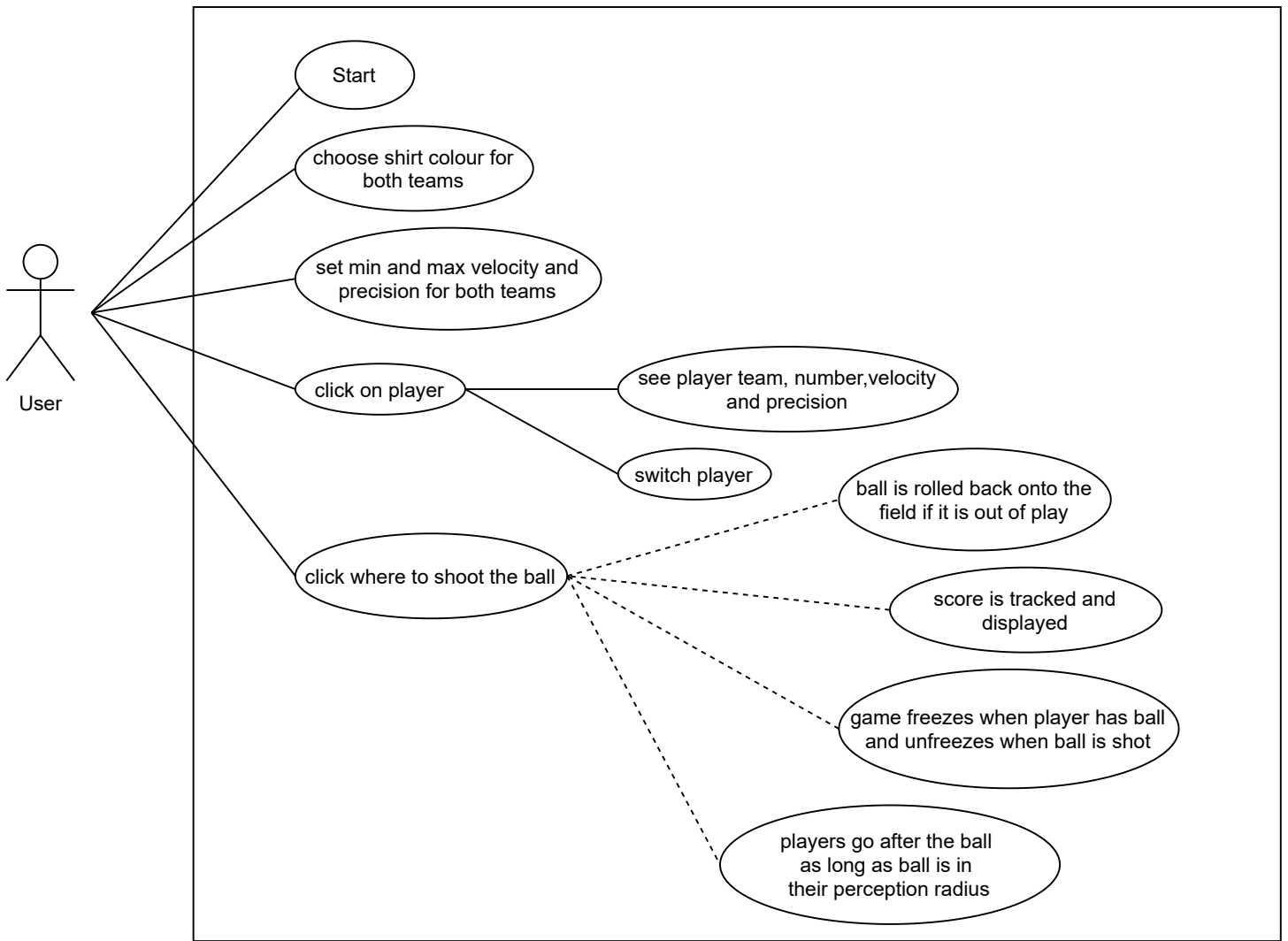
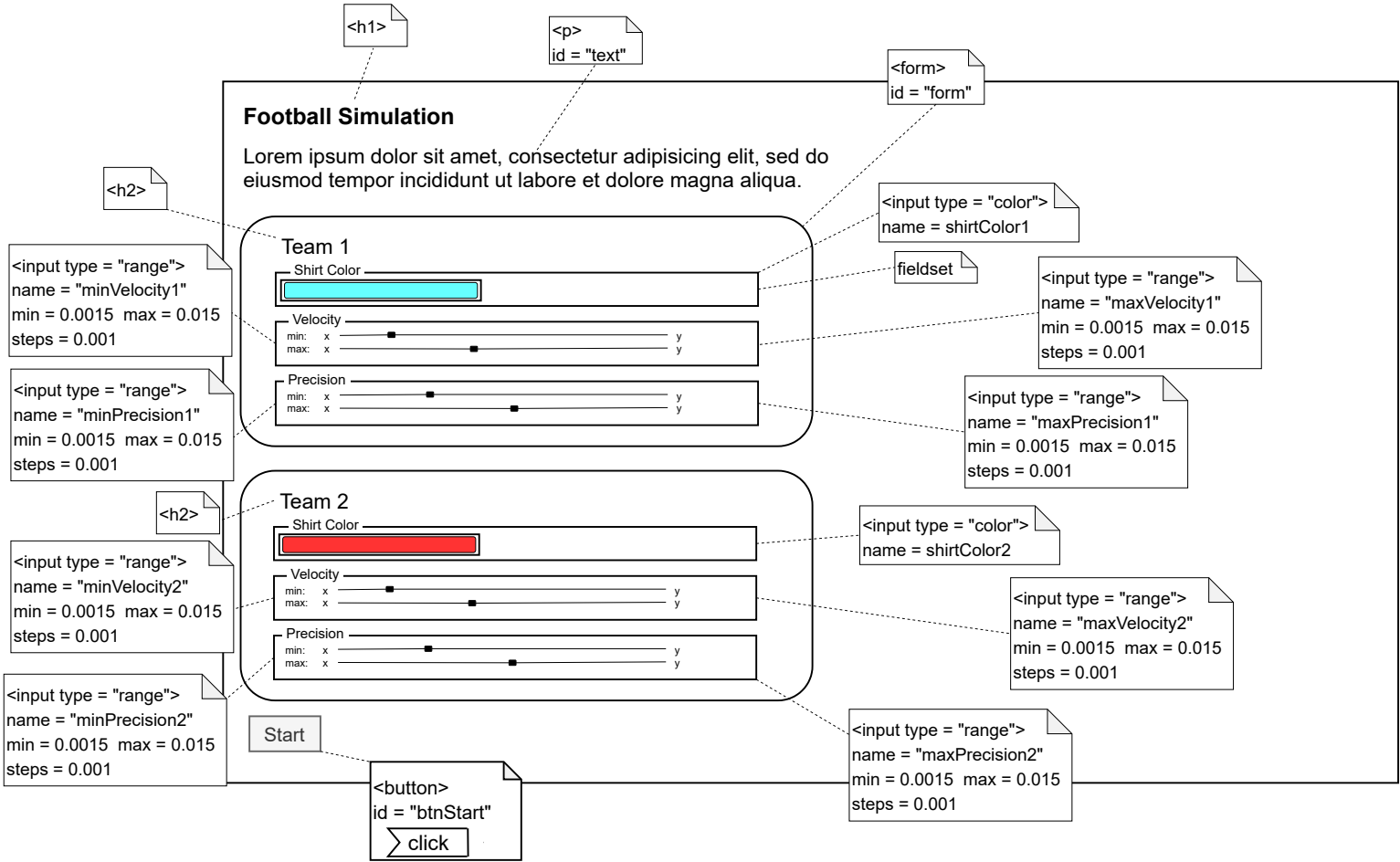


