### SPACE INVADER 코드 구조

# Core + WIDTH: int + HEIGHT: int + FPS: int + MAX\_LIVES: int + MAX\_LIVES: int + EXTRA\_LIFE\_FRECUENCY: int + NUM\_LEVELS: int + SETTINGS\_LEVEL\_x: GameSettings -> x 의 범위는 1~7 로 각각 난이도가 설정되어있다. + frame: Frame + currentScreen: Screen + gameSettings: List<GameSettings> + LOGGER: java.util.logging.Logger + fileHandler: java.util.logging.Handler + consoleHandler: java.util.logging.ConsoleHandler + main(String[]): void + Core(): Constructor + getLogger(): java.util.logging.Logger + getDrawManager(): DrawManager + getInputManager(): InputManager + getInputManager(): FileManager + getFileManager(): FileManager + getCooldown(int): Cooldown + getVariableCooldown(int, int): Cooldown

# GameState + level : int + score : int + livesRemaining : int + bulletsShot : int + shipsDestroyed : int + GameState(int, int, int, int) : Constructor + getLevel() : int + getScore() : int + getLivesRemaining() : int + getBulletsShots() : int + getShipsDestroyed() : int

	GameSettings
+ formationWidth : int	
+ formationHeight : int	
+ baseSpeed : int	
+ shootingFrecuency : int	
+ GameSettings(int, int, int, int) + getFormationWidth(): int + getFormationHeight(): int	): Constructor

	Frame	
+ width : int		
+ height : int		
+ currentScreen : screen.Screen		
· Farman (lat. lat) · One attached		
+ Frame(int, int) : Constructor + setScreen(screen.Screen) : int + getWidth() : int		

	Score	
+ name : String		
+ score : int		
. SecretString int) - Constructor		
+ Score(String, Int) : Constructor		
+ getName() : String + getScore() : int		

## MinimalFormatter + FORMAT : java.text.DateFormat + LINE\_SEPARATOR : String - format(java.util.logglng.LogRecord) : String

#### DrawManager + instance : DrawManager + frame : Frame + fileManager : FileManager + logger : Logger + graphics : java.awt.Graphics + backBufferGraphics : java.awt.Graphics + backBuffer : java.awt.image.BufferedImage + fontRegular : java.awt.Font + fontRegularMetrics : java.awt.FontMetrics + fontBig : java.awt.Font + fontBigMetrics : java.awt.FontMetrics + spriteMap : Map<SpriteType, boolean[][]> + SpriteType(Ship, ShipDestroyed, Bullet, EnemyBullet, EnemyShipA1, EnemyShipA2, EnemyShipB1, EnemyShipB2, EnemyShipC1, EnemyShipC2, EnemyShipSpecial, Explosion) : enum + DrawManager(): Constructor + getInstance() : DrawManager + setFrame(Frame) : void + initDrawing(screen.Screen) : void

```
+ DrawManager(): Constructor
+ getInstance(): DrawManager
+ setFrame(Frame): void
+ initDrawing(screen.Screen): void
+ completeDrawing(screen.Screen): void
+ drawEntity(entity.Entity, int, int): void
+ drawBorders(screen.Screen): void
+ drawGrid(screen.Screen): void
+ drawGrid(screen.Screen, int): void
+ drawLives(screen.Screen, int): void
+ drawLives(screen.Screen, int): void
+ drawHorizontalLine(screen.Screen, int): void
+ drawMenu(screen.Screen): void
+ drawMenu(screen.Screen, int, int, float, boolean): void
+ drawResults(screen.Screen, int, int, int, float, boolean): void
+ drawRameInput(screen.Screen, char[], int): void
+ drawHighScoreMenu(screen.Screen, boolean, boolean): void
+ drawHighScores(screen.Screen, List<Score>): void
+ drawCenteredRegularString(screen.Screen, String, int): void
+ drawCenteredBigString(screen.Screen, String, int): void
+ drawCountDown(screen.Screen, int, int, boolean): void
```

## + instance : FileManager + Logger : java.util.logging.Logger + MAX\_SCORES : Int + FileManager() : Constructor + getInstance() : FileManager + loadSprite(Map<DrawManager.SpriteType, boolean[][]>) : void + loadFont(float) : java.awt.Font + loadDefaultHighScores() : List<Score> + loadHighScores() : List<Score> + saveHighScores() is void

