

構想與規格書 生成過程

Vehicle (車輛) – 抽象類別

屬性：

licensePlate (字串)：車牌號碼，不可為空。

type (列舉)：車輛類型 (MOTORCYCLE, SEDAN, SUV, TRUCK)。

行為：

getHourlyRate()：抽象方法，回傳該車種每小時費率。

子類別實作：

Motorcycle : 50元/小時 (小型車位)，身障25元/小時。

Sedan : 100元/小時 (標準車位)，身障50元/小時。

SUV : 200元/小時 (大型車位)，身障100元/小時。

Truck : 300元/小時 (需佔用兩個連續標準車位)，身障150元/小時。



若時長 > 12 小時且策略為DailyMax(Penalty)，拋出 IllegalStateException。

ParkingSpot (車位)

屬性：

spotId (字串)：車位編號，不可為空或null。

isOccupied (布林)：目前是否被佔用，預設為false。

車位配對規則：

MotorcycleSpot: 僅限 Motorcycle。若機車停入其他位，拋出例外。

RegularSpot: 一般車位, Sedan SUV可停放。

LargeSpot: 僅限 SUV 停放。

TruckSpot: 僅限 Truck 停放 (本系統現改為 Truck 需尋找兩個連續 RegularSpot)。

occupyTwoSpots(List<ParkingSpot> spots): 進場時將兩個連續車位設為 isOccupied = true

vacateTwoSpots(List<ParkingSpot> spots): 離場時必須同時將該對應的兩個車位設為 isOccupied = false

ParkingLot.releaseTruckSpots(String plate): 同時釋放貨車佔用的多格車位

HandicappedSpot: 所有車種 (需驗證身障身分)。

ReservedSpot: 預約車位。

PricingStrategy(計費策略)

calculateFee(Duration duration, Vehicle vehicle): 計算總費用。

getDescription(): 回傳策略名稱

HourlyPricing 標準計費：小時數 * 車種標準費率。



DailyMaxPricing：每日上限計費(以一個車種每小時標準費*12小時計算，不能停超過12小時)

ProgressivePricing：累進計費: 最多停12小時，前8小時每小時的錢為標準費，後4小時每小時為標準費+50元

MemberPricing 會員專屬折扣 (總金額85折)。

PricingStrategy(計費策略)

去掉累進計費，新增：

TimeBasedPricing (分時進階)：

假日：費率 * 1.5。

夜間 (23:00-07:00)：費率 * 0.8。

若停車期間跨越夜間與日間，應依比例分別計算或以出場時間判定（建議採每小時分別判定）

`ParkingLot.calculatePenalty(long overtimeHours)`

罰金 (Penalty) 計算與優先級

超時罰金：

觸發條件：實際在場停車時間超過 12 小時限制。

費率：每小時加收 100 元（不滿一小時以一小時計）。

折扣排除：罰金為獨立項目，不套用身障優惠 (50%) 與會員折扣 (85折)。

預約違約金：

觸發條件：車輛進場時間晚於預約整點。

費率：固定加收一次性違約金 200 元。

繳費規則：限現金支付，未繳清前禁止執行進場流程及會員註冊。

計算順序：應繳總計 = `Math.round(基本費 * 身障折扣 * 會員折扣) + 超時罰金 + 預約違約金。`

ParkingLot (停車場)

屬性：

name: 停車場名稱。

floors (ArrayList<ParkingFloor>): 樓層清單。

maxCapacity (整數): 停車場總格數。

currentVehicleCount (整數): 目前場內車輛總數。

allSpots (Map<String, ParkingSpot>): 使用 Map 以 spotId 快速檢索所有單位，不可為 null，初始為空 Map。

行為：

processEntry(Vehicle v): 車輛進場，自動分配車位並產生 ParkingTicket。

processExit(ParkingTicket ticket): 車輛出場，根據策略計算費用並釋放車位。

getStatistics(): 回傳當前停車場狀態 (總車位、剩餘車格、各類型車位佔用率)。

isFull(): 判斷 currentVehicleCount >= maxCapacity。

後續增加

checkReEntryStatus(String licensePlate):

搜尋歷史紀錄中該車牌的最後一筆出場紀錄。

若無紀錄或間隔 ≥ 2 小時：回傳 NEW_SESSION。

若間隔小於 2 小時：回傳 CONTINUOUS_SESSION。

HandicappedSpot (身障車位)

行為： verifyAccess(Member m)

邏輯： 檢查 m.isHandicapped 是否為 true 且 m.disabilityCardId 是否有效。

失敗拋出： SecurityException (訊息："無效的身障識別證明，不可使用此車位")。

findAvailableSpot(Vehicle vehicle):

Truck 邏輯： 搜尋同一樓層、編號連續且皆為空的兩個 RegularSpot

依車種搜尋： 機車 ->MotorcycleSpot 轎車 -> RegularSpot 或身障位。

Truck 邏輯： 需在同一排找到連續兩個空的 RegularSpot。

優先級： 低樓層優先。 無位子時回傳 null。

verifyAccess(Member m):

檢查身障手冊編號。失敗拋出 SecurityException。

再進場檢查 (checkReEntryStatus):

搜尋歷史紀錄。若離場與再次進場間隔 小於 2 小時，回傳 CONTINUOUS_SESSION 並繼承前次 totalDurationUsed

autoReleaseExpiredReservation()

執行時機：每次有新預約或車輛進場輸入時間時觸發。

邏輯：若 now > reservedTime，立即執行 targetSpot.setOccupied(false) 並將預約紀錄移至違約待繳清單

ParkingFloor (樓層)