Abschlussarbeit SoSe 2021 - Football Simulation

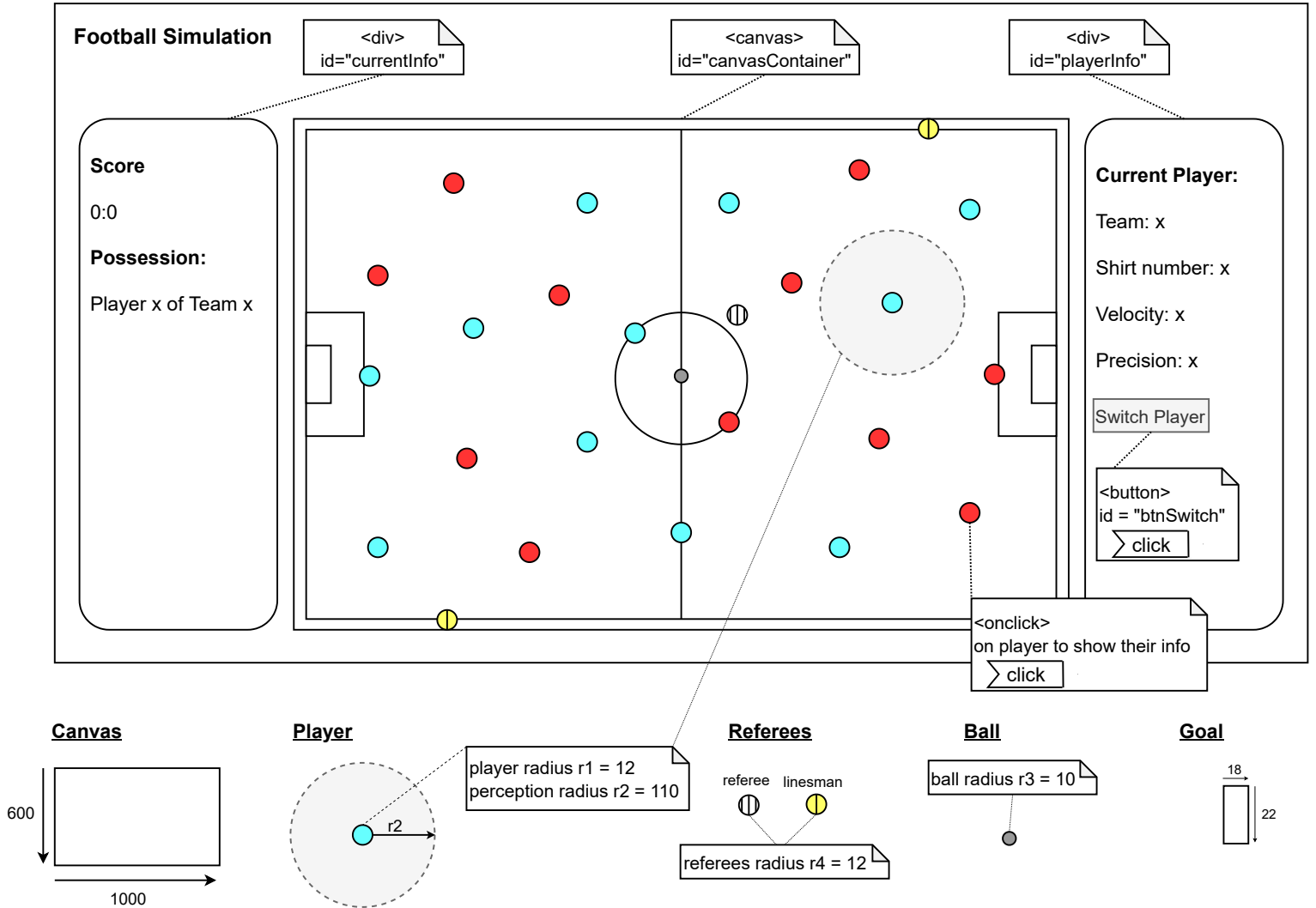
Use-Case-Diagramm



