

## **Anleitung EIA2-Endabgabe**

Willkommen beim Fußballsimulator 2021. Tauche als Trainer in die Welt des Fußballs ein und simuliere Spiele, um so neue Strategien zu entwickeln oder bereits geschehene Situationen besser analysieren zu können.

Primär muss die ZIP-Datei entpackt werden. Dafür sind Programme wie WinRar besonders hilfreich. Ist der Ordner entpackt, sind mehrere Ordner und Dateien drinnen. Mit Klick auf die HTML-Datei wird die Anwendung im Browser gestartet.

Zu sehen sind das Fußballfeld mit den Spielern und Schiedsrichtern. Oberhalb ist das Scoreboard sowie ein Startknopf, links und rechts sind Balken, um die Einstellungen zu ändern.

Um das Spiel zu starten, musst du auf einen gewünschten Punkt auf dem Spielfeld klicken, wo der Ball hingespielt werden soll. Spieler in der Nähe bewegen sich dann zum Ball. Der Ball bleibt stehen, sofern ein Spieler an ihn rangeht oder der Ball ausrollt. Nur bei ersterem Fall kann der Ball durch weiteres Klicken weitergespielt werden. Sind die Spieler nicht mehr „aufmerksam“ auf den Ball, kehren sie zurück zu ihrer Position. Erzielt man ein Tor, erscheint eine Benachrichtigung, welches Team das Tor erzielt hat. Mit Klick auf „Ok“ wird das Spiel wieder angepfiffen.

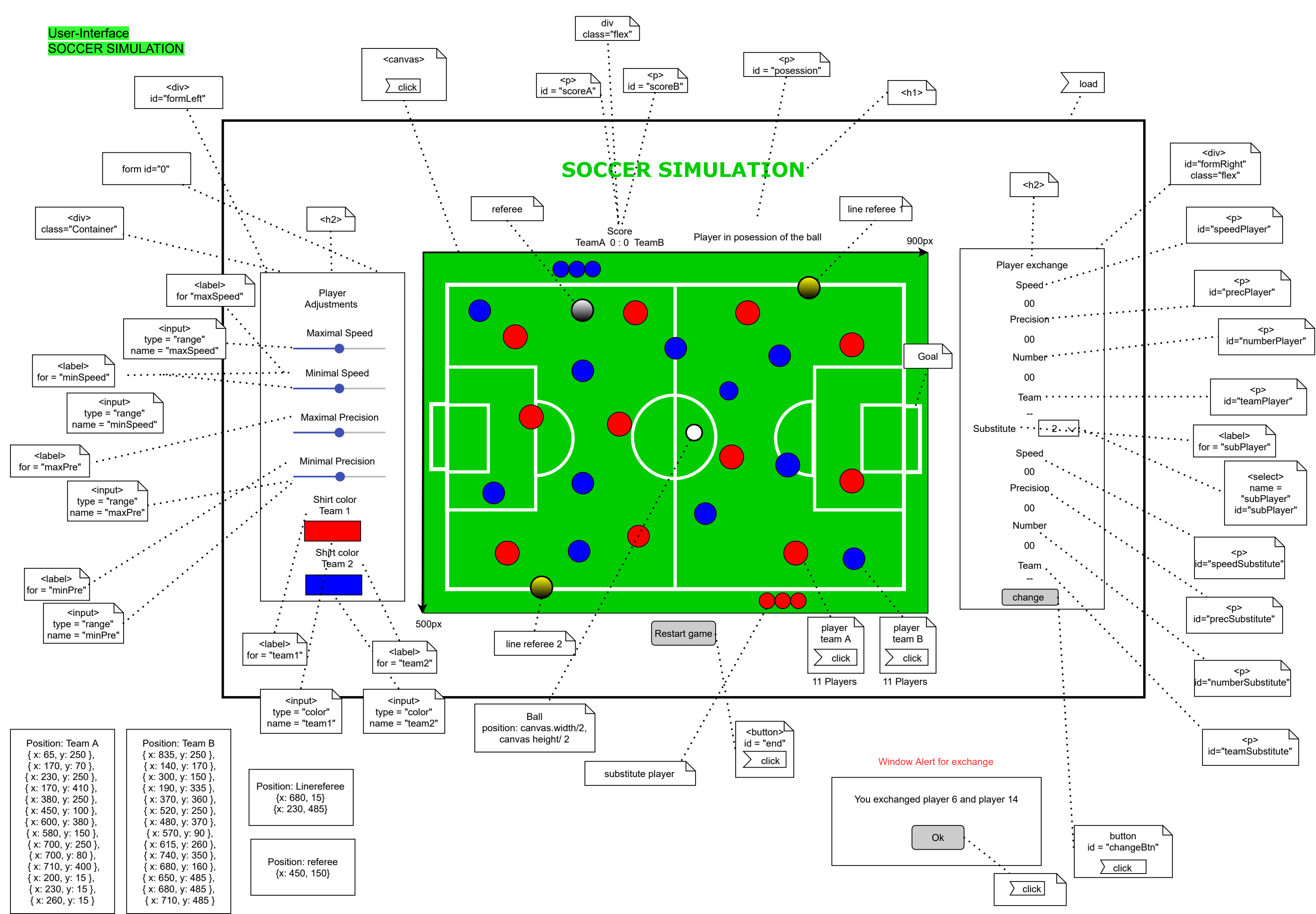
Links in der Tabelle kann man allgemeine Änderungen vornehmen. Mit Höchst- und Minimalgeschwindigkeit werden die Bewegungen der Spieler erhöht bzw. gesenkt. Hierbei muss schlicht der Balken nach links für geringer und rechts für höher geschoben werden. Gleiches gilt für die Präzision der Spieler bei Höchst- und Mindestpräzision. Hierbei wird die Pass- bzw. Schussgenauigkeit adjustiert. Die Farbpaletten drunter sind für das Ändern der Trikotfarben zuständig. Einstellungen werden hierbei nur übernommen, sofern man die Palette wieder schließt.

Auf der rechten Seite kann man Spieler auswechseln, die bereits am Fußballrand stehen. Mit den Pfeiltasten sucht man sich einen Spieler aus und wählt dann in der Tabelle im Dropdown-Menü einen Auswechselspieler aus. Auf den Button „Change“ werden diese dann ausgewechselt.

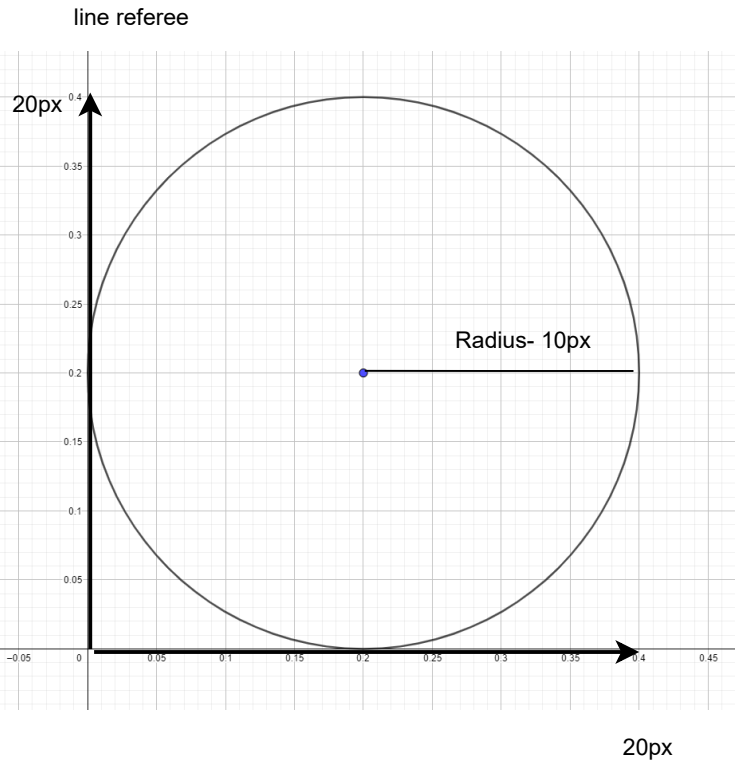
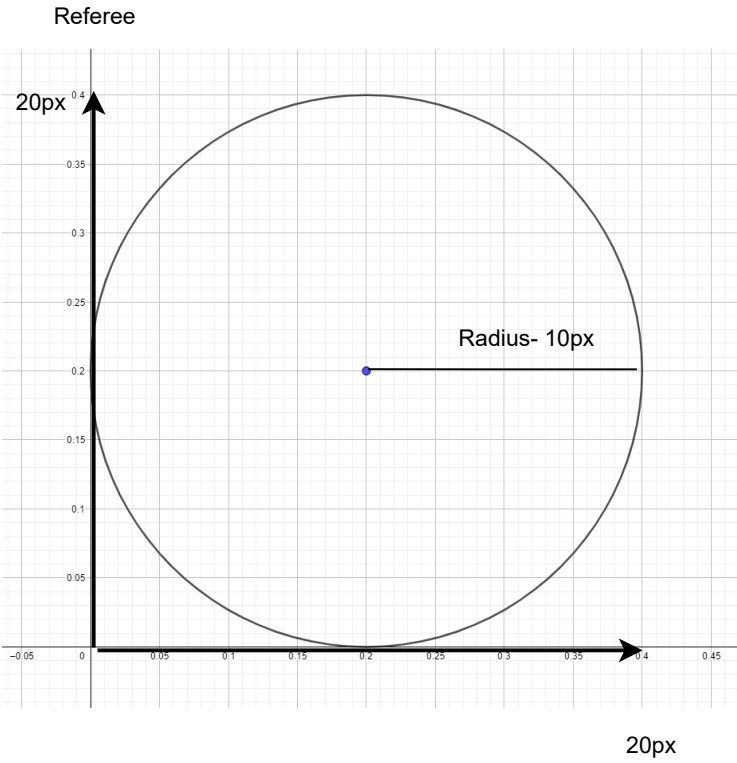
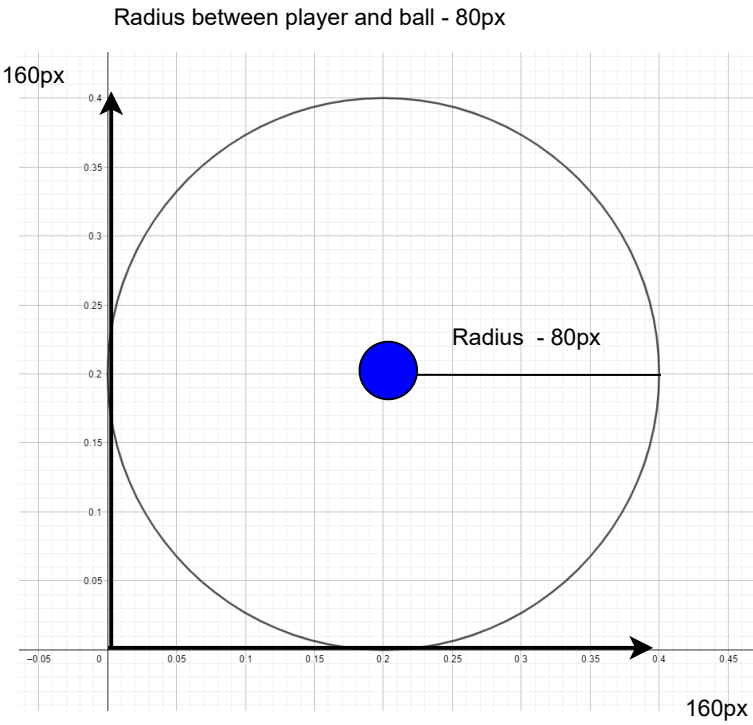
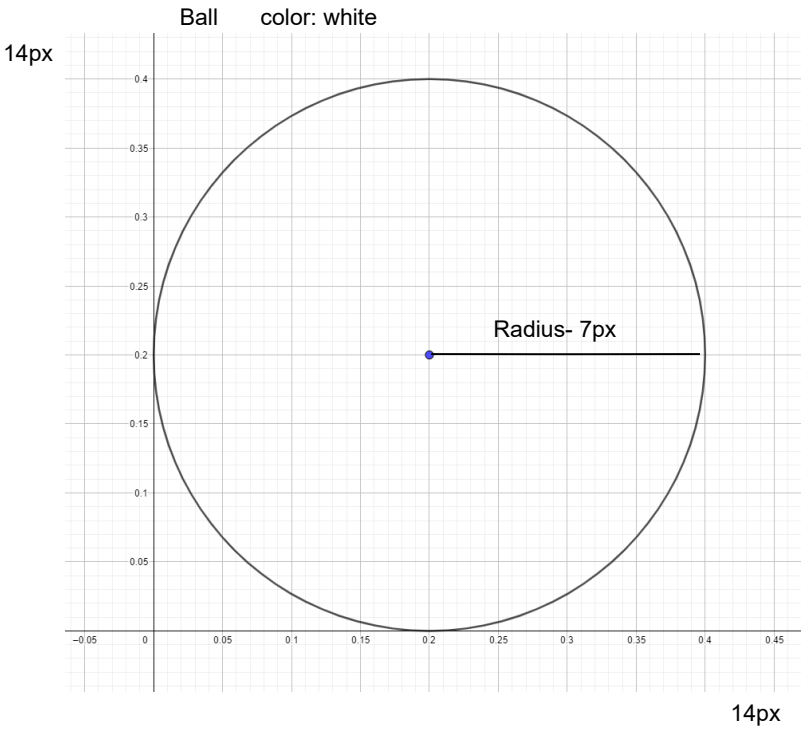
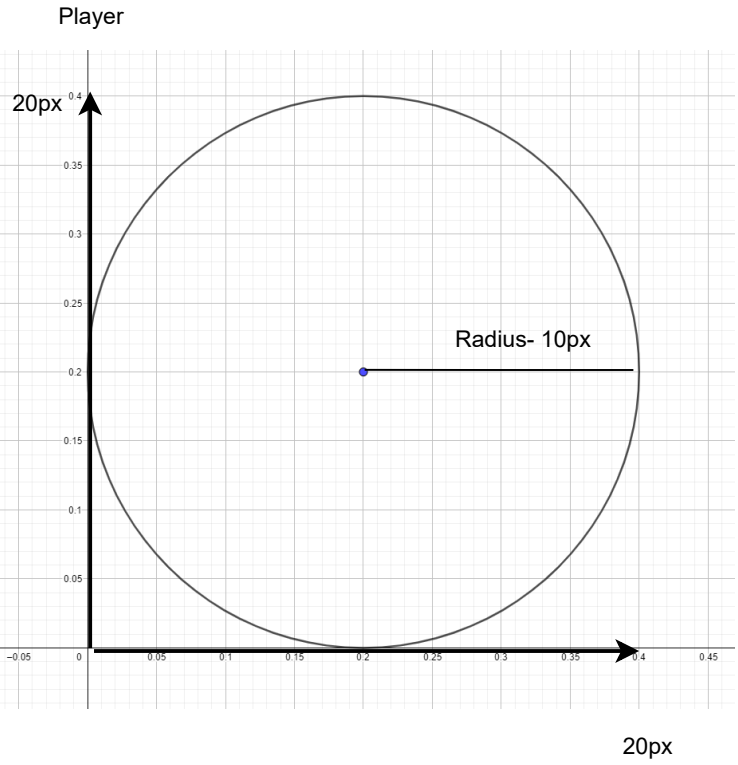
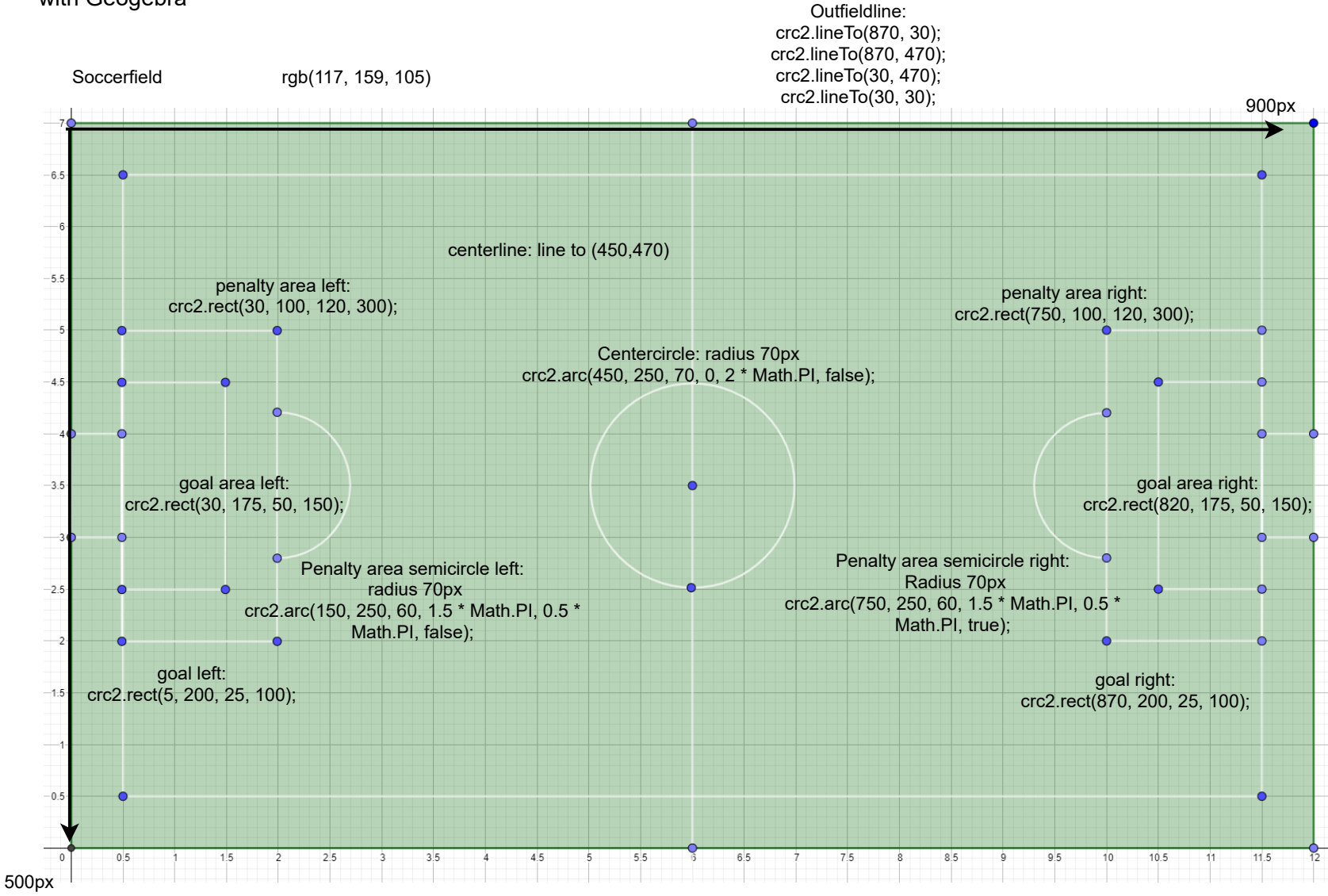
Möchte man das Spiel neu starten, drückt man auf den „Restart“ Button, unterhalb des Spielfeldes. Außerdem wird der Torzähler bei einem Tor aktualisiert und ständig der Ballbesitz angezeigt.

