Plane

- cockpit: Cockpit

+ descend(): void

Cockpit

- command: ISetAllFlapsCommand
- controlUnit: IControlUnit

+ descend(): void

«interface»
IControlUnit

+ addPlanePositionDataListener(listener: IPlanPositionDataListener): void
+ moveAllFlaps(degree: double): void
+ registerLeftWing(wing: Wing): void
+ registerRightWing(wing: Wing): void
+ removePlanePositionDataListener(listener: IPlanPositionDataListener): void

«interface»
IMoveAllWingFlapsCommand

+ moveAllWingFlaps(degree: double): void

«interface»
ISetAllFlapsCommand

+ setAllFlaps(): void

ControlUnit

- isDescending: boolean
- leftWing: Wing
- planeDistance: double
- planeHeight: double
- planeId: String
- planeSpeed: double
- positionDataListeners: List<IPlanePositionDataListener>
- rightWing: Wing
- timer: Timer

+ addPlanePositionDataListener(listener: IPlanePositionDataListener): void
- calculatePositionData(): void
+ moveAllFlaps(degree: double): void
+ outputInfoToConsole(): void
+ registerLeftWing(wing: Wing): void
+ registerRightWing(wing: Wing): void
+ removePlanePositionDataListener(listener: IPlanePositionDataListener): void
+ updatePositionData(): void

Wing

- controlUnit: IControlUnit
- leftFlap: Flap
- rightFlap: Flap

+ moveAllWingFlaps(degree: double): void
+ moveFlaps(degree: double): void
+ moveLeftFlap(degree: double): void
+ moveRightFlap(degree: double): void

SetAllFlapsToThreeDegreeCommand

- controlUnit: ControlUnit

+ setAllFlaps(): void

Flap

- currentAngle: double

+ moveToDegree(degree: double): void

«interface»
IPlanePositionDataListener

+ positionDataUpdate(speed: double, height: double, distance: double, id: String): void

«interface»
IDescendCommand

+ execute(): void

Tower

- command: IDescendCommand
- currentPlane: Plane
- descendCommandIsSent: boolean

+ isDescendPointReached(height: double, distance: double): boolean
+ positionDataUpdate(speed: double, height: double, distance: double, id: String): void
+ sendDescendCommand(): void

DescendCommand

- plane: Plane

+ execute(): void