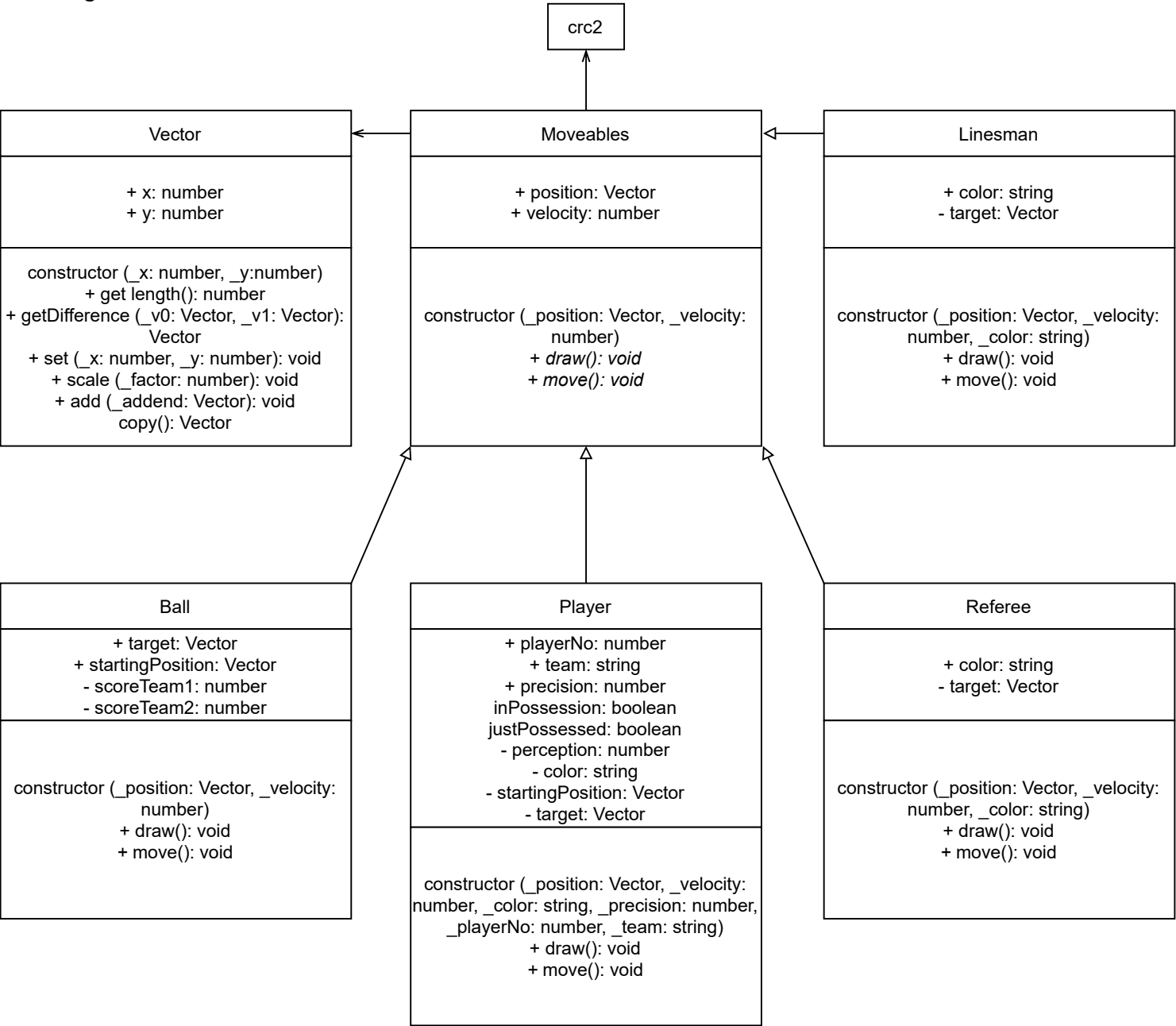
UI Scribble - Form



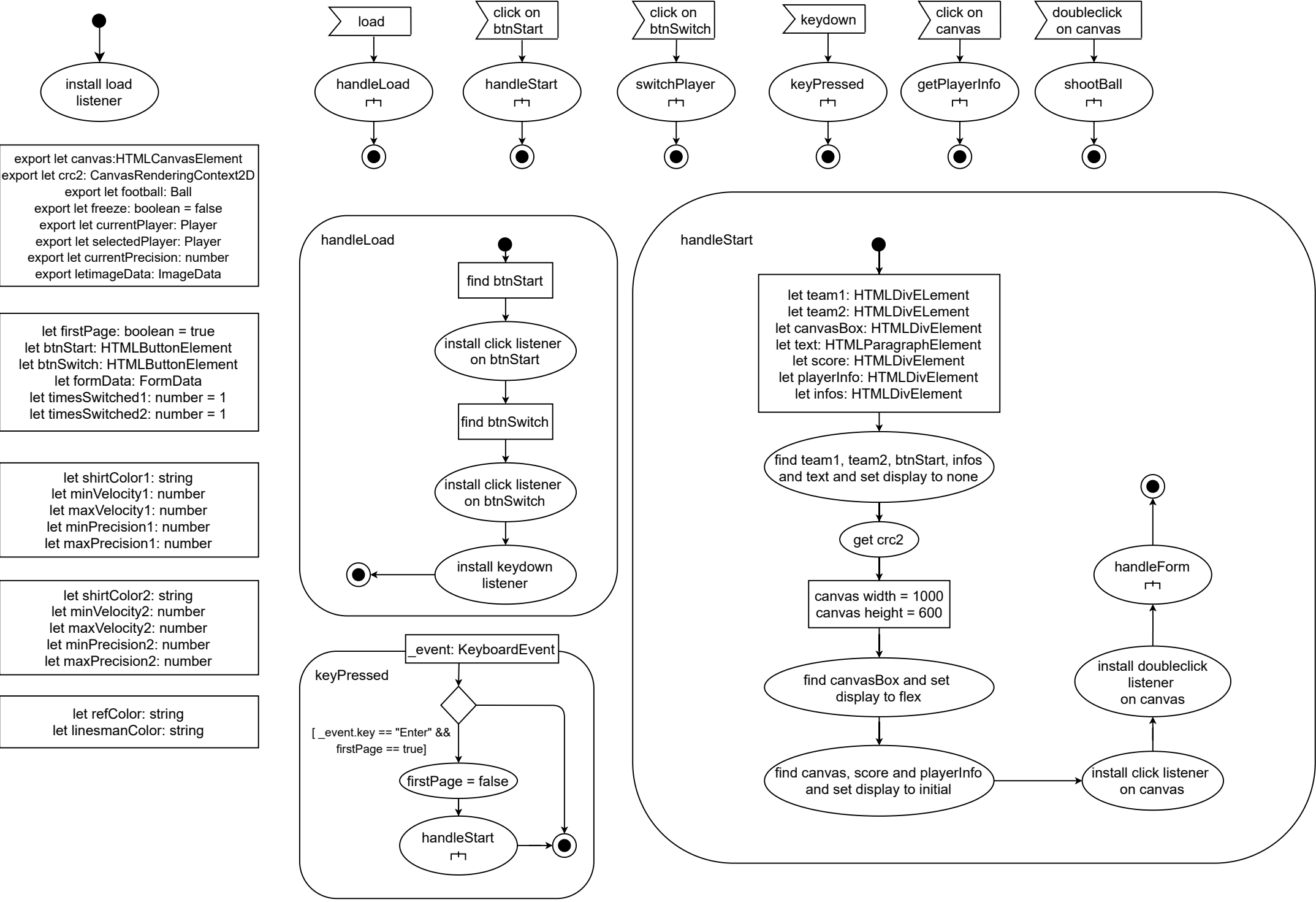
UI Scribble - Field