Die Endabgabe entstand durch die Zusammenarbeit mit Mona Kabelka, Huu Thien Phan Ngoc, Mariia Kolkutova und Timur Yildirim.

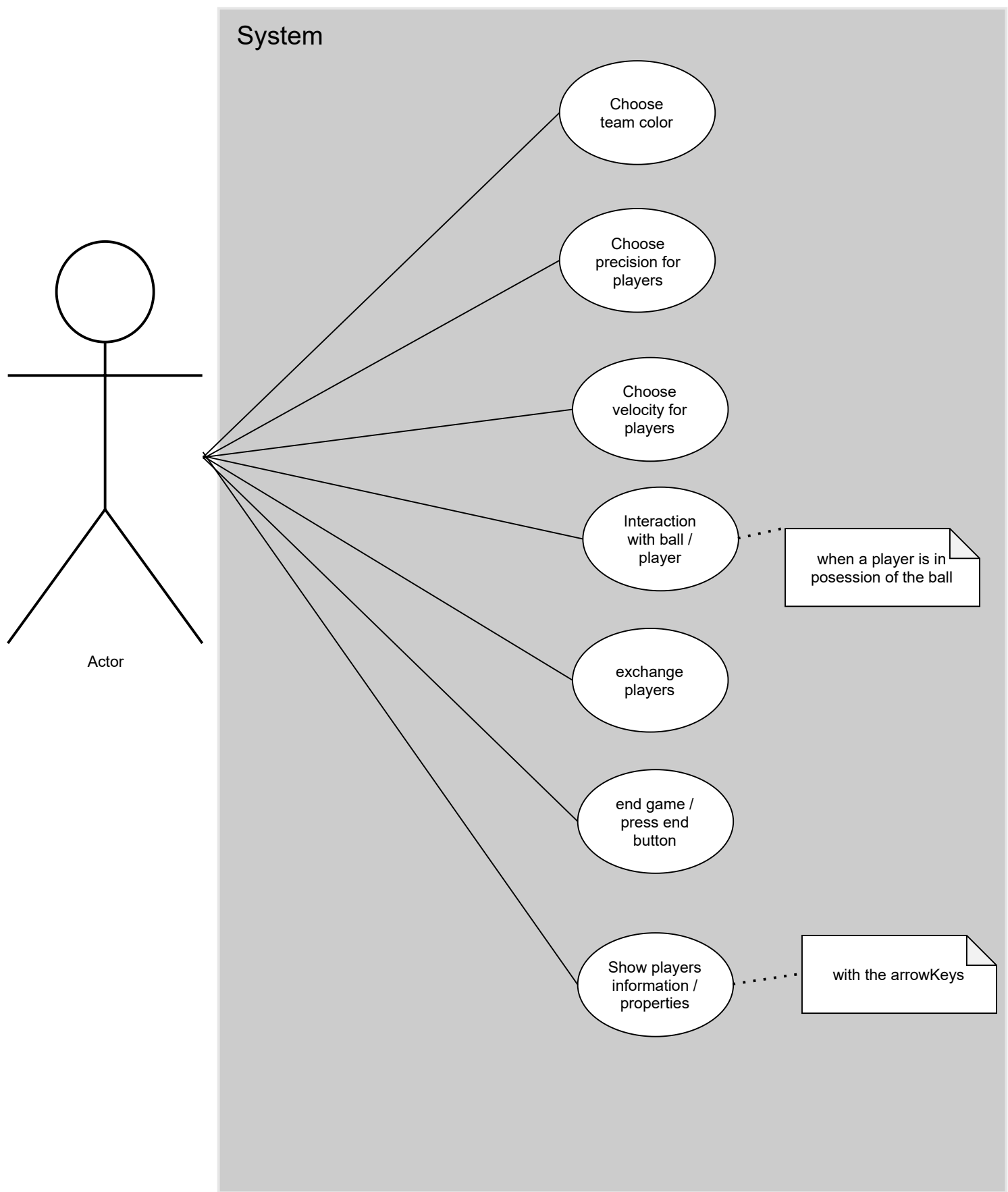
User-Interface  
SOCCER SIMULATION



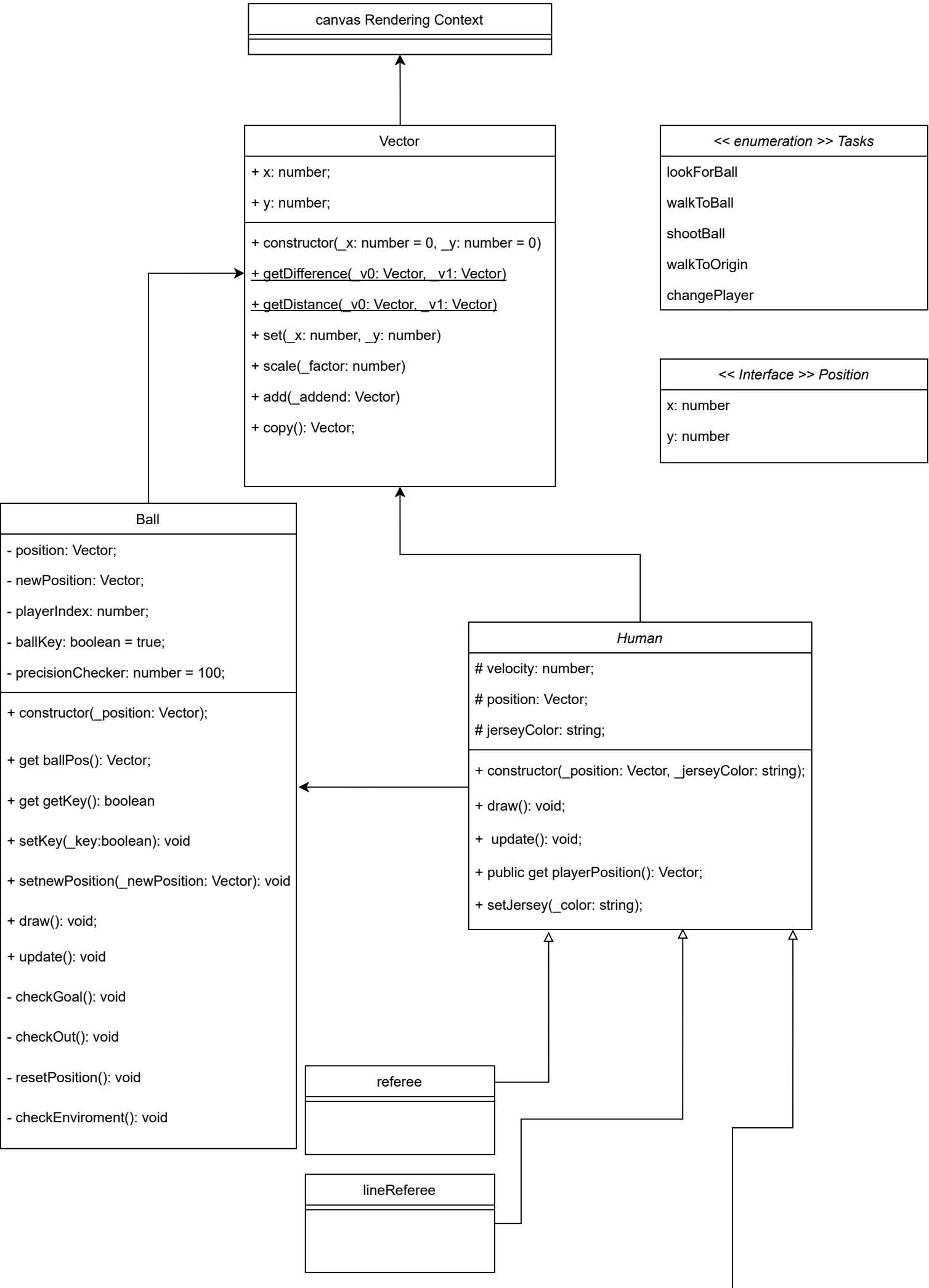
Measurements  
with Geogebra



# Soccer Simulation: Use-Case-Diagram



Klassendiagramm



Player
<div>- task: Task; - origin: Vector; - precision: number; - radius: number = 80; - jerseyNumber: number; - distancePlayerBall: number; - onField: boolean;</div>
<div>+ constructor(_position: Vector, _onField: boolean, _jerseyColor: string, _team: string) + get jerseyNumberPlayer(): number  + get playerSpeed(): number + get distance(): number + get playerPrecision(): number + get playerOnField(): number + get playerTeam(): number + setOnField(_onField): void  + setProperties(_minSpeed: number, _maxSpeed: number, _minPrecision: number, _maxPrecision: number): void  + setDistance(): void + drawRadius(): void  - movePlayer(_position: Vector): void</div>

