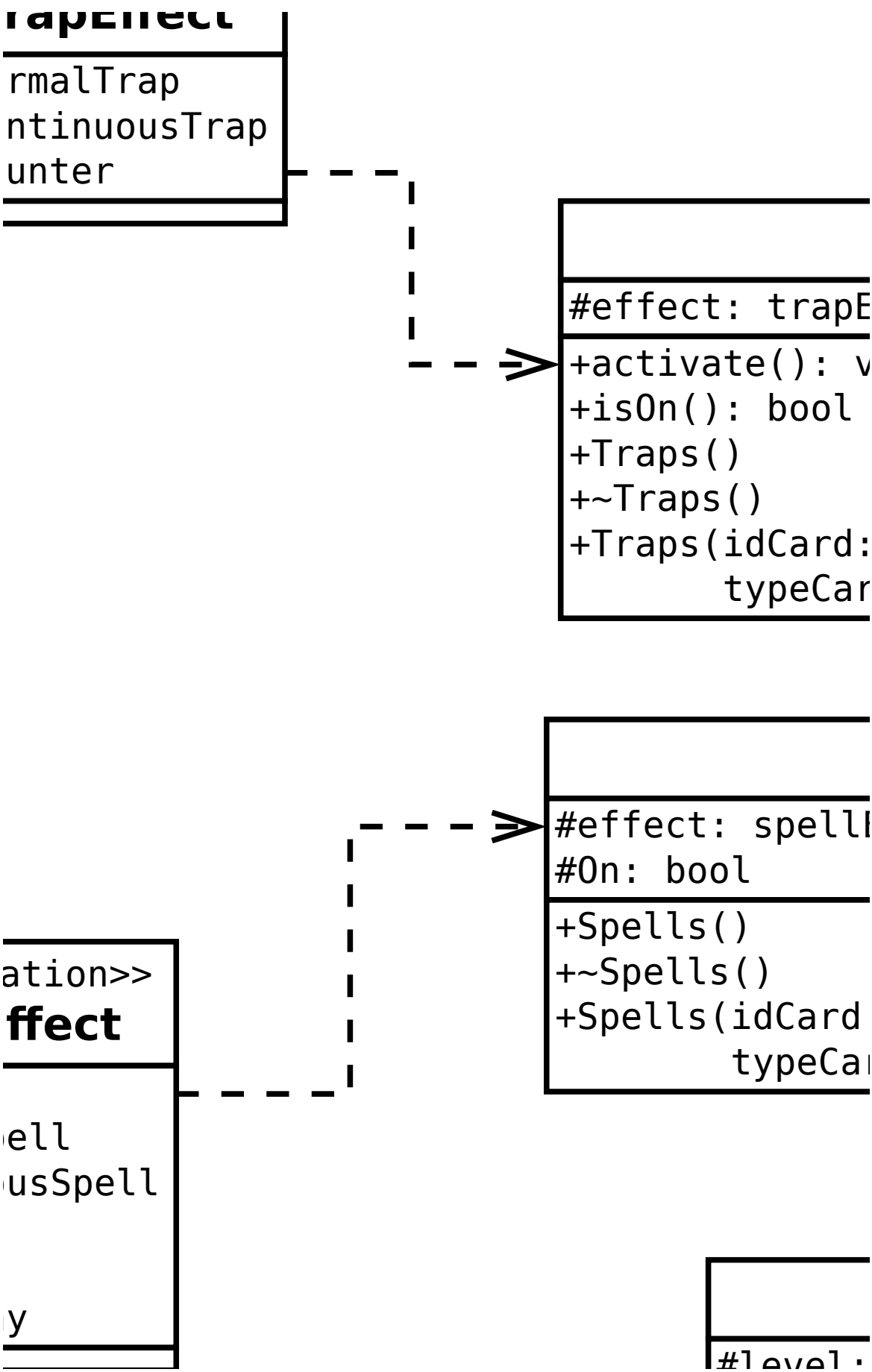

+stateEver
+text: sto
+infoText
+statsTex
+isInBase
+StateEver
+setStateE
<Cards>)