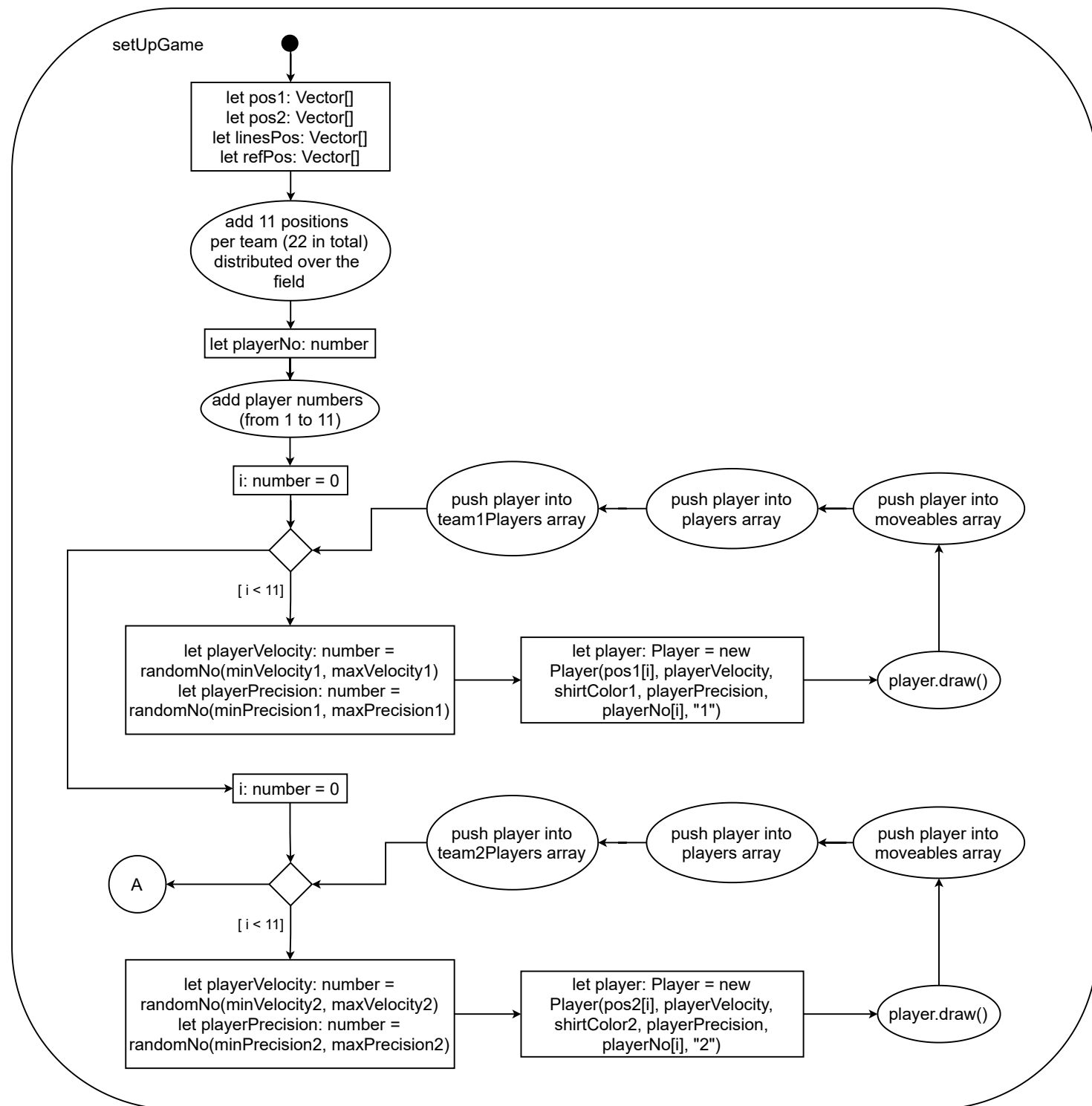
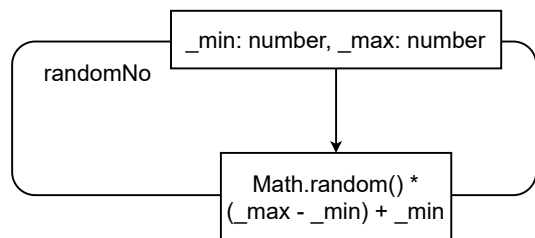
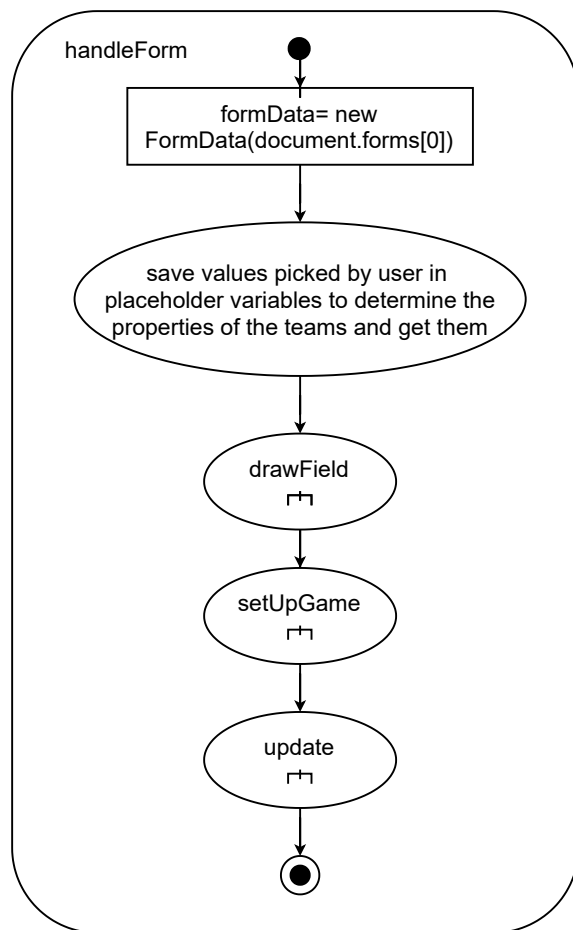