# Activity-Diagram: main



```
let canvas: HTMLCanvasElement =  
<HTMLCanvasElement> document.querySelector("canvas");  
let crc2 CanvasRenderingContext2D = <CanvasRenderingContext2D> canvas.getContext("2d");
```



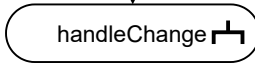
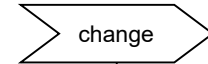
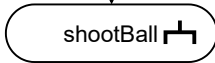
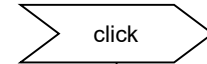
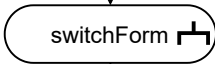
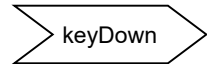
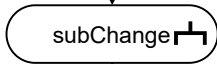
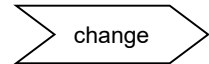
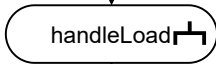
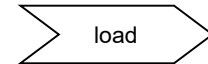
```
export enum Task{};  
let imageData: ImageData;
```



```
let ball: Ball;  
let key: boolean;  
let animationKey: boolean = true;  
let shootKey: boolean = false;  
let humans: Human[] = [];  
let scoreA: number = 0;  
let scoreB: number = 0;
```



```
let form: HTMLDivElement;  
let playerNumberDOMELEMENT: HTMLParagraphElement;  
let teamDOMELEMENT: HTMLParagraphElement;  
let teamADOMELEMENT: HTMLButtonElement;  
let teamBDOMELEMENT: HTMLButtonElement;  
let speedPlayer: HTMLParagraphElement;  
let precPlayer: HTMLParagraphElement;  
let numberPlayer: HTMLParagraphElement;  
let teamPlayer: HTMLParagraphElement;  
let speedSub: HTMLParagraphElement;  
let precSub: HTMLParagraphElement;  
let numberSub: HTMLParagraphElement;  
let teamSub: HTMLParagraphElement;  
let subPlayerDOMELEMENT: HTMLSelectElement;  
let scoreADOMELEMENT: HTMLElement;  
let scoreBDOMELEMENT: HTMLElement;  
let changeBtn: HTMLButtonElement;  
let scoreA: number = 0;  
let scoreB: number = 0;  
let posession: HTMLParagraphElement;
```

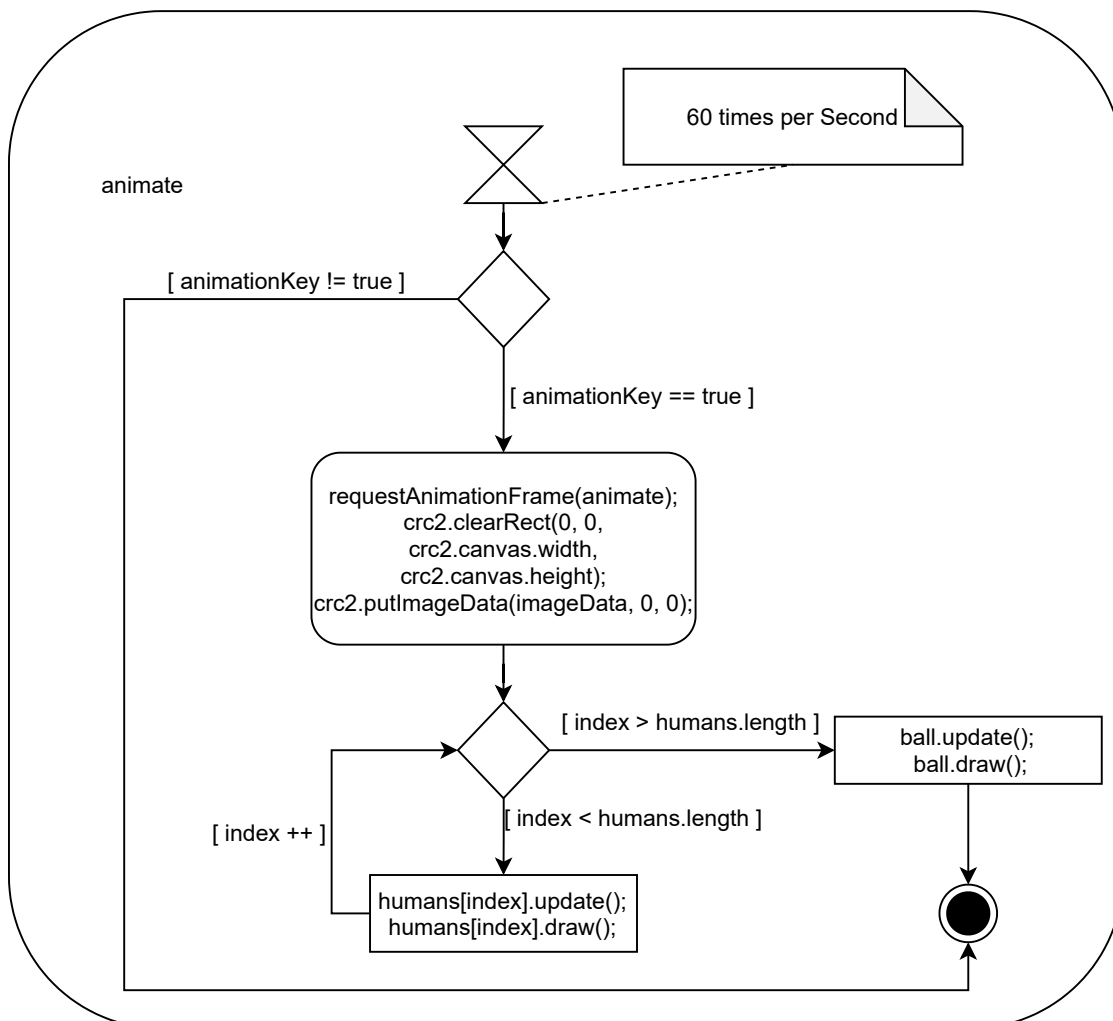
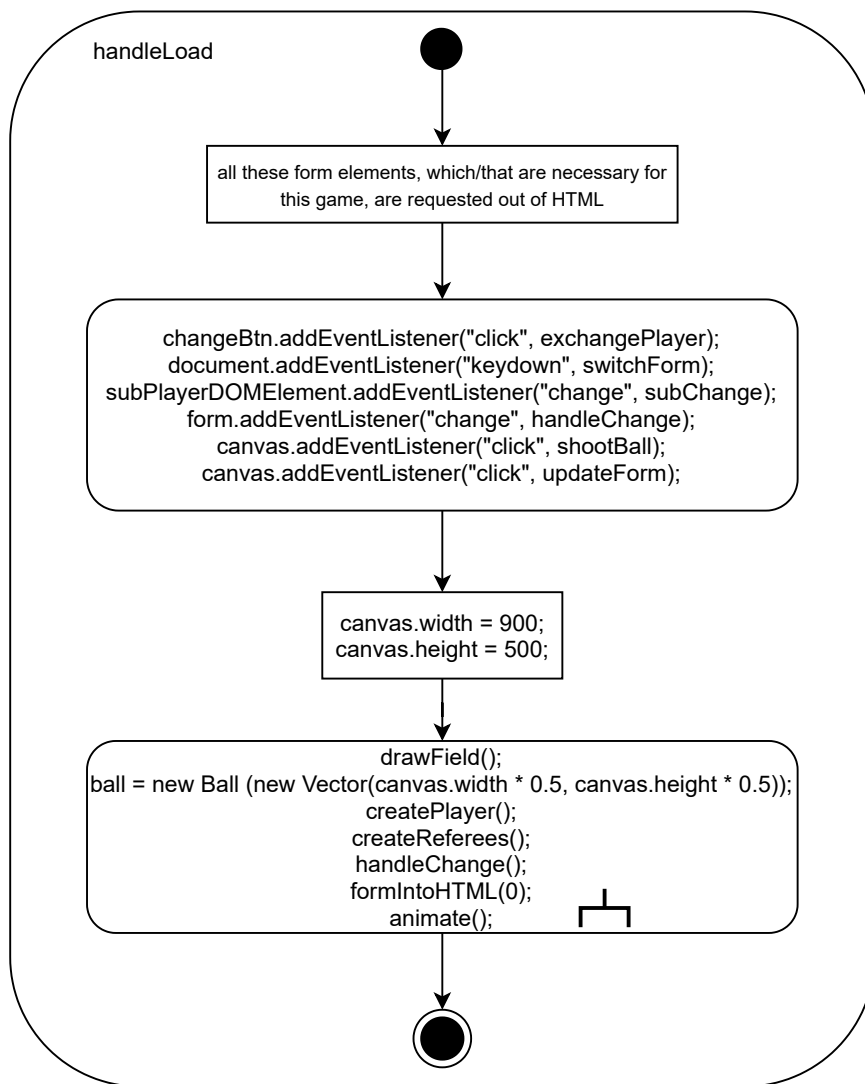


createReferees



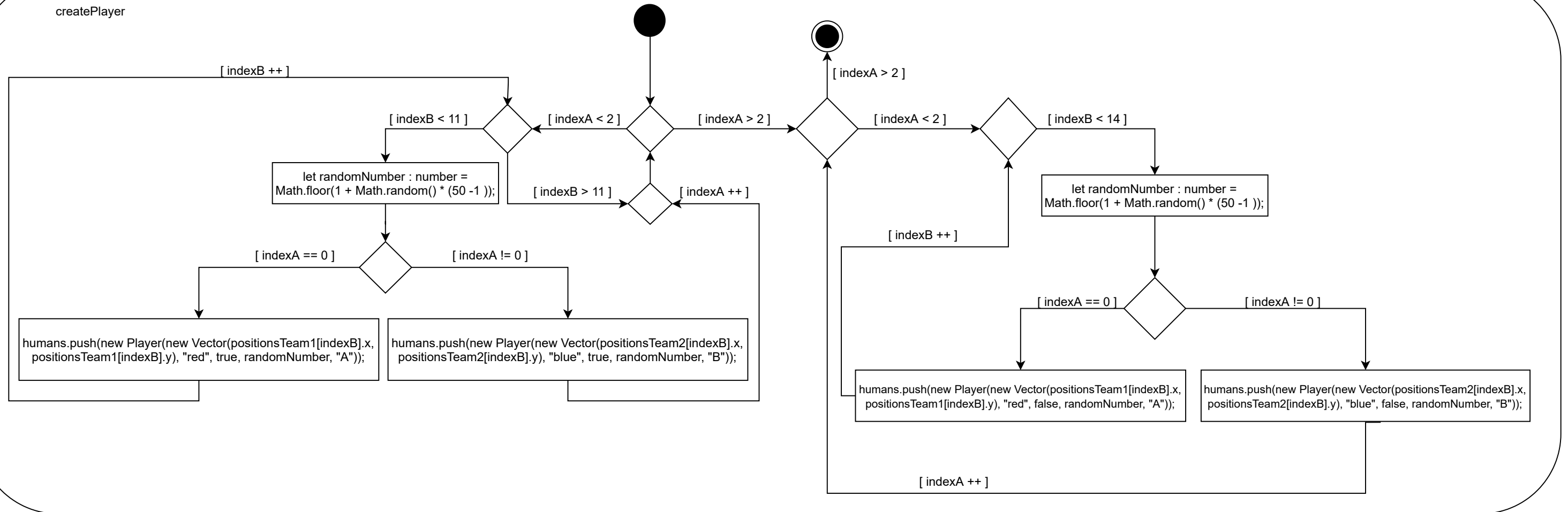
```
players.push(new Referee(new Vector(450, 150), "white"));  
players.push(new Linereferee(new Vector(680, 15), "pink"));  
players.push(new Linereferee(new Vector(230, 485), "pink"));
```