StateEvent
ntID: StateEventID
d::string
: std::string
t: std::string
: bool
nt(se:StateEventID)
EventID(newID:StateEventID): void
<<enumeration>>
TypePlayer
+HUMAN
+BOT


<<enume
StateB
+ALLCHANGE
+TURNCHANG
+CHARACTER
+TEXTCHANG
+CURSOR_CH
+ALERT = 5
+WINNER =
<<enumeration>>
GameStatus
+PLAYING
+WON
+LOOSE


eration>>
EventID
ED: 0
GED: 1
RCHANGED: 2
GED: 3
HANGED = 4
5
6


<<enumer
spellE
+Ritual
+NormalSp
+Continuo
+Field
+Equiped
+Quickpla


rapEffect
rmalTrap
ntinuousTrap
unter
#effect: trapE
+activate(): v
+isOn(): bool
+Traps()
+~Traps()
+Traps(idCard:
typeCar
#effect: spellE
#On: bool
+Spells()
+~Spells()
+Spells(idCard
typeCar
ation>>
ffect
ell
usSpell
y
#level:

Traps

Effect

void

int,name:std::string,description:std::s
rte:CardTypes,effect:trapEffect)

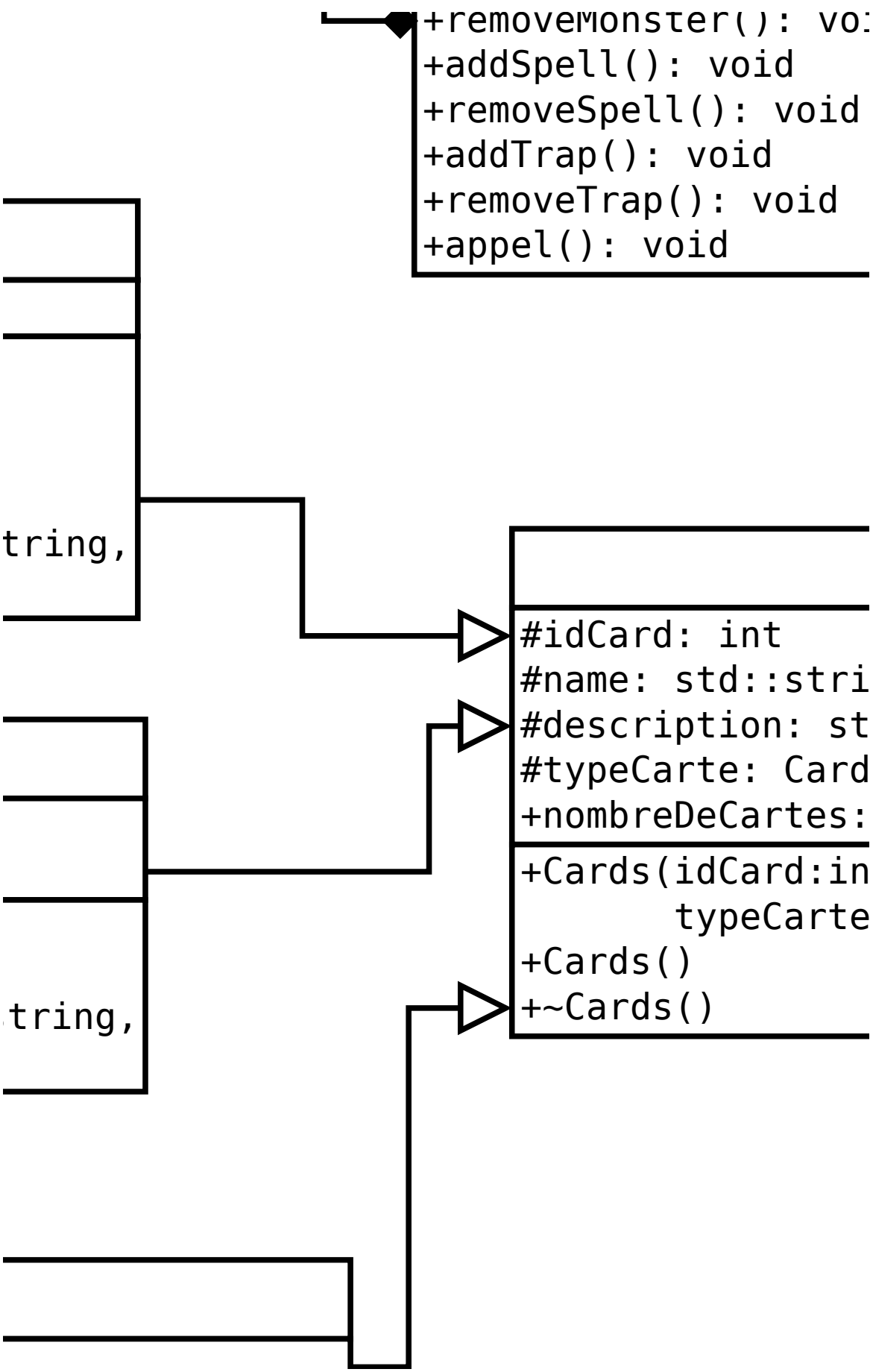
Spells

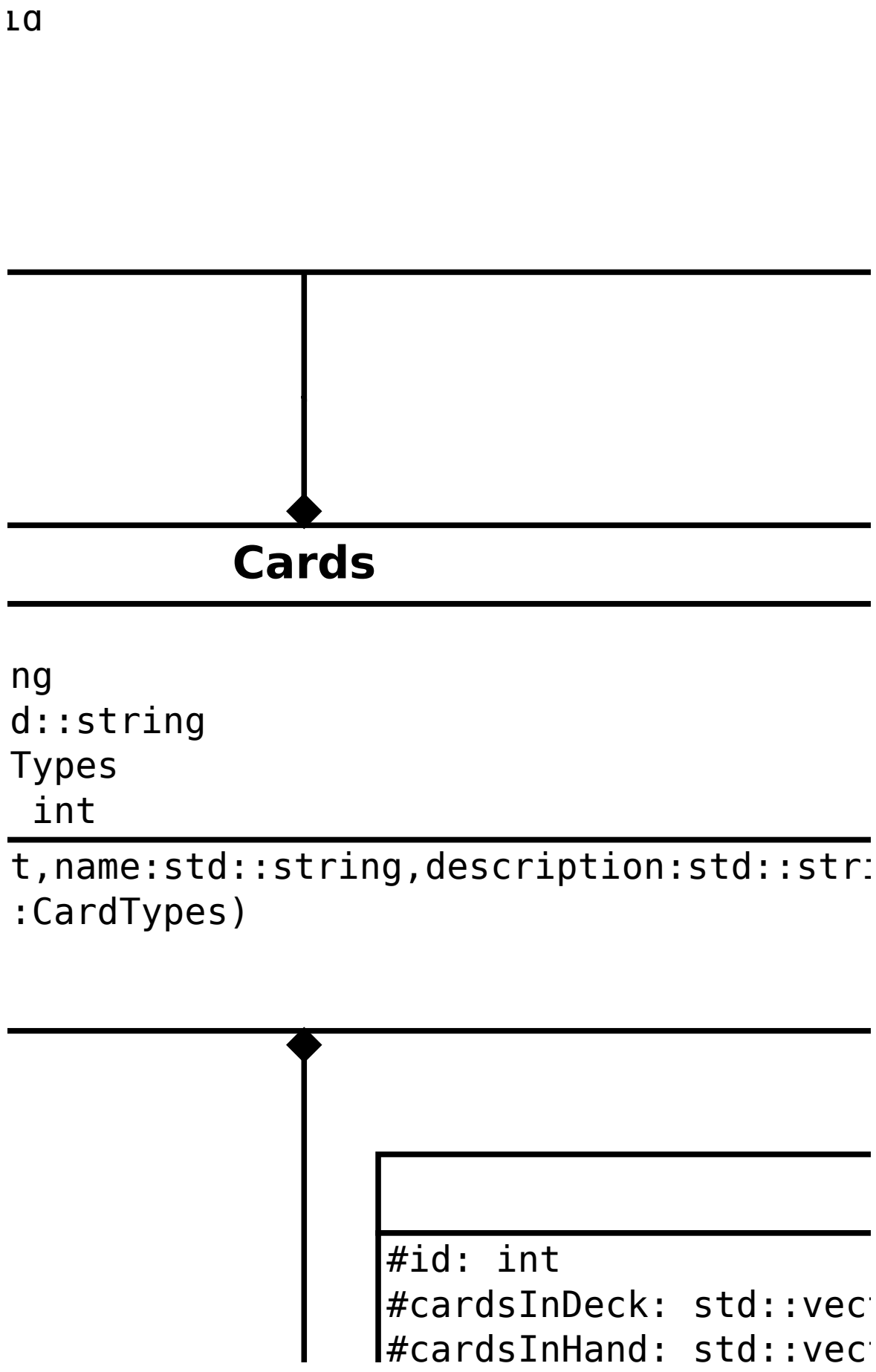
Effect

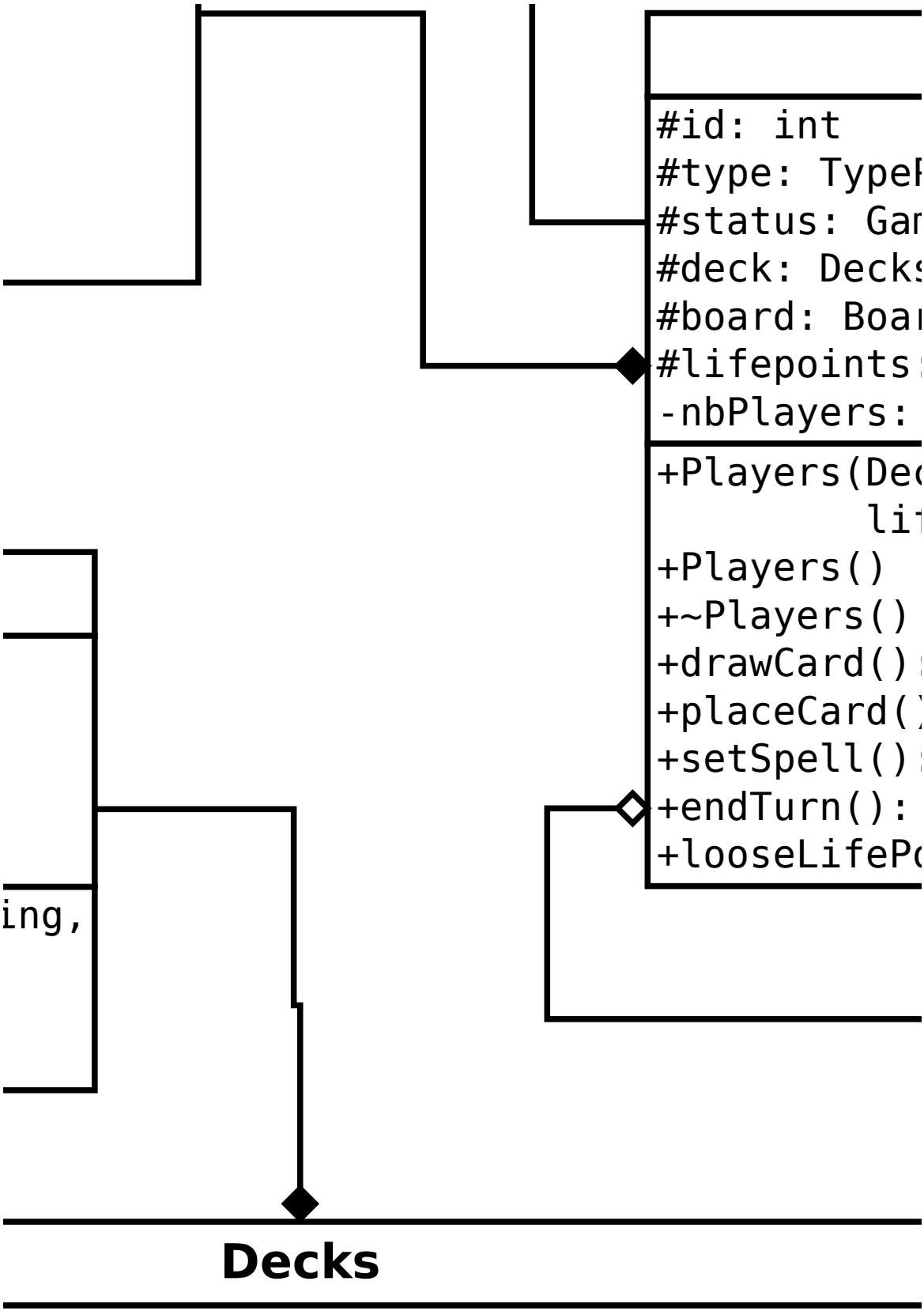
:int,name:std::string,description:std::s
rte:CardTypes,effect:spellEffect)