屬性：

name: 停車場名稱。

floors (ArrayList<ParkingFloor>): 樓層清單。

maxCapacity (整數): 停車場總格數。

currentVehicleCount (整數): 目前場內車輛總數。

allSpots (Map<String, ParkingSpot>): 使用 Map 以 spotId 快速檢索所有車位，不可為 null，初始為空 Map。

行為：

processEntry(Vehicle v): 車輛進場，自動分配車位並產生 ParkingTicket。

processExit(ParkingTicket ticket): 車輛出場，根據策略計算費用並釋放車位。

getStatistics(): 回傳當前停車場狀態（總車位、剩餘車格、各類型車位佔用率）。

isFull(): 判斷 currentVehicleCount >= maxCapacity。

Reservable (可預約)

reserve(LocalDateTime start, Duration duration): 預約。

cancelReservation(String reservationId): 取消預約。

isAvailableAt(LocalDateTime time): 檢查特定時間是否可用。

行為：

autoReleaseExpiredReservation()

執行時機：每次有新預約或車輛進場輸入時間時觸發。

邏輯：若 now > reservedTime，立即執行 targetSpot.setOccupied(false) 並將預約紀錄移至違約待繳清單

ParkingTicket (停車票券)

ticketId (字串): 唯一識別碼。

entryTime (LocalDateTime): 進場時間。

vehicle (Vehicle): 對應車輛。

assignedSpot (ParkingSpot): 分配的車位。



boolean isPaid, 初始值為 false

行為: * pay(ParkingTicket ticket): 執行繳費, 成功後將 isPaid 設為 true。

processExit(ParkingTicket ticket):

檢查機制: 若 ticket.isPaid 為 false, 拋出IllegalStateException。

訊息: "此票券尚未完成繳費, 無法離場"。

票券紀錄: ParkingTicket 須關聯所有佔用車位編號, 確保離場釋放資源的一致性

Member (會員)

屬性：

memberId, name: 根本資訊。

balance (double): 儲值金餘額。

hasMonthlyPass (布林): 是否持有月票。

disabilityCardId (字串): 身心障礙手冊編號。

isHandicapped (布林值): 是否具備身障資格

行為：

deductBalance(double amount):

扣款，餘額不足時拋出 IllegalStateException。

訊息："停車時間超過 12 小時上限"

若 balance 小於 amount，拋出 IllegalStateException，訊息：「會員帳戶餘額不足，請先加值」

後續增加

屬性:

expiryDate (LocalDateTime): 月票到期日。

行為:

checkMonthlyPass(): 若未過期，本次停車費自動結算為 0 元。

若當前時間未超過 expiryDate，則本次停車基本費為 0 元（超時罰金另計）

月票生效判定:

isMonthlyPassValid(): 檢查 LocalDateTime.now() 是否在 expiryDate 之前。

折抵範圍: 月票僅折抵「基本停車費」，「超時罰金」與「違約金」需照價扣款

Member.getRoundedBalance(): 回傳 Math.round(balance) 處理後的整數餘額

ParkingSession (停車紀錄)

屬性：

包含 ticketId(不可為空或null),

exitTime, totalFee, paymentStatus， 不可為null。

用途： 用於事後報表查詢與歷史紀錄。

licensePlate (車牌號碼)

vehicleType (車種)：

totalDurationUsed (Duration)： 紀錄該車輛在當前有效週期內已使用的時數。

entryTime: 進場時間

lastExitTime (LocalDateTime)： 最後一次駛離的時間。

例外處理規則

IllegalArgumentException:

車牌或會員 ID 為空時。"不可為空或 null"

查無此票券 ID 時。

非電動車進入充電位 "此車位僅供電動車使用"

票券無效或查無紀錄 "無效的停車票券或車牌號碼"

身障驗證失敗 "無效的身障識別證明，不可使用此車位"

車種與車位類型不符 "此車種不符合該車位類型限制，請停往專屬區域"

時間格式解析失敗：當使用者輸入的時間字串不符合 yyyy.MM.dd.HH.mm 格式時: "時間格式錯誤，請使用範例格式: 2025.12.25.14.30"。

非數值輸入：在「選擇車種」、「加值金額」或「支付金額」欄位輸入了非數字的字元時: "輸入格式錯誤，請輸入有效的數字金額"。

身障編號格式不符: "身障編號格式錯誤，需為 a 或 b 開頭並加上 5 個數字"。

重複註冊：使用已經存在的會員 ID 進行新會員註冊時: "此會員 ID 已被註冊，請使用其他 ID"。

例外處理規則

IllegalStateException:

超過 12 小時限制 "停車時間超過 12 小時上限"

停車場已滿 "停車場已滿，目前無可用車位"

找不到對應車種的車位 "找不到適合此類別車輛的車位"

貨車找不到連續車位 "找不到供貨車停放的連續車位"

會員餘額不足 "會員餘額不足"

兩小時內重複進場且累計超過12小時 "2小時內重複進場，累停時間已達 12 小時上限"

車輛已在場內：同一車牌在尚未離場的情況下，嘗試再次執行進場: "此車輛已在場內，不可重複進場"。

離場時間早於進場時間; "離場時間不可早於進場時間"。

支付中斷強制離場：會員餘額不足且拒絕加值，卻嘗試完成出場程序時: "支付程序未完成，禁止開啟柵門"。

月票購買資格不符：非會員嘗試購買月票，或餘額不足且拒絕加值時: "月票購買程序失敗，請檢查會員狀態或餘額"

負數停車時數：若因手動輸入錯誤導致 Duration 計算出負數: "計算錯誤：檢測到異常的停車期間"