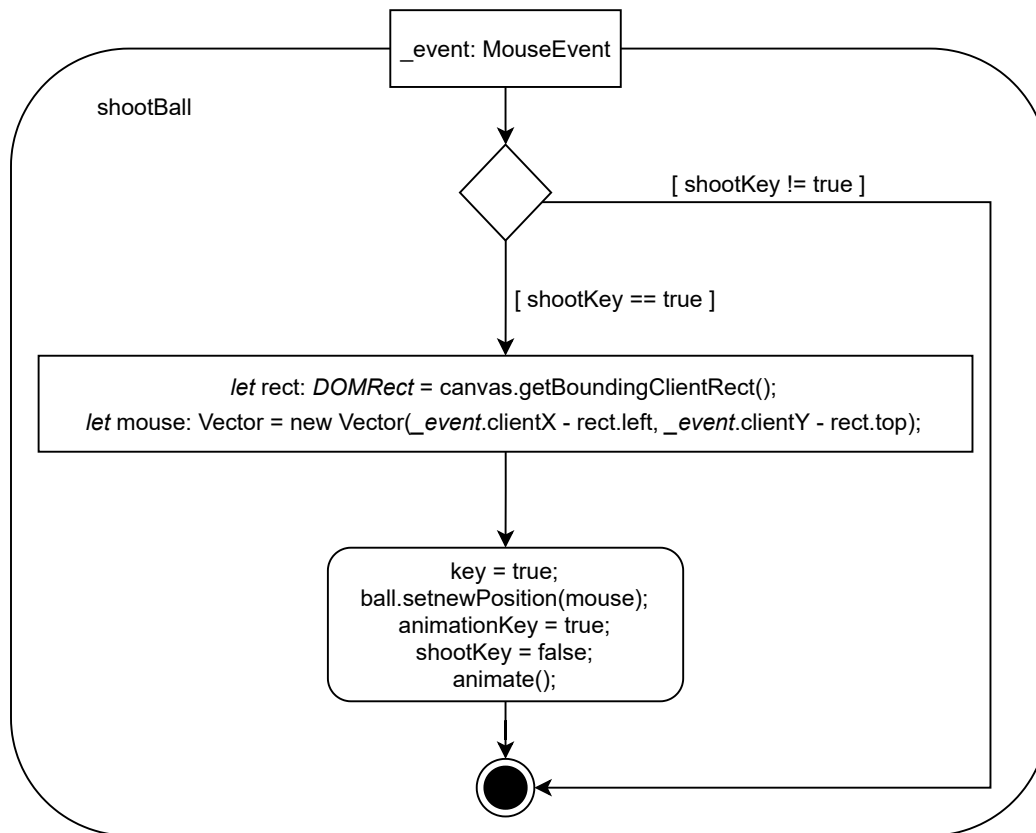




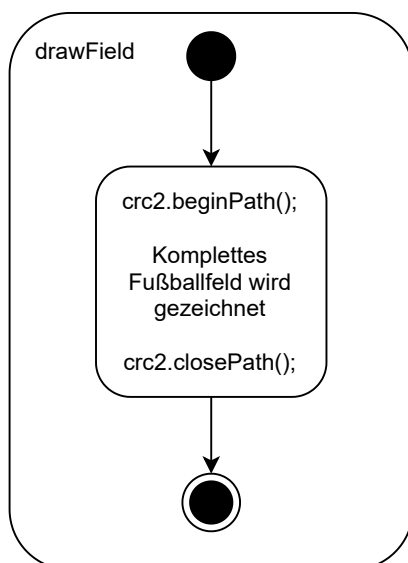


createPlayer

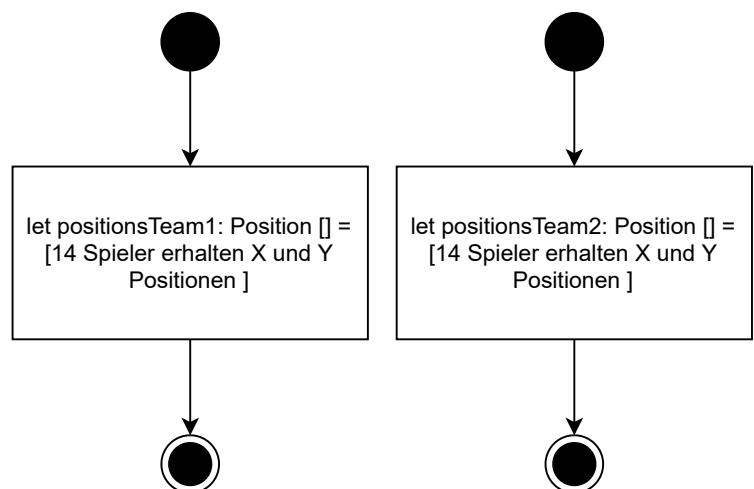




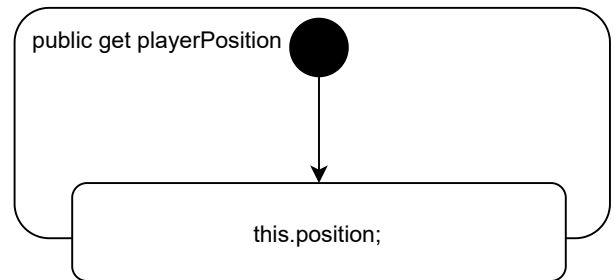
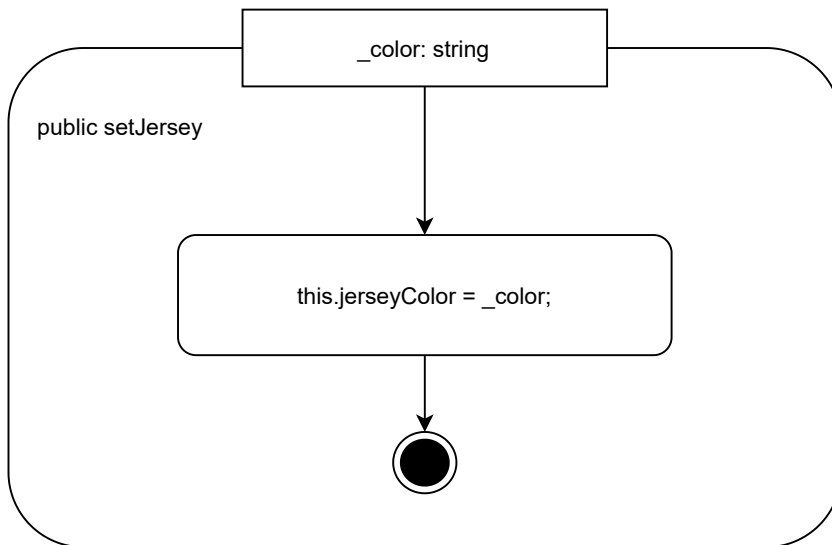
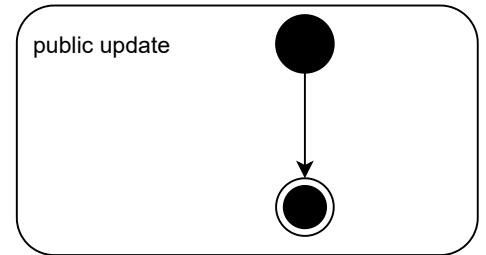
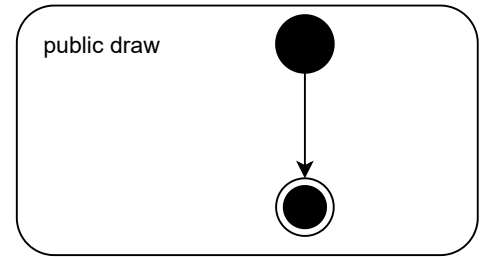
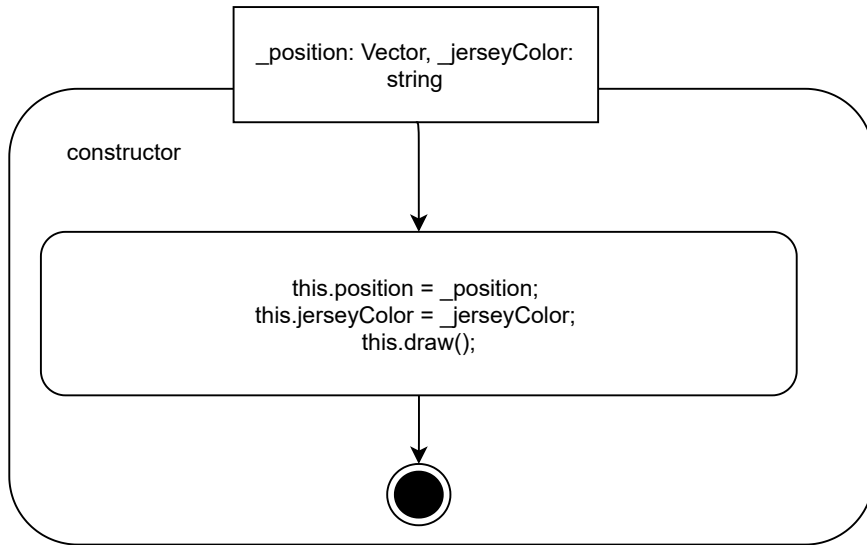
## Activity-Diagram: footballField



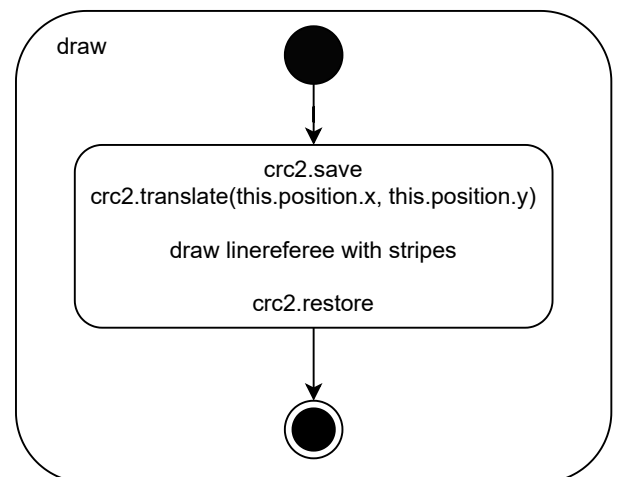
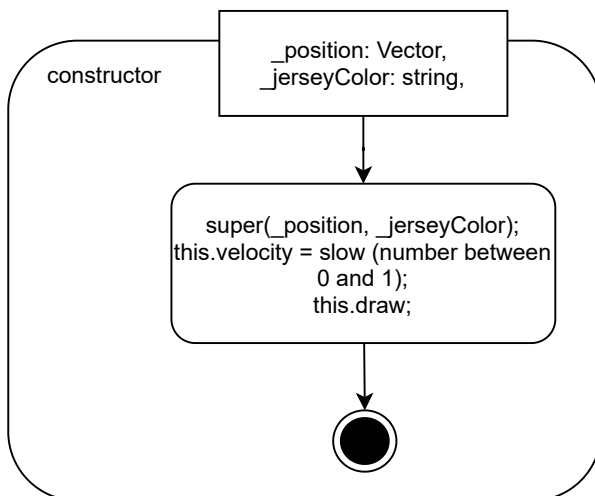
## Activity-Diagram: position