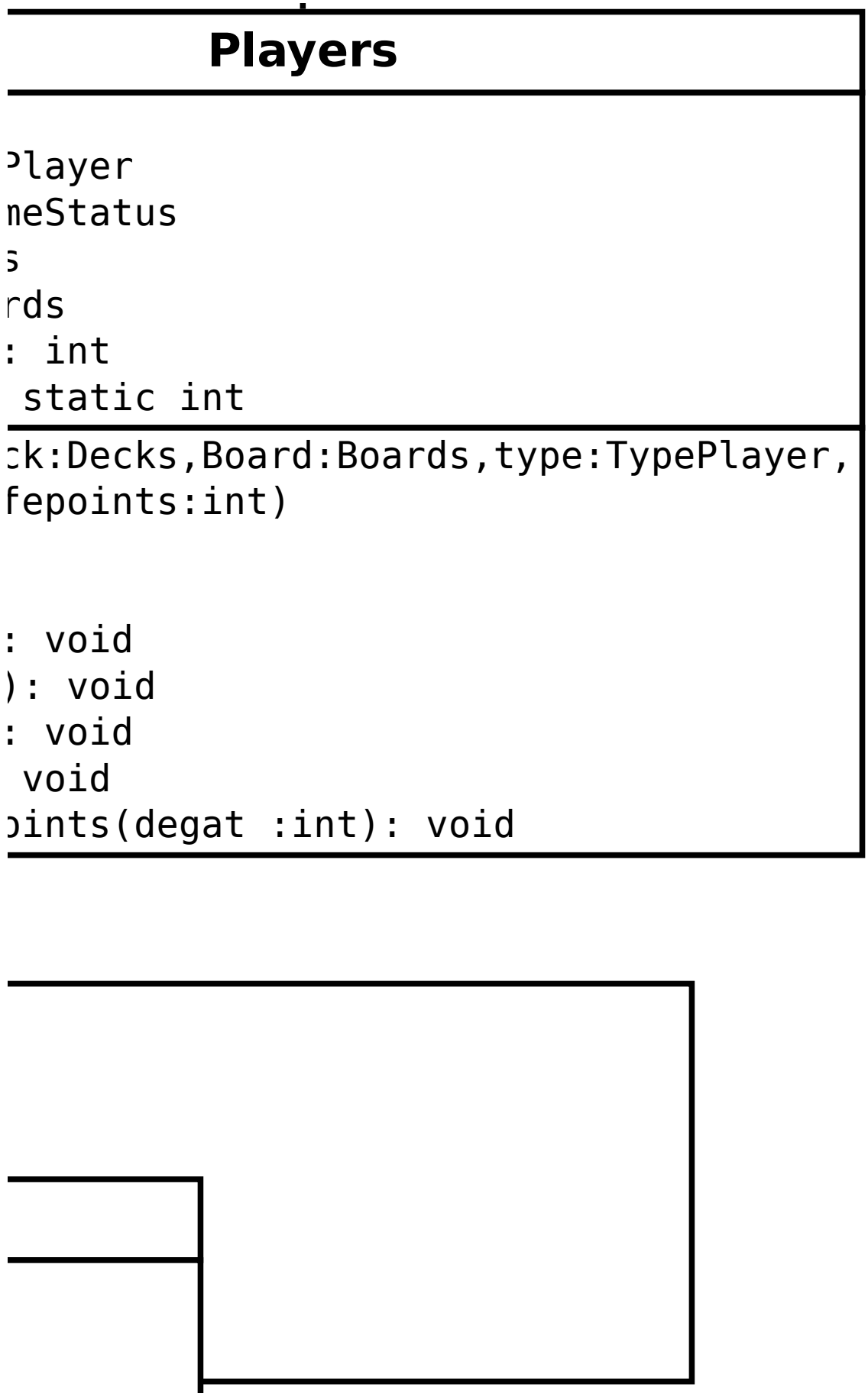
Monsters

int


+removeMonster(): voi
+addSpell(): void
+removeSpell(): void
+addTrap(): void
+removeTrap(): void
+appel(): void
tring,
#idCard: int
#name: std::stri
#description: st
#typeCarte: Card
+nombreDeCartes:
+Cards(idCard:in
typeCarte
+Cards()
+~Cards()
tring,


ıa
Cards
ng
d::string
Types
int