Class Diagram



Activity Diagram - Main

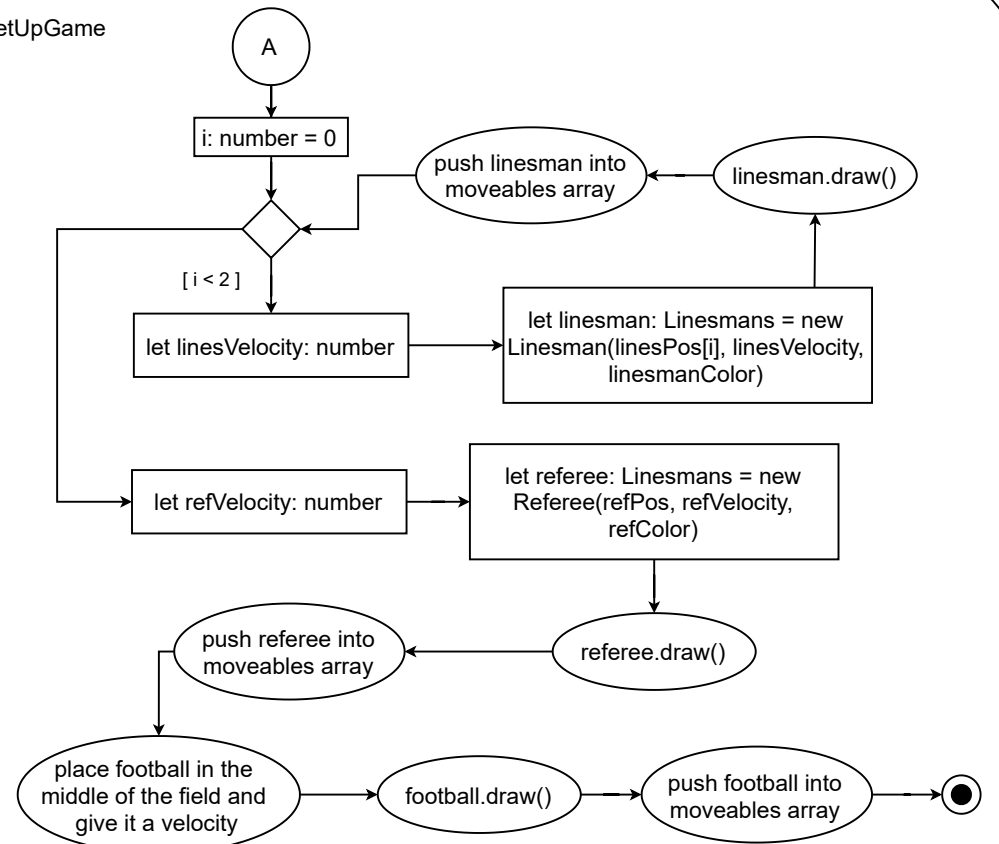




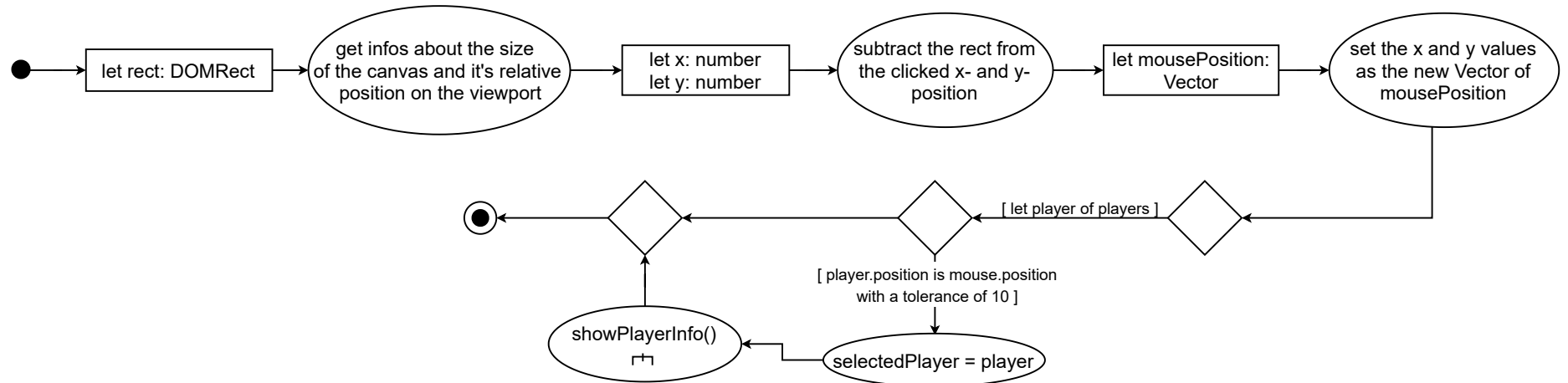