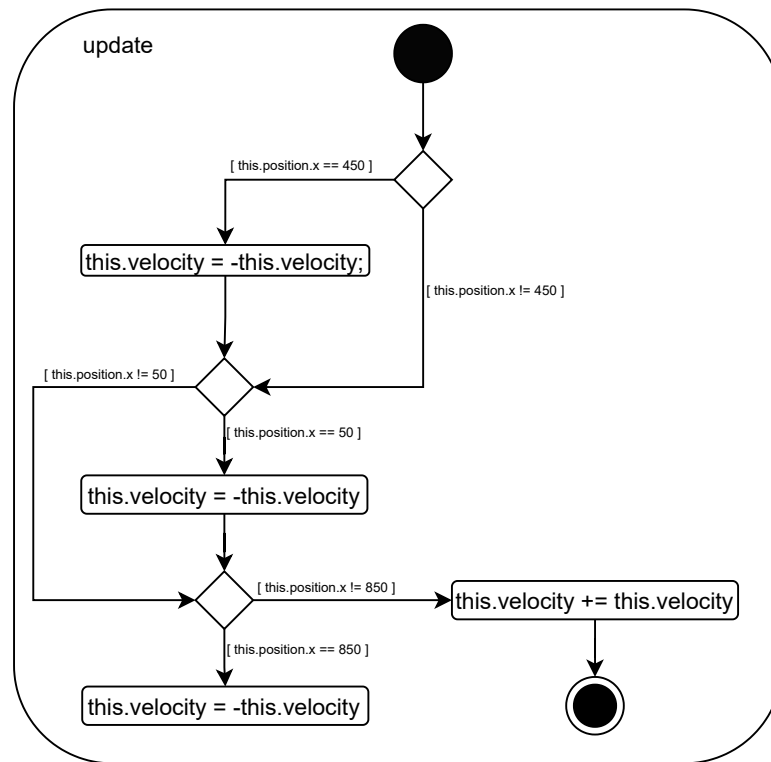


# Activity-Diagram: human

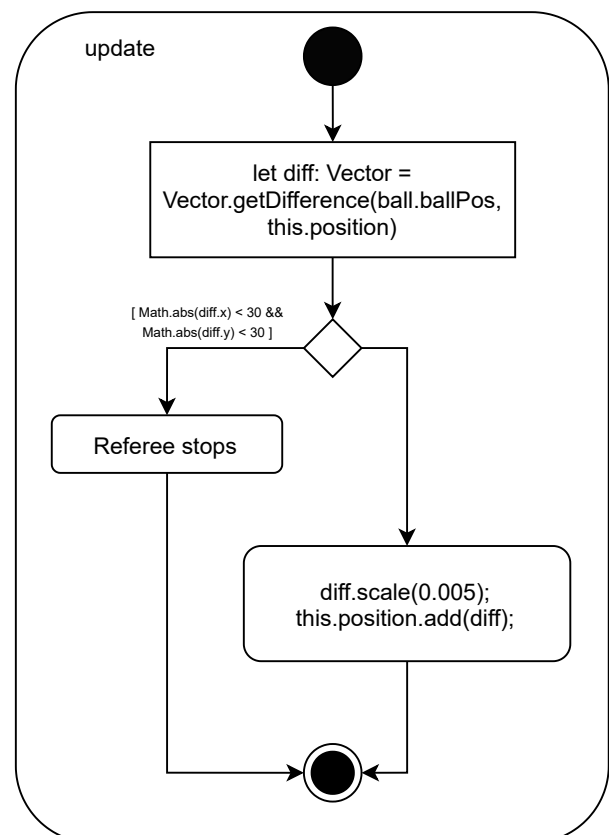
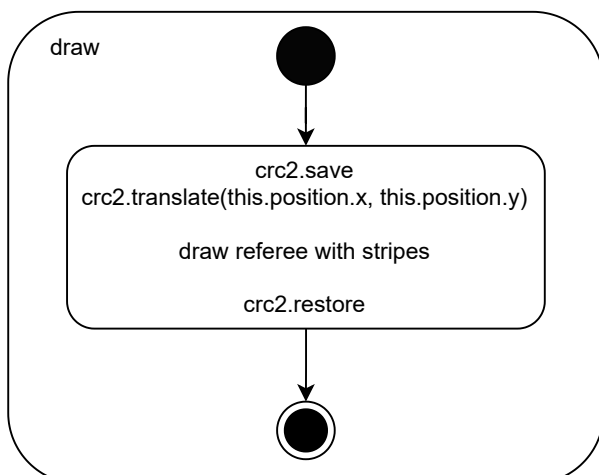
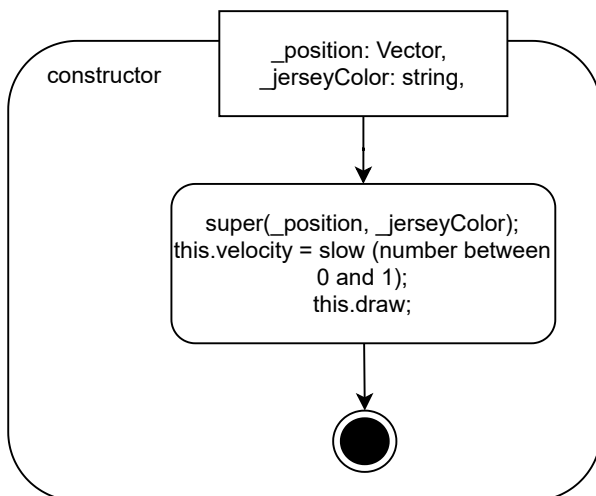


# Activity-Diagram: lineReferee

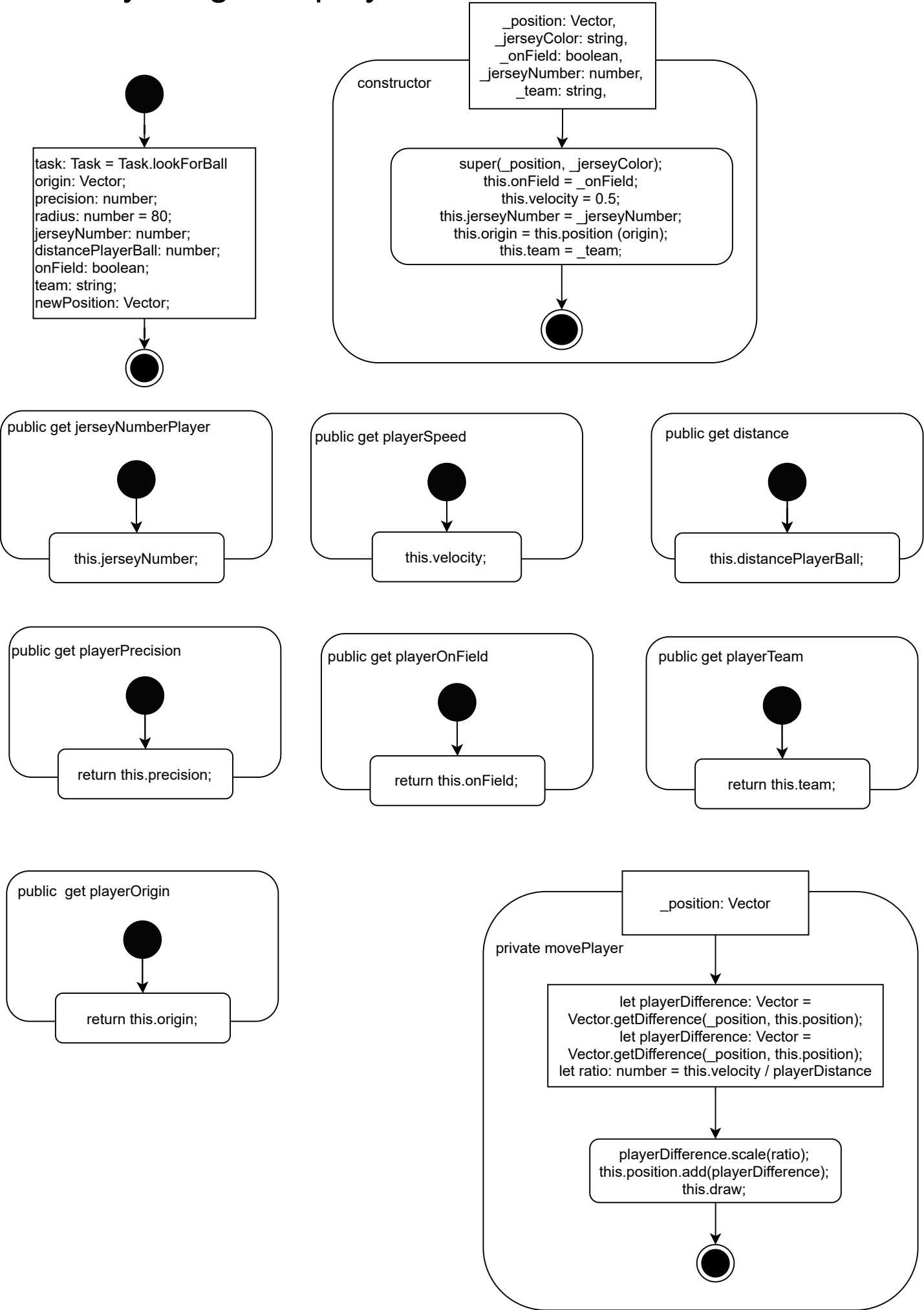


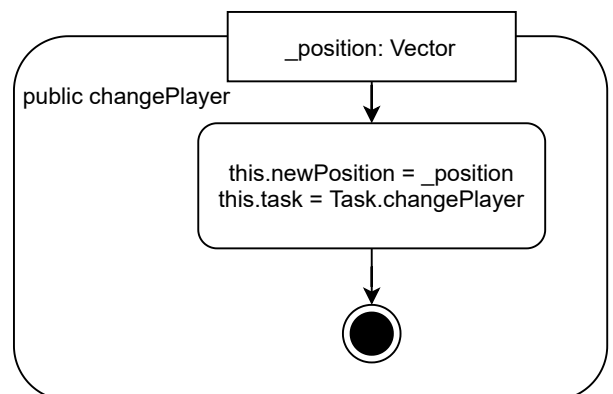
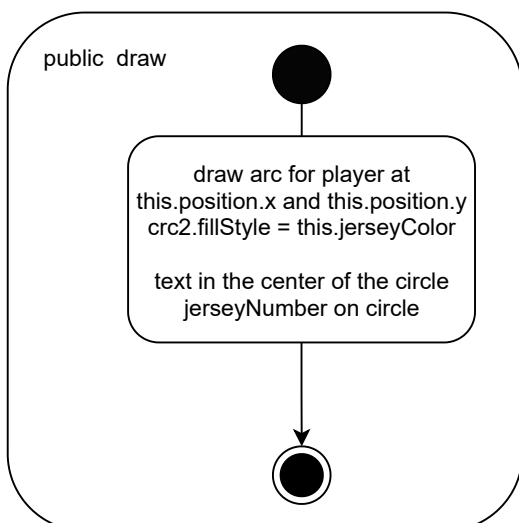
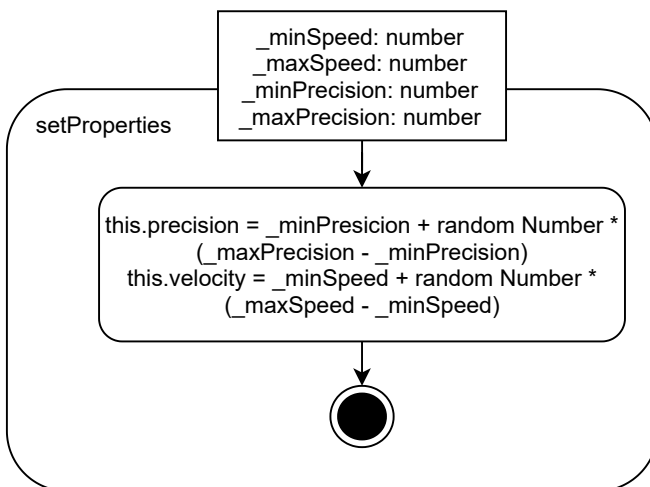
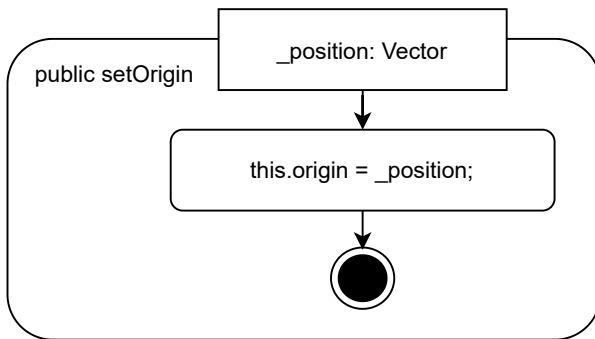
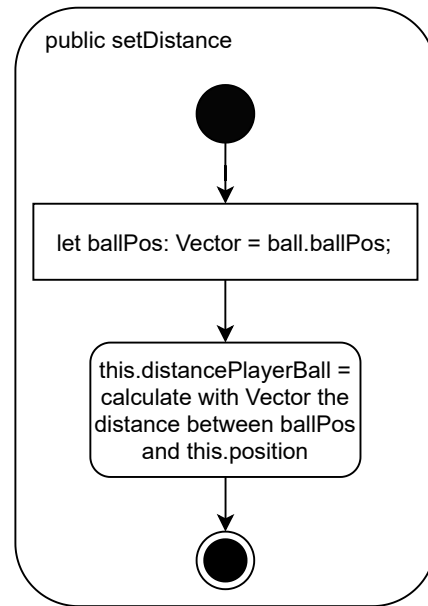
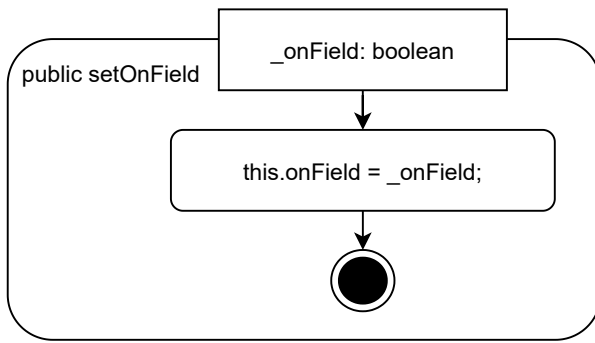