t,name:std::string,description:std::stri
:CardTypes)
#id: int
#cardsInDeck: std::vec
#cardsInHand: std::vec

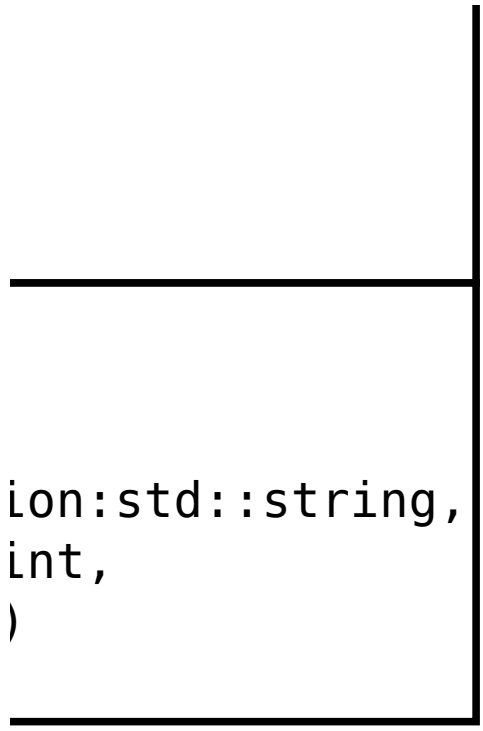

#id: int
#type: TypeP
#status: Gam
#deck: Decks
#board: Boar
#lifepoints:
-nbPlayers:
+Players(Dec
lif
+Players()
+~Players()
+drawCard():
+placeCard()
+setSpell():
+endTurn():
+looseLifePo
ing,
Decks
tor<Cards>
tor<Cards>