update

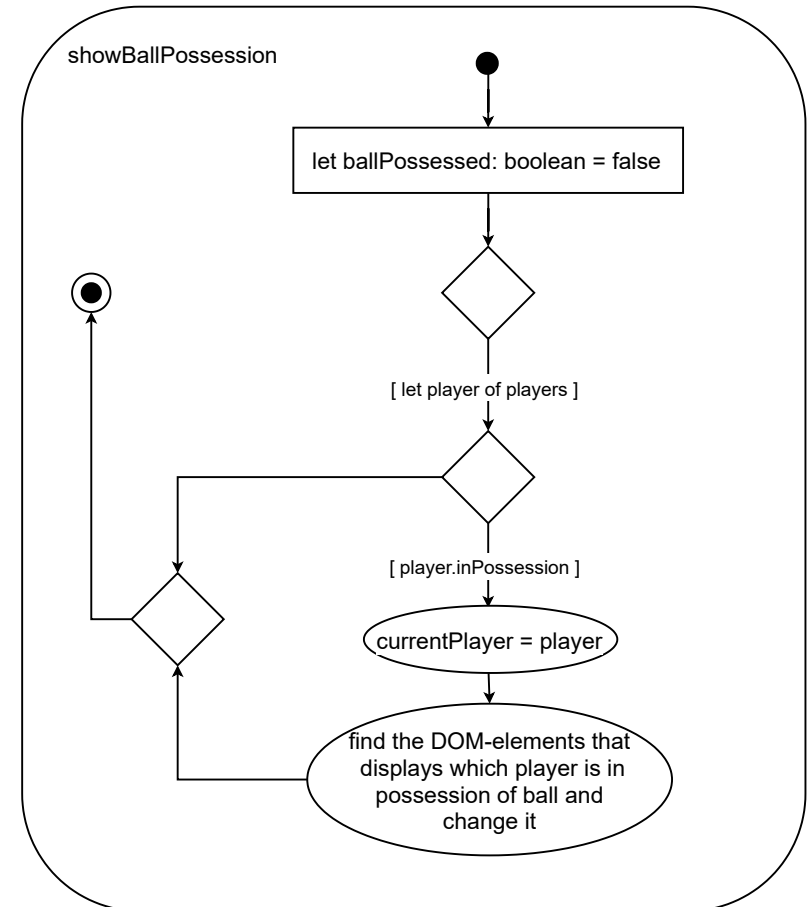
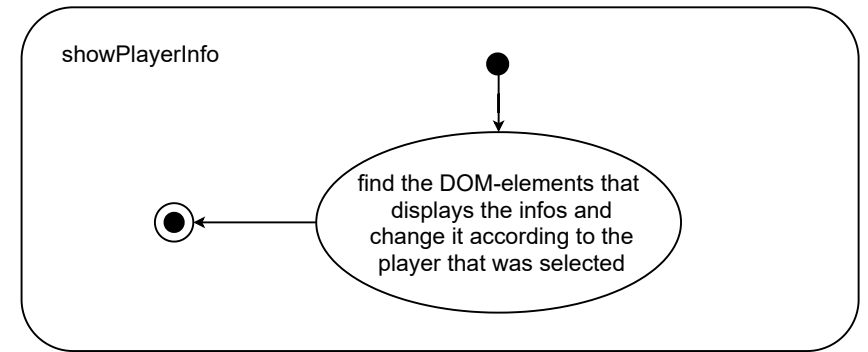
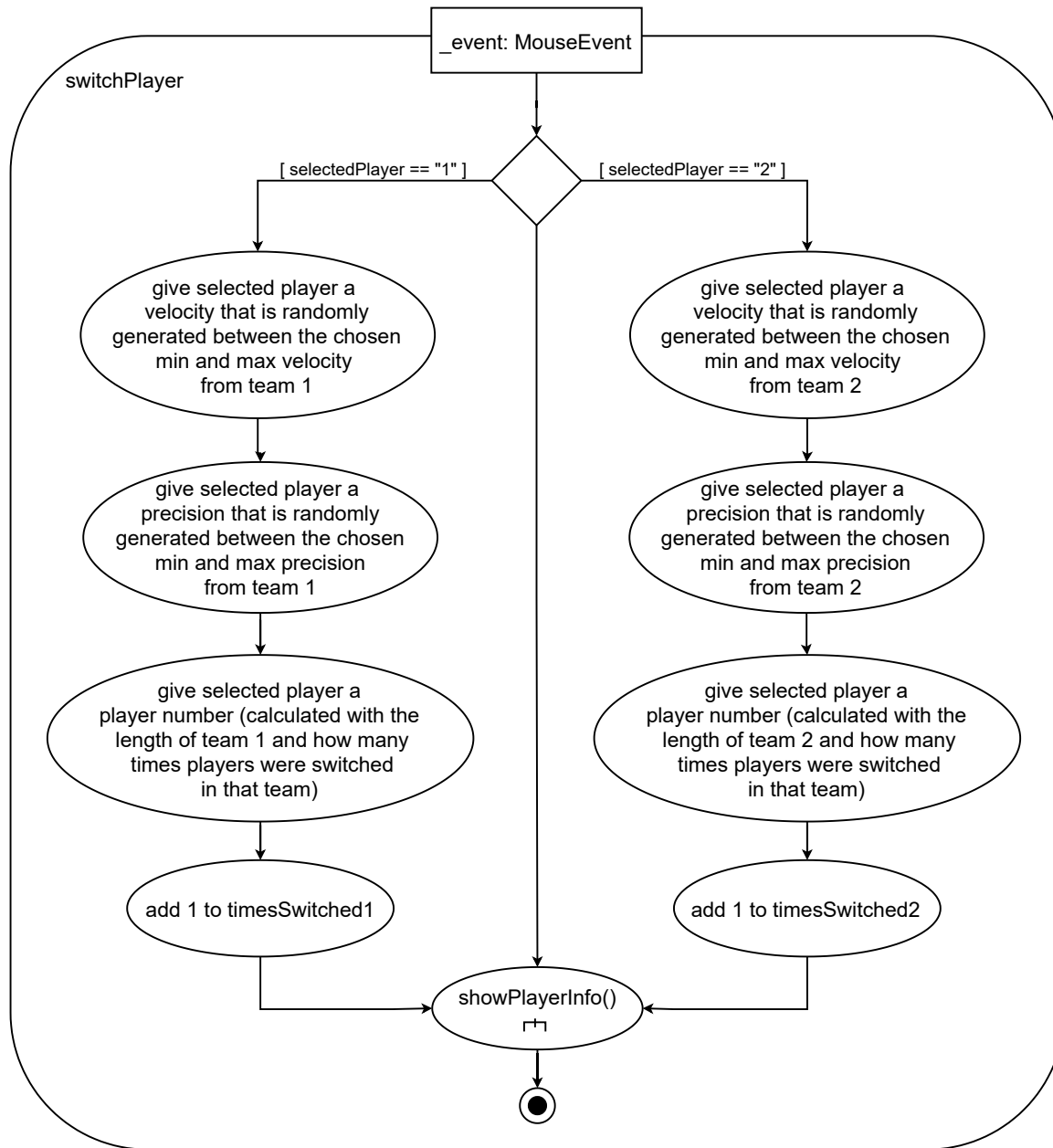