## Activity-Diagram: Referee

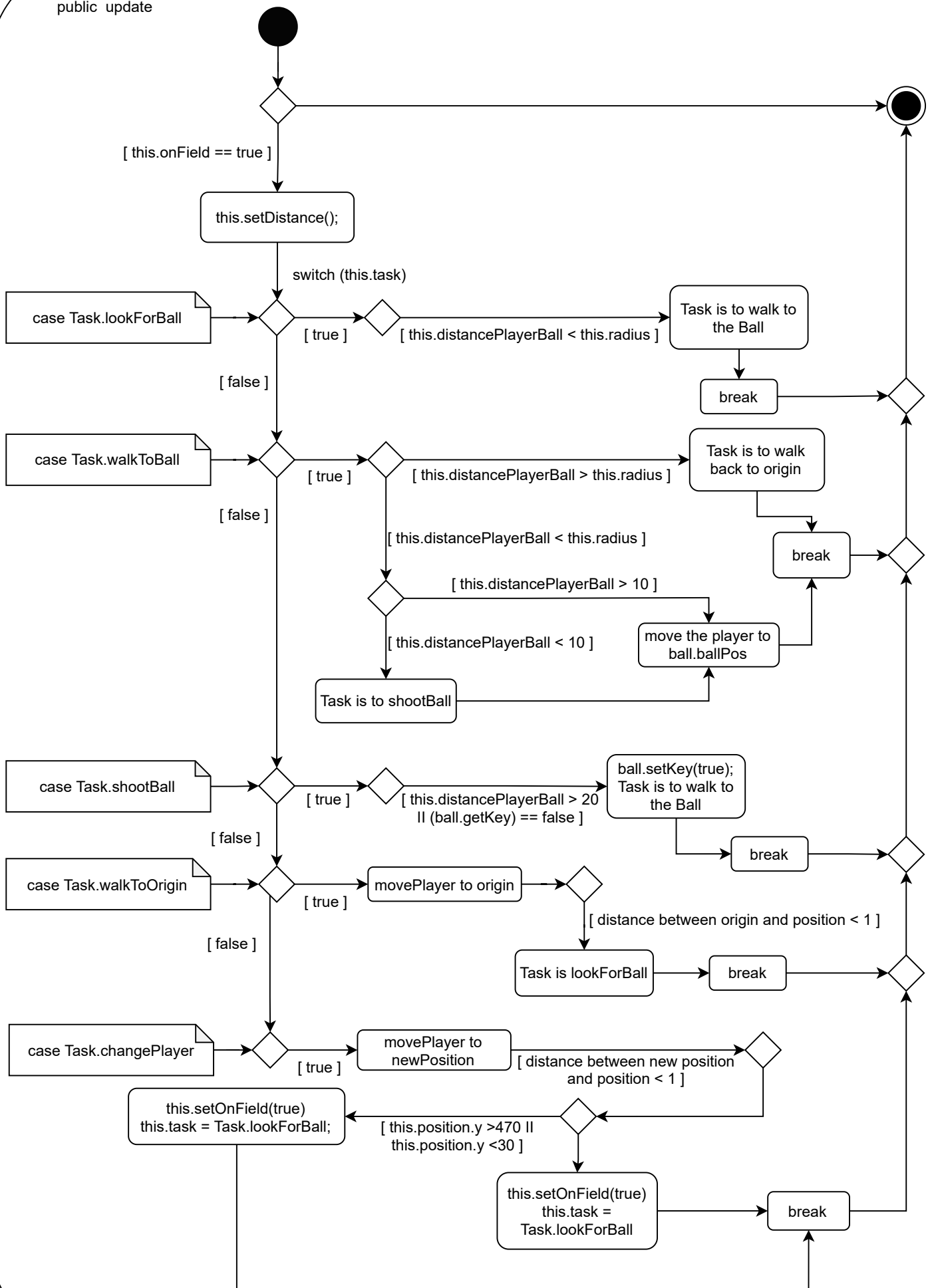


# Activity-Diagram: player

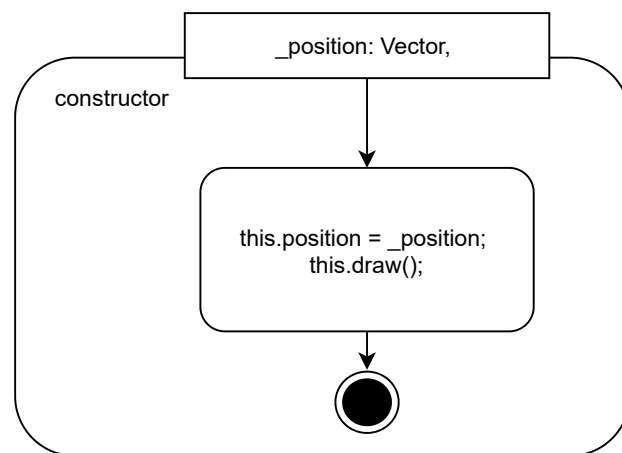
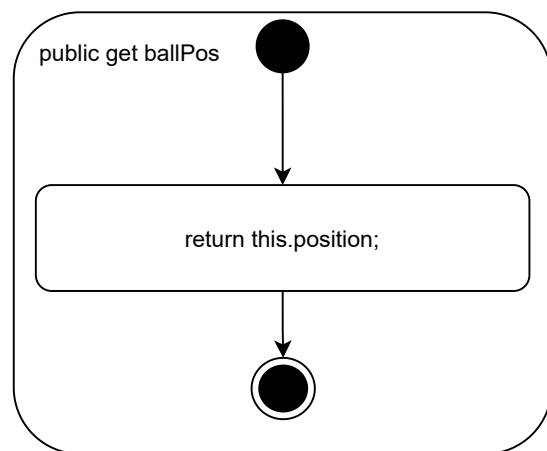
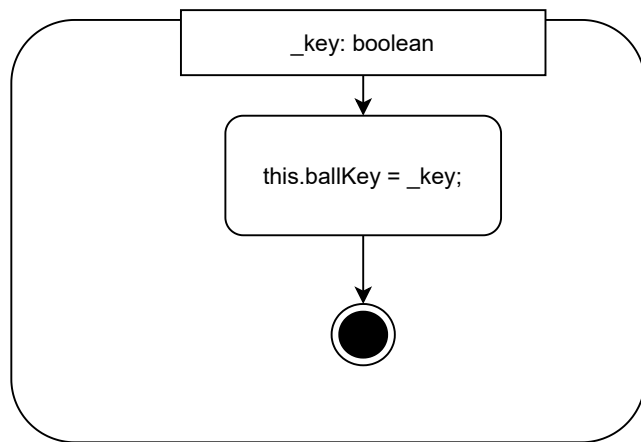
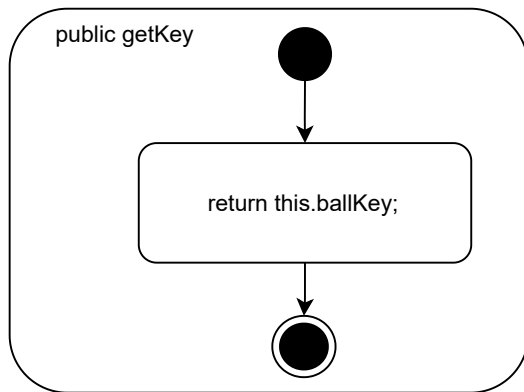




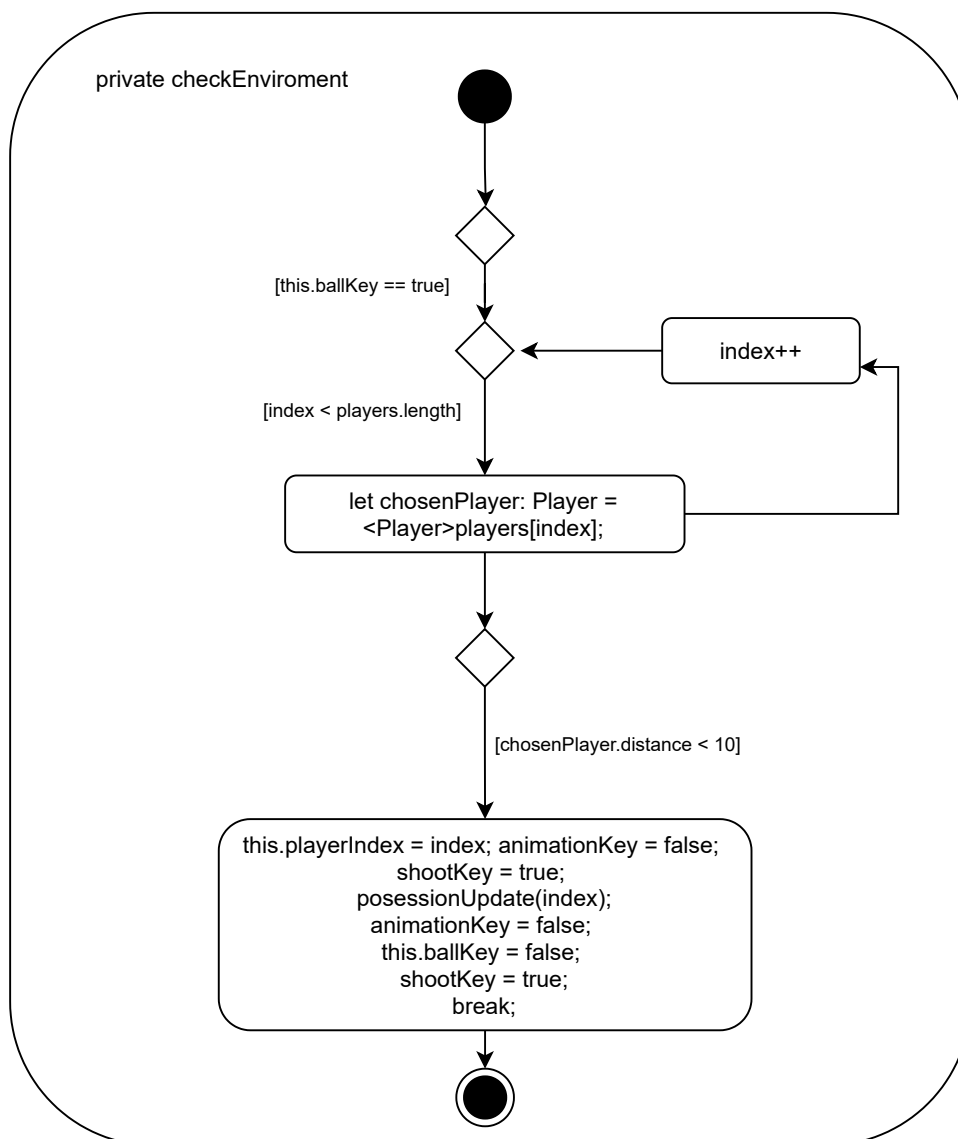
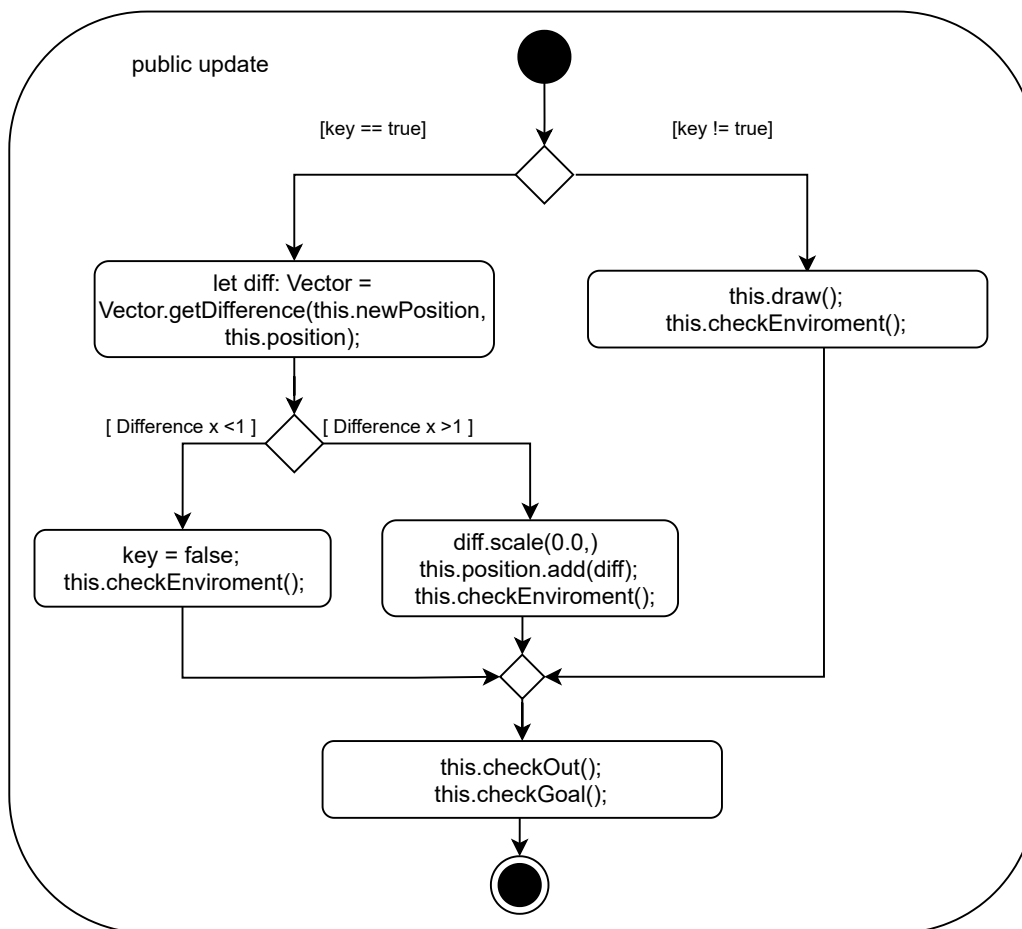
public update