InputManager	
+ NUM_KEYS : int + keys : boolean[]	
+ instance : InputManager	
+ InputManager() : Constructor	
+ getInstance() : InputManager	
+ isKeyDown(int) : boolean	
+ keyPressed(java.awt.event.KeyEvent) : void	
+ ketReleased(java.awt.event.KeyEvent) : void - keyTyped(java.awt.event.KeyEvent) : void	

Cooldown		
+ milliseconds : int		
+ variance : int		
+ duration : int		
+ time : long		
+ Cooldown(int) : Constructor		
+ Cooldown(int, int) : Constructor		
+ Cooldown(int) : Constructor + Cooldown(int, int) : Constructor + checkFinished() : boolean + reset() : void		

#### + positionX : int + PositionY : int + SHOTING\_INTERVAL : int + BULLET\_SPEED : int + SPEED : 2 + shootingCooldown : engine.Cooldown + width : int + height : int + color : java.awt.Color +destructionCooldown : engine.Cooldown + spriteType : engine.DrawManager.SpriteType + Ship(int, int) : Constructor + moveRight() : void + moveLeft() : void + Entity(int, int, int, Color) : Constructor + getColor() : java.awt.Color + shoot(set<Bullet> bullets) : boolean -> shootingCooldown 을 확인한 후 조건이 성립하면, Bullet pool 에서 Bullet object 를 가져온다. + getPositionX(): int + getPositionY(): int + setPositionX(int): int + setPositionY(int): int + update() : void -> Ship object 의 상태에 따라 SpriteType 을 변경한다. -> destructionCooldown 을 engine.Cooldown.reset() 하여 파괴된 상태로 설정한다. + isDestroyed() : boolean + getSpeed() : int + getSpriteType(): engine.DrawManager.SpriteType + getwidth(): int + getHeight(): int EnermyShip + A\_TYPE\_POINTS : int + B\_TYPE\_POINTS : int + C\_TYPE\_POINTS : int + speed : int + Bullet(int, int, int) : Constructor + BONUS\_TYPE\_POINTS : int + setSprite(): void + animationCooldown : engine.Cooldown + update(): void + isDestroyed : boolean + setSpeed(int) : void + getSpeed() : int + pointValue : int + EnemyShip(int, int, engine.DrawManager.SpriteType) : Constructor + EnemyShip() : Constructor (BONUS\_TYPE) + getPointValue() : int + move(int, int) : void + update() : void + update() : void -> EnemyShip object 의 종류에 따라 SpriteType 을 변경한다. BulletPool + destroy() : void + pool : Set<Bullet> + isDestroyed() : boolean + BulletPool() : Constructor + getBullet(int, int, int) : Bullet + recycle(set<Bullet>) : void EnermyShipFormation + INIT\_POS\_X : int + INIT\_POS\_Y : int + INIT\_POS\_Y : int + SEPARATION\_DISTANCE : int + PROPORTION\_C : double + PROPORTION\_B : double + Y SPEED : int + X\_SPEED : Int + Y\_SPEED : int + BULLER\_SPEED : int + SHOOTING\_VARIANCE : double + SIDE\_MARGIN : int + BOTTOM\_MARGIN : int + DESCENT\_DISTANCE : int + MINIMUM\_SPEED : int + drawManager : engine.DrawManager + logger : java.util.logging.Logger + screen : screen.Screen + enemyShips : List<List<EnemyShip>> + shootingCooldown : engine.Cooldown + nShipsWide : int + nShipsHigh : int + shootingVariance : int + baseSpeed : int + movementSpeed : int + currentDirection : Direction + previousDirection : Direction + movementInterval : int + width : int + height : int + positionY : int + shipWidth : int + shipHeight : int + shooters : List<EnemyShip> + shipCount : int + Direction{RIGHT, LEFT, DOWN} : enum + EnemyShipFormation(engine.GameSettings) : Constructor + attach(screen.Screen) : void + draw(): void + update() : void + cleanUp() : void + shoot(Set<Bullet>) : void + destroy(EnemyShip) : void + getNextShooter(List<EnemyShip>) : EnemyShip iterator() : Iterator<EnemyShip> + isEmpty() : boolean