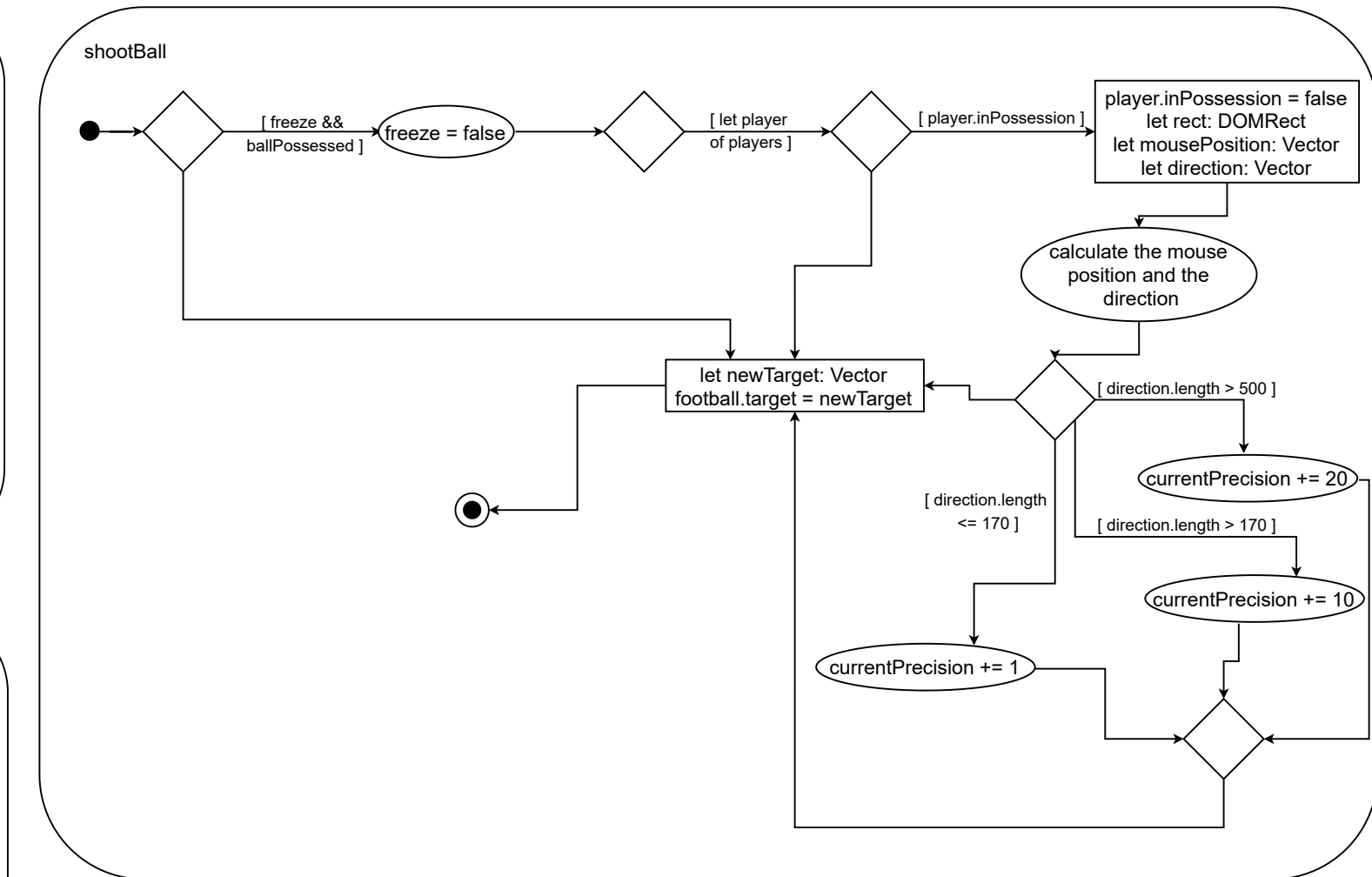
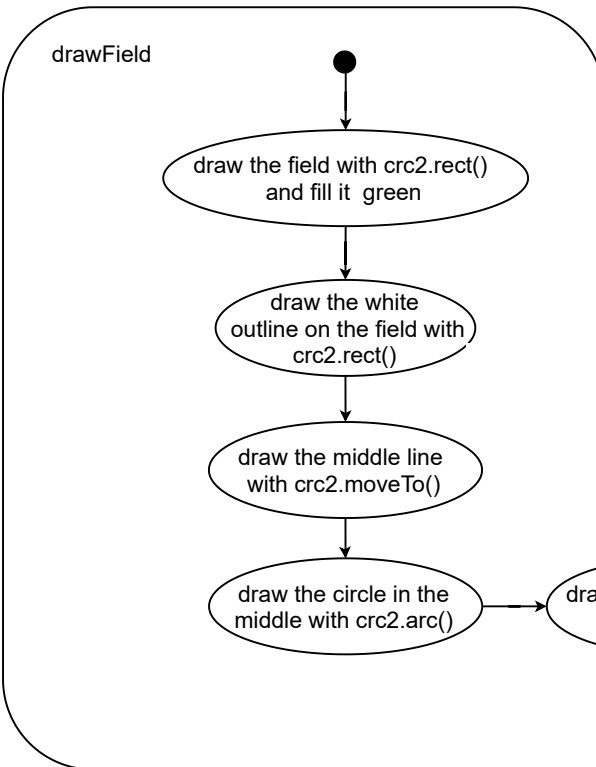
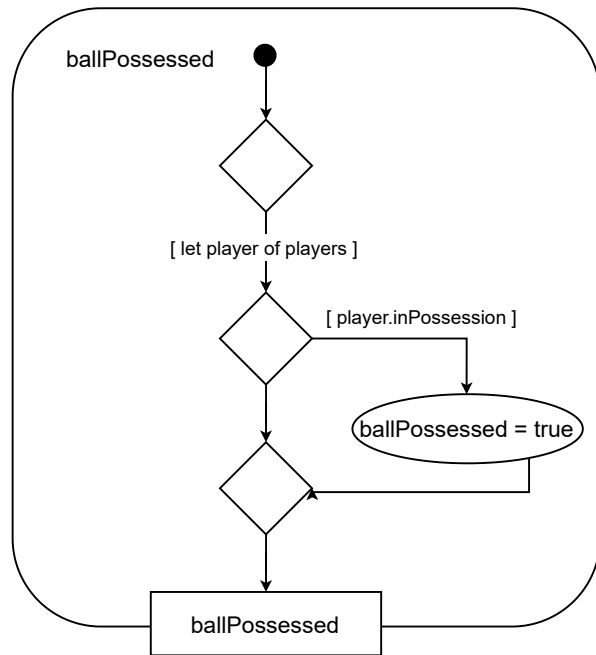
setUpGame