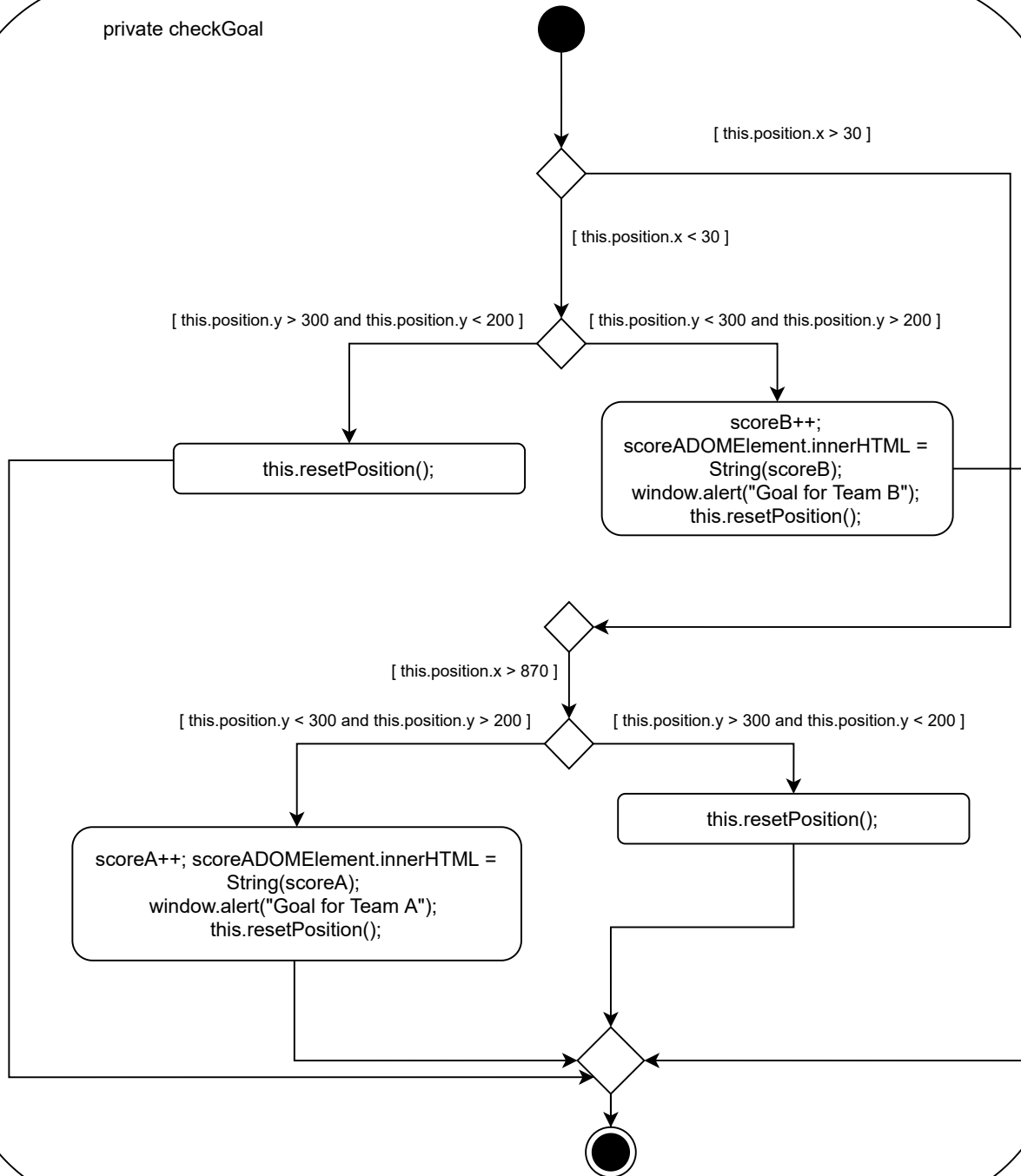
# Activity-Diagram: ball

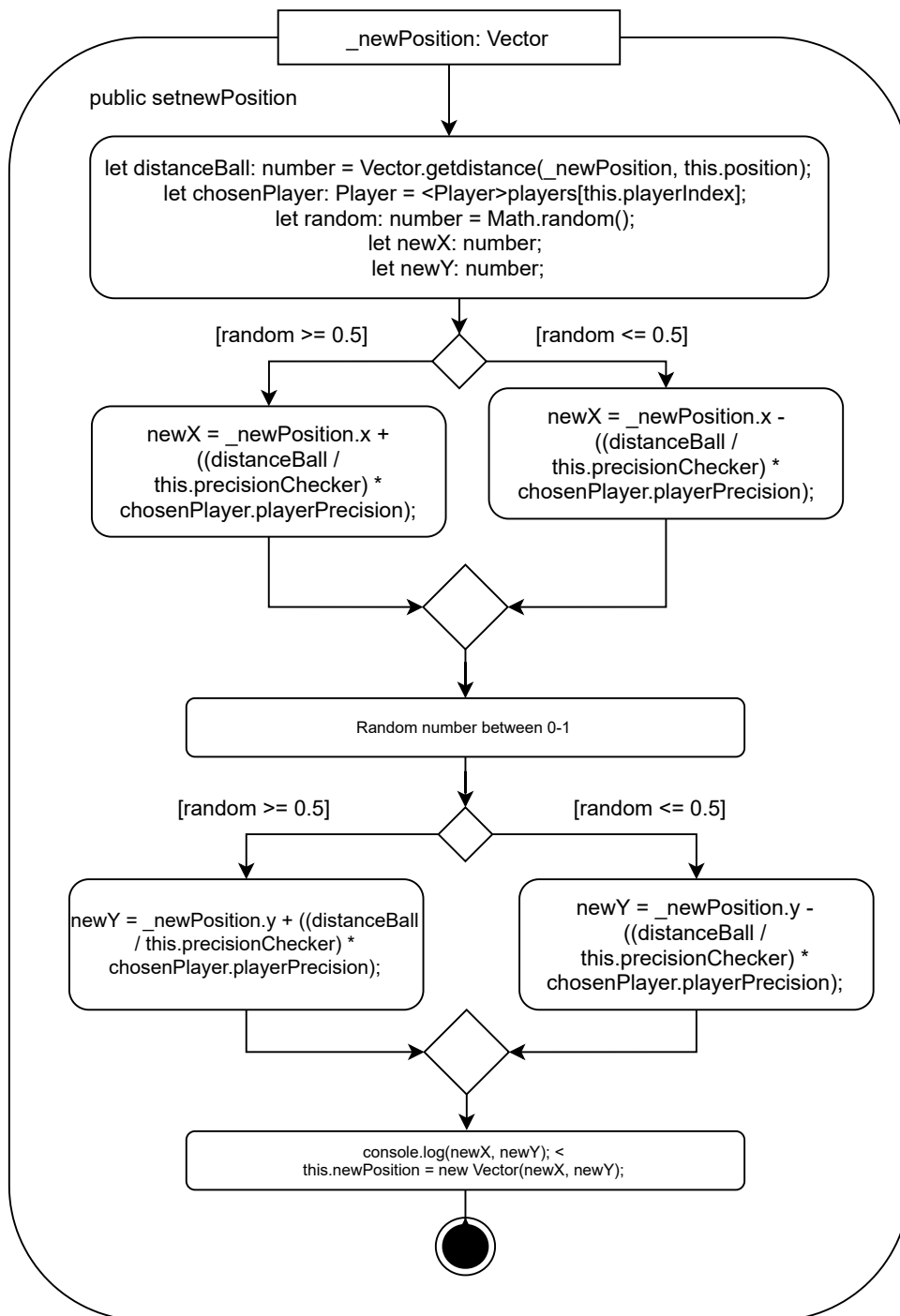




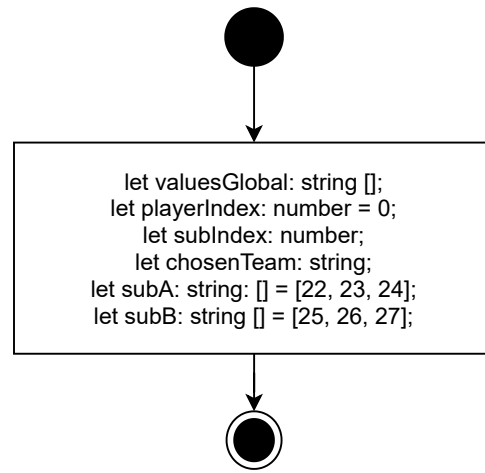


private checkGoal

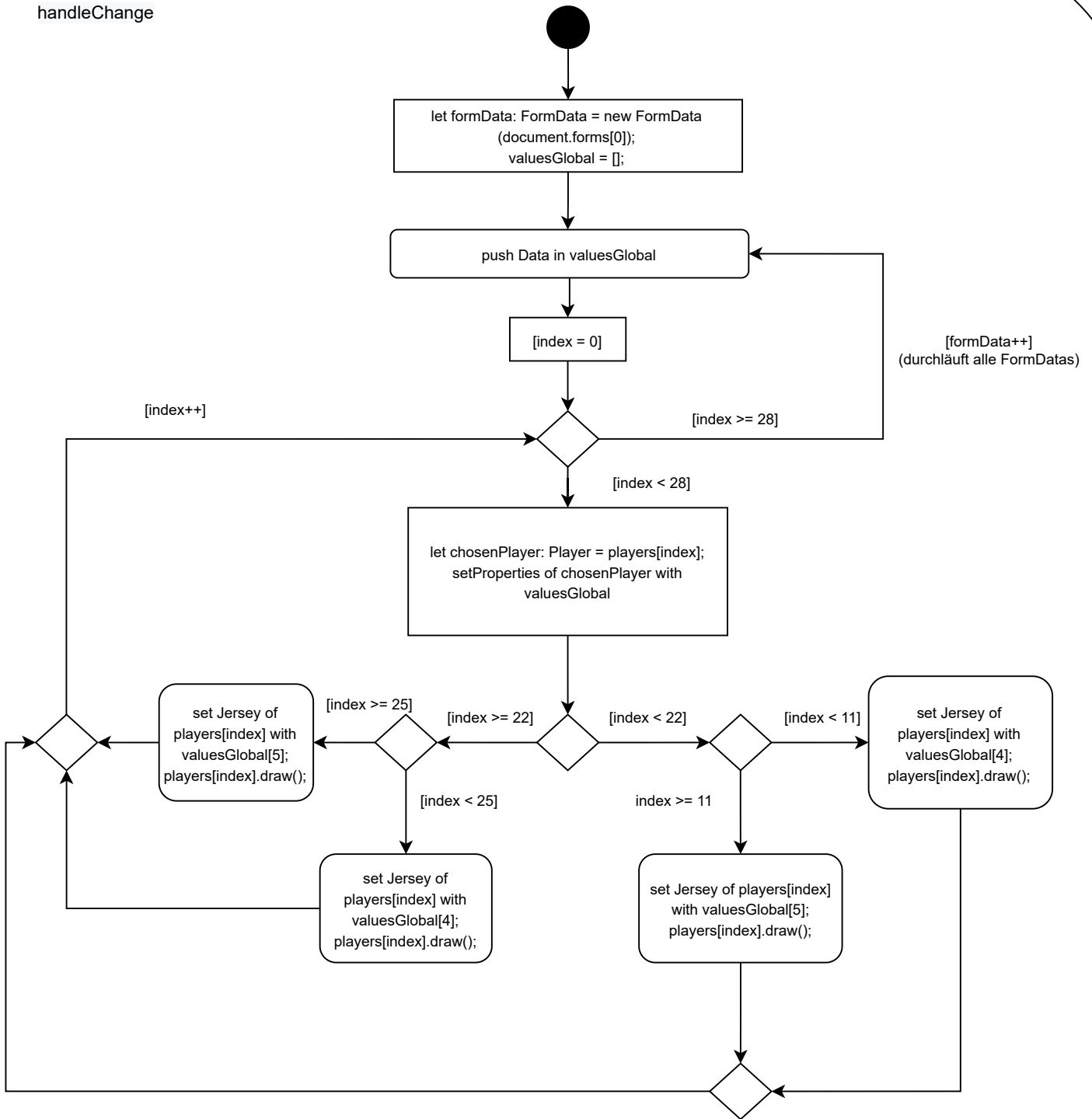


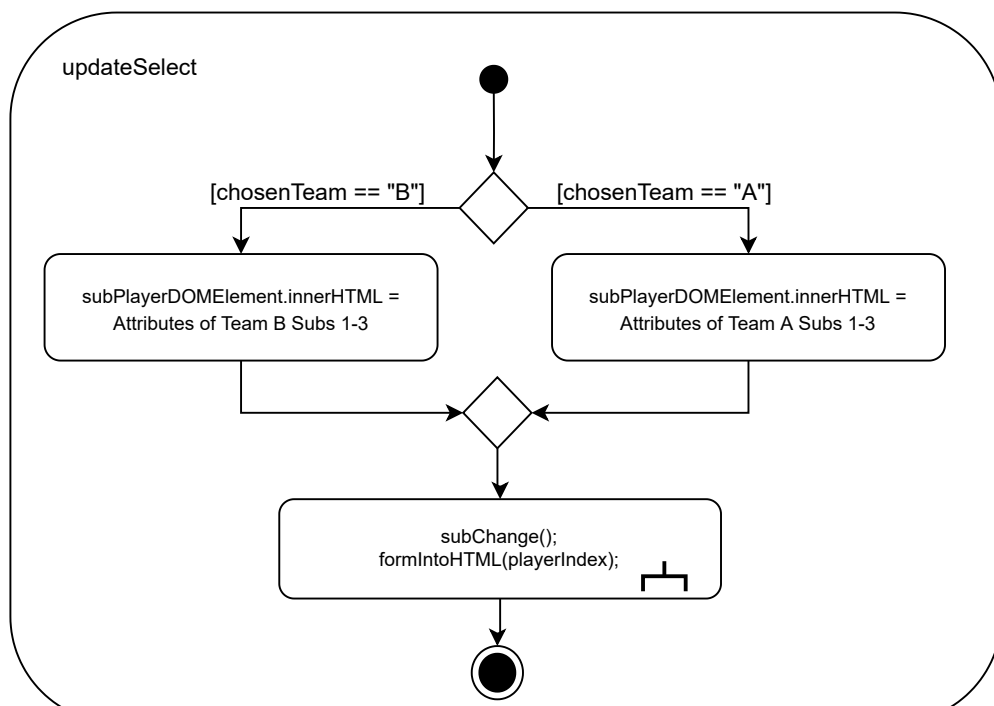
