# Escape of the Arachnids UML

**Cole Robinson | June 8, 2023**

## Player

**<<PlayerController>>**

+ Instance: PlayerController {get; private set;}
+ joystick: VariableJoystick
- m_health: int
- m_maxHealth: int
- m_speed: float
- m_normalSpeed: float
- m_attackRange: float
- m_attackDamage: int
- m_animator: Animator
- m_rb: Rigidbody2D
- horizontal: float
- vertical: float

+ Health: int {get; set;}
+ MaxHealth: int {get;}
+ Speed: float {get; set;}
+ NormalSpeed: float {get; set;}
+ Die(): void
+ TakeDamage(damage: int): void
- Awake(): void
- FixedUpdate(): void

**<<Shooting>>**

- bulletPrefab: GameObject
- bulletSpeed: float
- touchMgr: TouchManager | -
- shootingDelay: float
- spawnDistance: float
- m_isShooting: bool
- m_animator: Animator

+ Awake(): void
+ Update(): void
-ShootWithDelay(tapPosition: Vector2): IEnumerator

## Enemy

**<<Enemy>>**

- elapsedTime: float
- m_updatePathInterval: float
# m_health: int
# m_maxHealth: int
# m_movementSpeed: float
# m_normalSpeed: float
# m_damage: int
# m_agent: NavMeshAgent
# player: Transform

+Speed: float {get, set}
+NormalSpeed: float {get, set}
+TakeDamage(int): void
+ Die(): void
- UpdatePath(): void
# Start(): void
# Update(): void
# IsVisibleInCamera(Camera): bool

**<<FastSpider>>**

# Start(): void (overrides)
# Update(): void (overrides)
- OnCollisionEnter2D(Collision2D): void

**<<ShootingSpider>>**

+ bulletPrefab: GameObject
- m_isShooting: bool
- m_shootDelay: float
- m_bulletForce: float
- m_firePointPadding: float

# Start(): void (overrides)
# Update(): void (overrides)
- ShootBullet(): IEnumerator

**<<SpiderBoss>>**

+ webPrefab: GameObject
+ eggPrefab: GameObject
+ fastSpiderPrefab: GameObject
+ m_healthBar: Slider
- m_animator: Animator
- m_bulletForce: float
- m_firePointPadding: float
- m_shootDelay: float
- m_isLaying: bool
- m_isShooting: bool

+ Die(): void (overrides)
+ TakeDamage(int): void (overrides)
# Start(): void (overrides)
# Update(): void (overrides)
- LayEgg(): IEnumerator
-ShootWeb(): IEnumerator

**<<WebBullet>>**

- m_bulletDamage: int
- characterSlowdown: float
- enemySpeedup: float

- OnTriggerStay2D(Collider2D): void
- OnTriggerExit2D(Collider2D): void

**<<EggSpawner>>**

+ spiderPrefab: GameObject
+ numSpidersPerSpawn: int
+ spawnCooldown: float
- isSpawning: bool
- m_health: int

+ TakeDamage(int): void
+Die(): void
- Update(): void
+ IsVisibleInCamera(Camera): bool
- SpawnSpiders(): IEnumerator

## Controllers

**<<DoorLevelManger>>**

+ lvl1: Transform
+ lvl2: Transform
+ lvl3: Transform
+ lvl4: Transform
+ fader: ScreenFader
- levels: Transform[]
- currentLevel: int

+ Start(): void
+ GoForward(): void
+ BackToStart(): void

**<<TouchManager>>**

+ cameraCanvas: Canvas
+ joystickRect: RectTransform
- tapLocation: Vector2
- isTapping: bool
- joystick_id: int

+ TapLocation: Vector2 {get;}
+ IsTapping: bool {get;}
+ Update(): void

**<<EventSystemManager>>**

+ Awake(): void

**<<CameraController>>**

- offset: Vector3
- smoothSpeed: float
- velocity: Vector3
+ playerTransform: Transform

+ Update(): void

**<<SoundManager>>**

- volumeSlider: Slider

+ Start(): void
+ ChangeVolume(): void
- Load(): void
- Save(): void

## UI

**<<UIManager>>**

+ Instance: UIManager {get; private set;}
- startPanel: GameObject
- pausePanel: GameObject
- optionsPanel: GameObject
- endGamePanel: GameObject
- winGamePanel: GameObject
+ isPlaying: bool

+ Awake(): void
+ Start(): void
+ ShowStartPanel(): void
+ ShowPausePanel(): void
+ HidePausePanel(): void
+ HideOptionsPanel(): void
+ ShowOptionsPanel(): void
+ ShowEndGamePanel(): void
+ ShowWinGamePanel(): void
+ DeactivateAllPanels(): void

**<<PauseMenu>>**

- pausePanel: GameObject
- optionsPanel: GameObject

+ Options(): void
+ Pause(): void
+ Unpause(): void
+ BackFromOptions(): void
+ ReturnToMenu(): void

**<<LostMenu>>**

+ ReturnToMenu(): void

**<<OptionsMenu>>**

+ Back(): void

**<<WonMenu>>**

+ ReturnToMenu(): void

**<<StartMenu>>**

- optionsPanel: GameObject
- startPanel: GameObject

+ Start(): void
+ StartGame(): void
+ PlayGame(): void
+ OpenOptions(): void
+ StartOptionsBack(): void

**<<HealthBar>>**

+ Instance: HealthBar {get; private set;}
+ heartPrefab: GameObject
+ fullHeart: Sprite
+ emptyHeart: Sprite
- m_regenerationTime: float
- m_isRegenerating: bool
- xPadding: float
- hearts: List<Image>

+ Awake(): void
+ Update(): void
+ InitializeHealth(int): void
- RegenerateHealth(): IEnumerator
+ UpdateHealth(int): void

**<<ScreenFader>>**

- fadeDuration: float
- initialAlpha: float
- fadeImage: Image

+ Awake(): void
+ StartFade(): void
- FadeImage(): IEnumerator