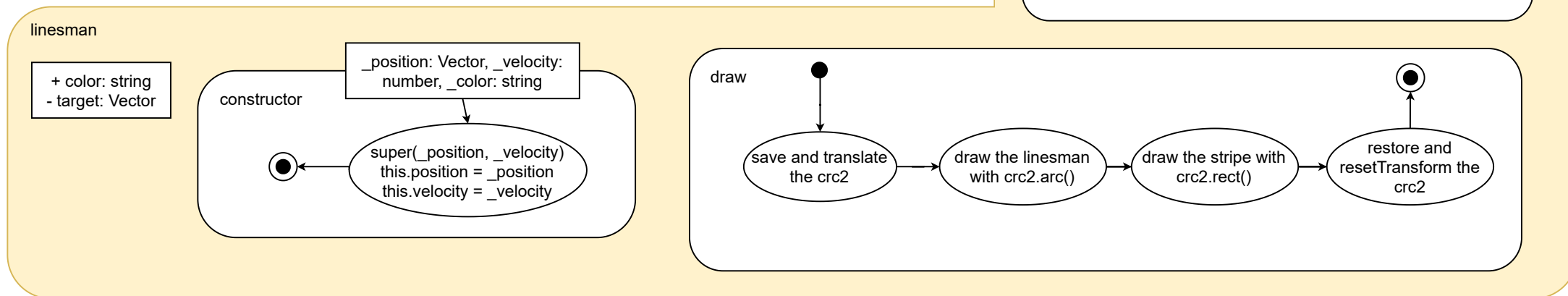
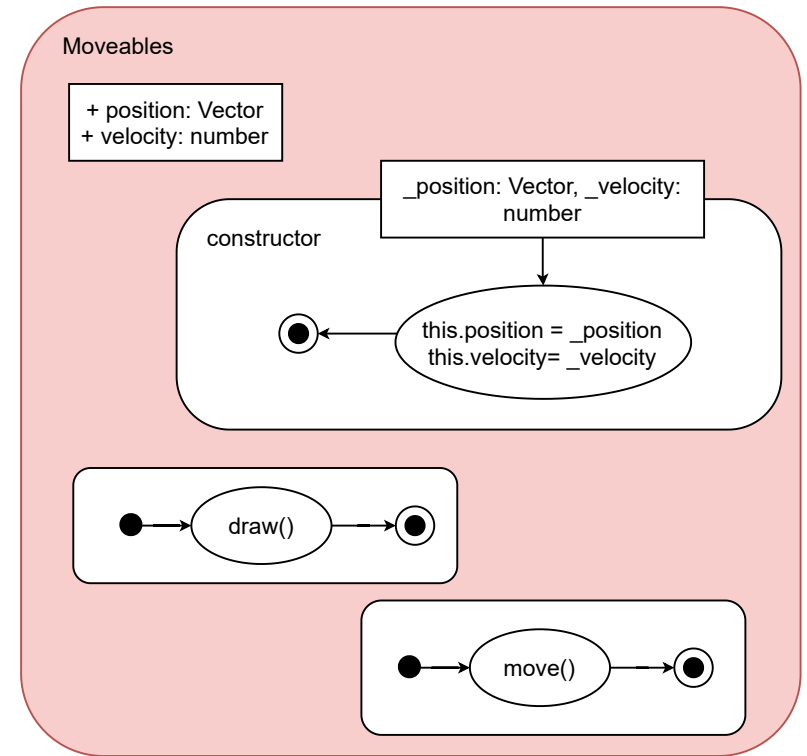
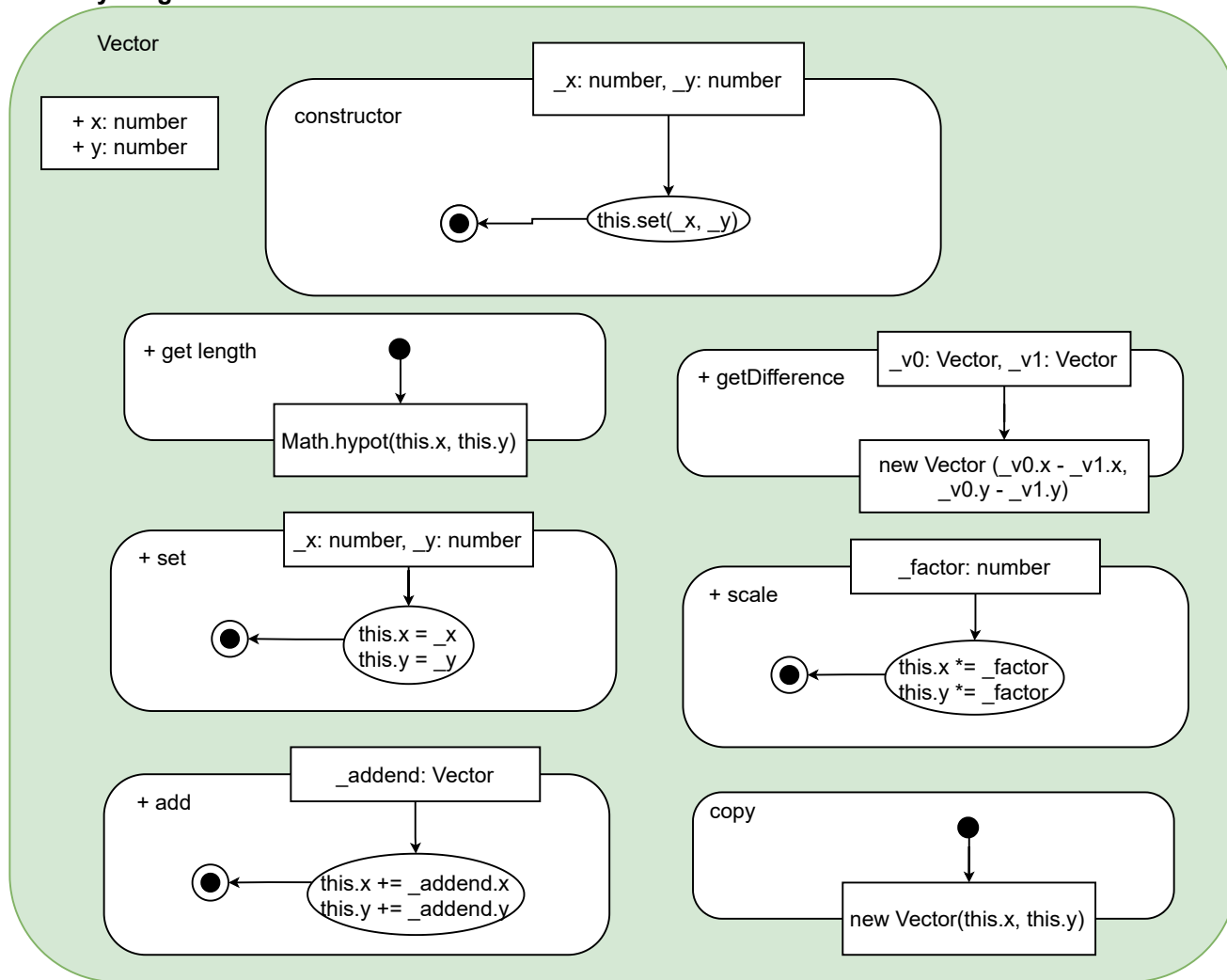
getPlayerInfo







Activity Diagram - Classes



referee

+ color: string
- target: Vector

constructor

_position: Vector, _velocity:
number, _color: string

```
super(_position, _velocity)
this.position = _position
this.velocity = _velocity
```

draw

save and translate
the crc2

draw the referee with
crc2.arc()

draw the stripes with
crc2.rect()

restore and
resetTransform the
crc2

player

+ playerNo: number
+ team: string
+ precision: number
inPossession: boolean
justPossessed: boolean
- color: string
- perception: number
- target: Vector
- startPosition: Vector

constructor

_position: Vector, _velocity: number,
_color: string, _precision: number,
_playerNo: number, _team: string

```
this.color = _color
this.precision = _precision
this.playerNo = _playerNo
this.team = _team
this.startingPosition = _position.copy()
this.velocity = _velocity
this.position = _position
```

draw

save and translate
the crc2

draw the players with
crc2.arc()

restore and
resetTransform the
crc2

move

this.target = new Vector
let direction: Vector

scale the direction
according to the
velocity and add it to
the position

returnToStarts

this.target =
this.startPosition

let direction: Vector

scale the direction
according to the
velocity and add it to
the position

move

this.target =
new Vector

let direction:
Vector

[direction.length >
this.perception]

this.returnToStart()

scale the direction
according to the
velocity and add it to
the position

[player reached
ball]

[let player
of players]

[player.justPossessed]

player.justPossessed =
false

this.inPossession = true
this.justPossessed = true
football.target = football.position
freeze = true
currentPrecision = this.precision

ball

+ startingPosition: Vector
+ target: Vector
- scoreTeam1: number = 0
- scoreTeam2: number = 0

constructor

_position: Vector, _velocity:
number

super(_position, _velocity)
this.target = _position
this.startingPosition = _position

draw

save and translate
the crc2

draw the ball with
crc2.arc()

restore and
resetTransform the
crc2

move

let direction:
Vector

scale the direction
according to the
velocity and add it to
the position

this.target.x += 100
[this.position.x < 0]

[this.position.y
< 0]

this.target.x += 100

[this.position.x > 1000]

this.target.x -= 100

[this.position.y > 600]

this.target.x -= 100

add a point
for Team 2
[ball in goal
of Team 1]

[ball in goal of Team 2]

add a point
for Team 1

find the DOM-element
that displays the score
and change it

return ball to starting
position

alert for the goal

find the DOM-element
that displays the score
and change it

return ball to starting
position

alert for the goal