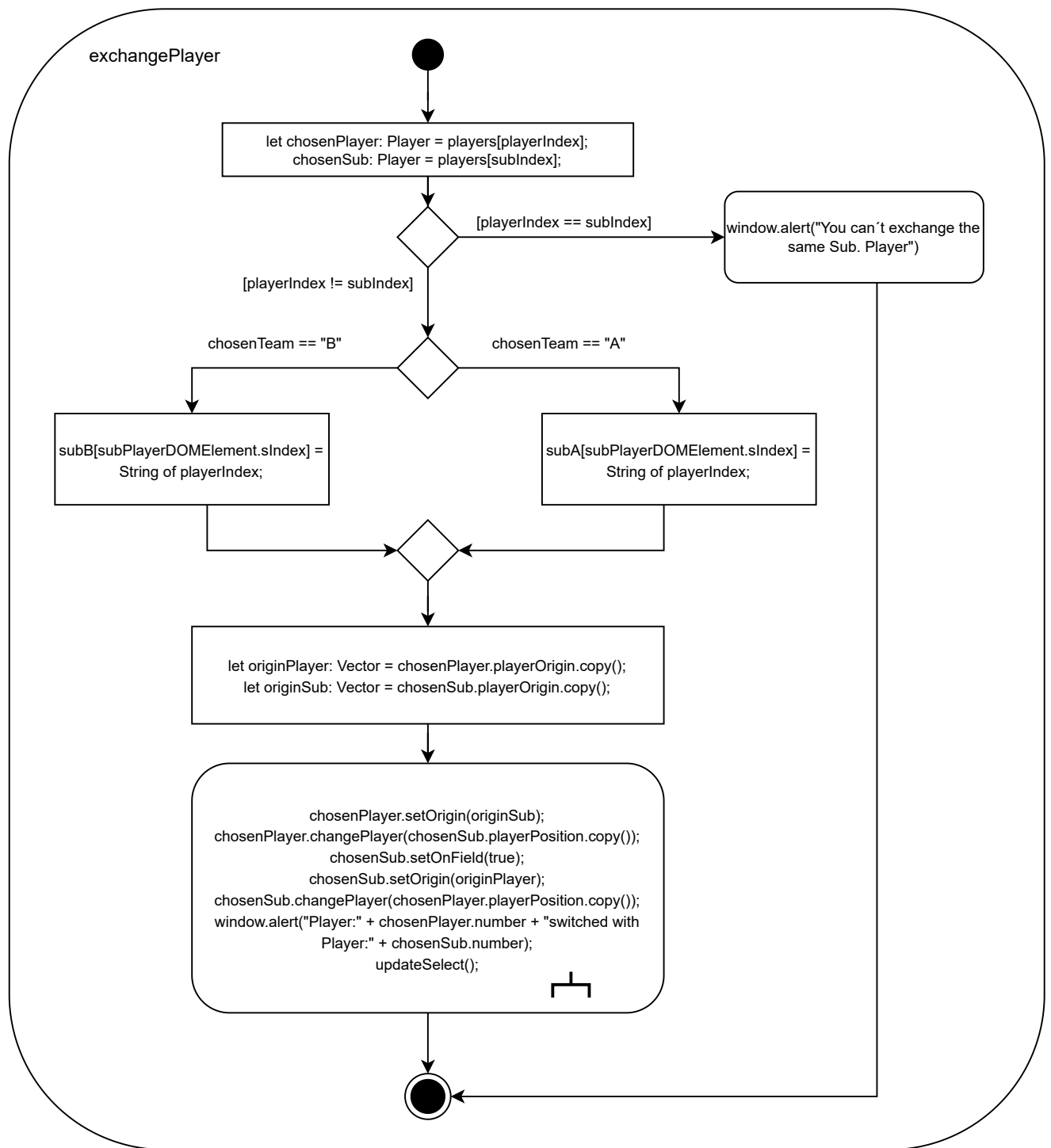


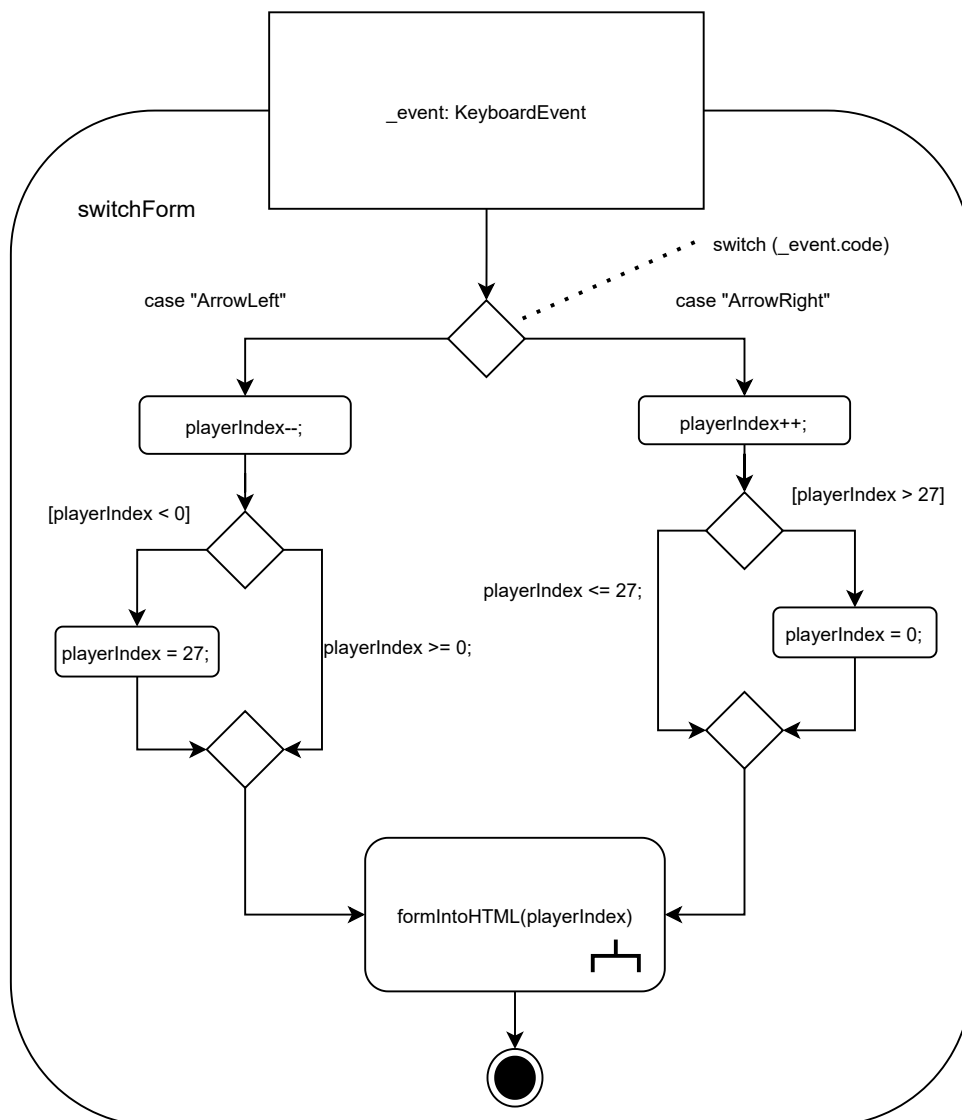
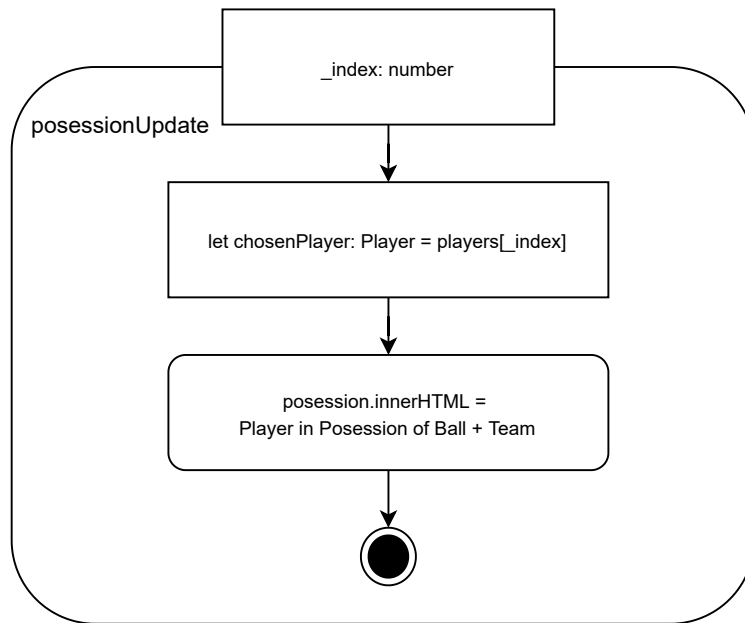
# Activity-Diagram: forms

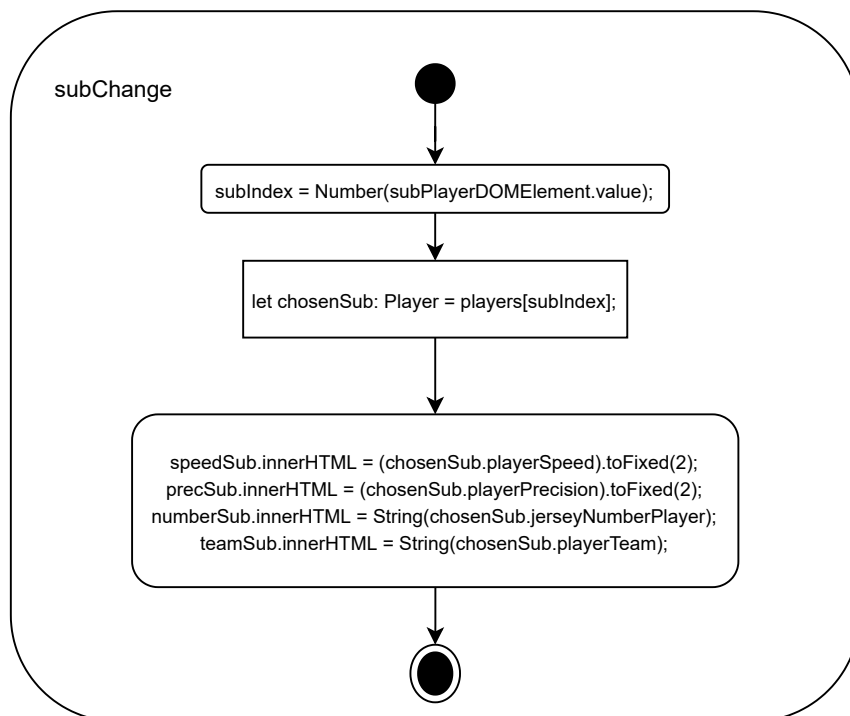
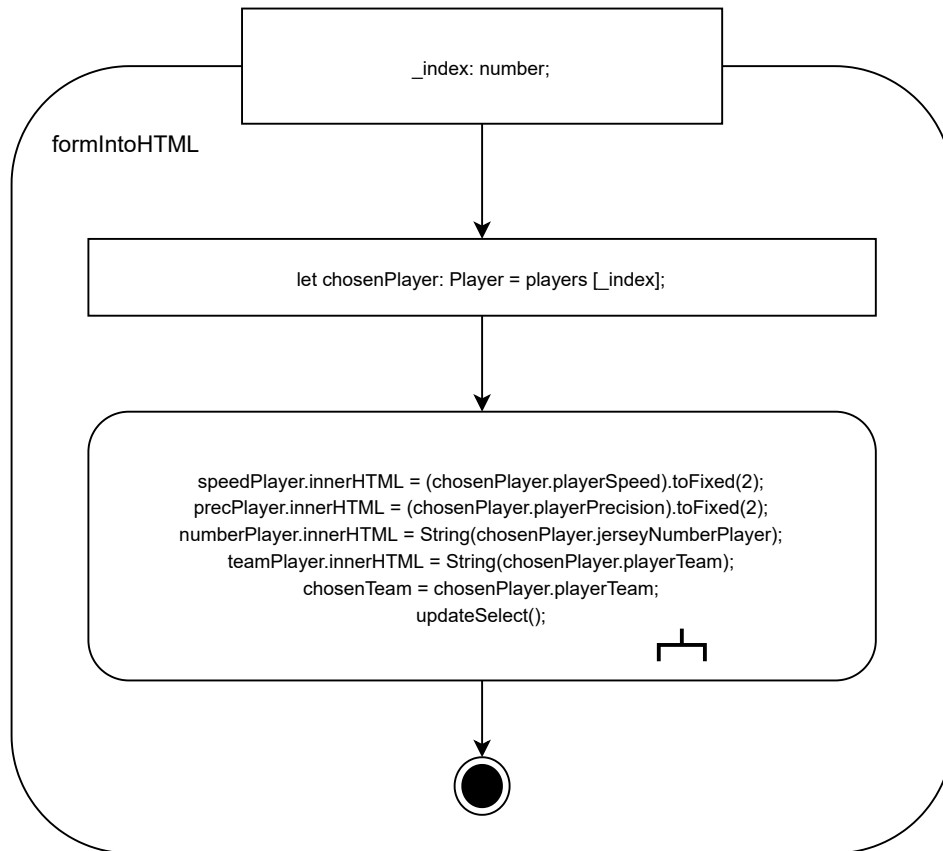


handleChange









# Activity-Diagram: vector