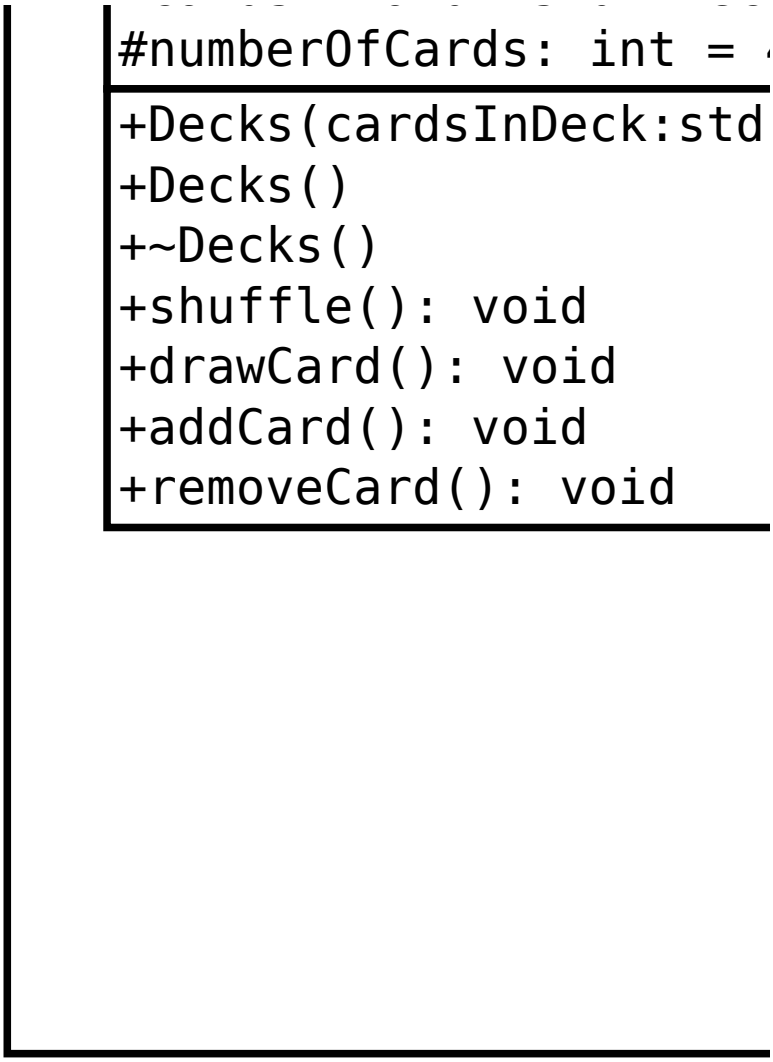

Players
Player
meStatus
s
rds
: int
static int
ck:Decks,Board:Boards,type:TypePlayer,
fepoints:int)
: void
): void
: void
void
oints(degat :int): void


#level
#attack
#defens
#positi
#effet
+attack
+attack
+setSpe
+Monste
+~Monst

: int
e: int
on: bool
: int

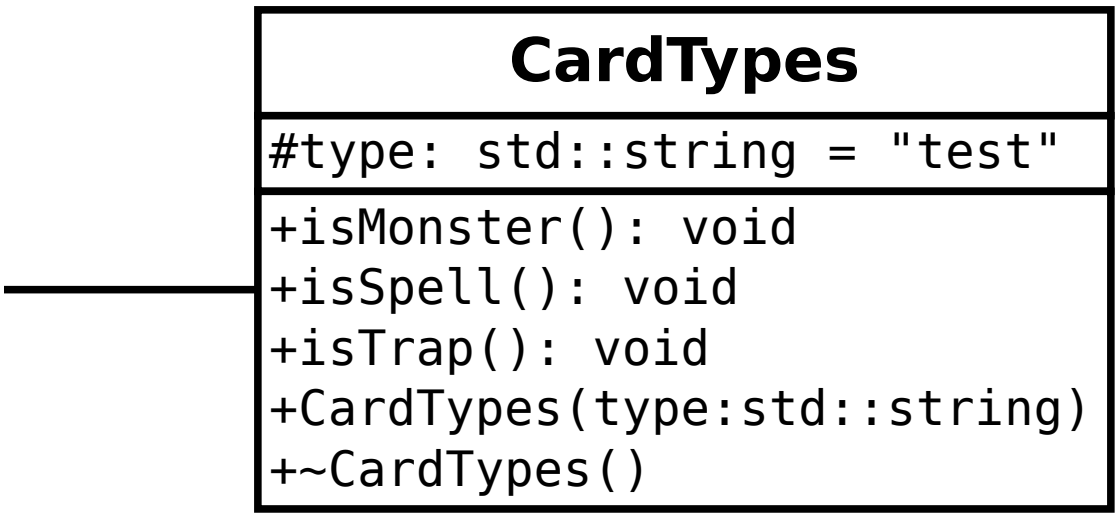
Card(card:Cards): void
Player(player:Players): void
ll(): void
rs(idCard:int,name:std::string,descripti
   typeCarte:CardTypes,effet:int,level:i
   attack:int,defense:int,position:bool)
ers()


ion:std::string,
int,
)


#numberOfCards: int =
+Decks(cardsInDeck:std
+Decks()
+~Decks()
+shuffle(): void
+drawCard(): void
+addCard(): void
+removeCard(): void

40

::vector<Cards>,cardsInHands:std::vector




<Cards>