Runde	
-timeline: int[]	
c Runde(sekunden: int)	

- iv1: ImageView - ivbg: ImageView - ivbum: ImageView



## **Projectile** - damage: int - speed: double - targetX: double - targetY: double - stepX: double

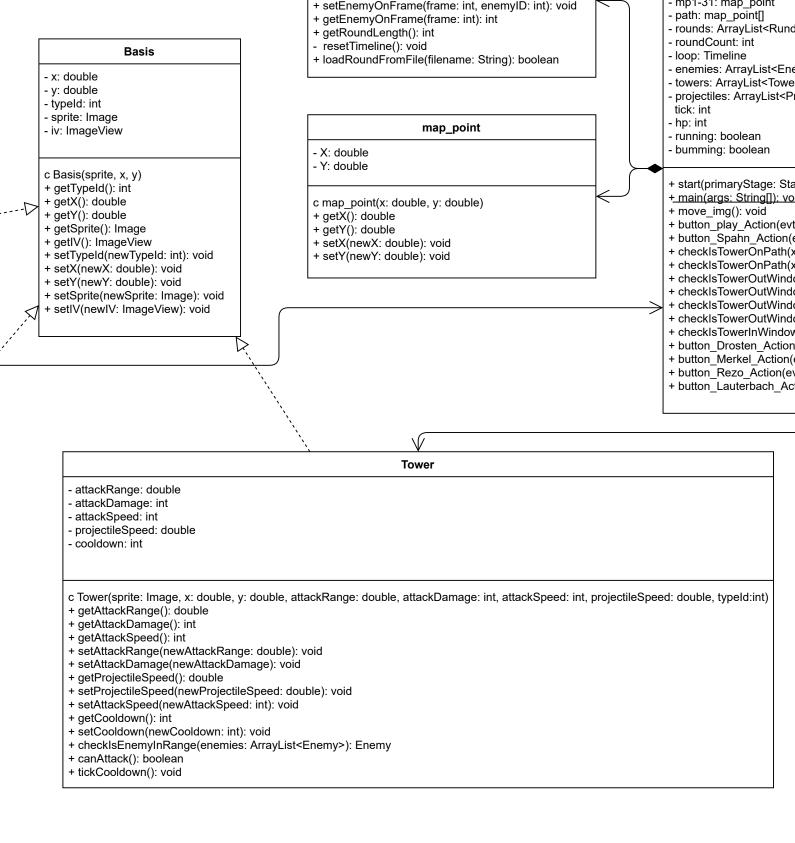
- has passedX: boolean - has passedY: boolean - deviation: int
- originX: double - originY: double

- stepY: double

- c Projectile(sprite: Image, x: double, y: double, speed: double, damage: int, targetX: double, targetY: double)
- + getSpeed(): double
- + setSpeed(newSpeed: double): void
- + getDamage(): int
- + setDamage(newDamage: int): void
- + getTargetX(): double
- + setTargetX(newTargetX: double): void
- + getTargetY(): double
- + setTargetY(newTargetY: double): void
- + calcSteps(): void
- + move(): void
- + hasPassedTarget(): boolean
- + checkIsEnemyOnProjectile(enemies: ArrayList<Enemy>): Enemy
- + toString(): String

## **Enemy**

- hp: int
- damage: int
- moneyValue: int
- speed: double
- target: int
- c Enemy(sprite: Image, x: double, y: double, speed: double, hp: int, damage: int, moneyValue: int)
- + getHp(): int
- + getDamage(): int
- + getMoneyValue(): int
- + getSpeed(): double
- + getTarget(): int
- + setHp(newHp: int): void
- + setDamage(newDamage: int): void
- + setMoneyValue(newMoneyValue: int): void
- + setSpeed(newSpeed: double): void
- + setTarget(newTarget: int): void
- + move(path: map\_point[]): void



de>
emy>
r>
rojectile>
age): void

: Event): void

:: Event): void evt: Event): void :: double, y: doul

c: double, y: double): boolean
c: double, y: double): boolean
cwUp(x: double, y: double): boolean
cwDown(x: double, y: double): boolean
cwRight(x: double, y: double): boolean
cwLeft(x: double, y: double): boolean
cw(x: double, y: double): boolean
cw(x: double, y: double): boolean
cwt: Event): void

(evt: Event): void evt: Event): void vt: Event): void tion(evt: Event): void