Merchandise `<<interface>>`

Attributes

Operations
+ getIsTradeable(): boolean
+ getOwner(): String
+ getNameOrTitle(): String
+ getId(): UUID
+ getTypeString(): String

PublicWalletRegistry

Attributes
+ wallets: List<Wallet>

Operations
+ PublicWalletRegistry():
+ showPublicWalletIDs(): List<UUID>
+ makeWalletPublic(Wallet): void
+ makeWalletPrivate(Wallet): void
+ getWallets(): List<Wallet>
+ isWalletPublic(Wallet): boolean

Market

Attributes
+ userRepository: UserRepository
+ itemsForSale: List<Merchandise>
+ listings: HashMap<UUID, String>
+ walletLibrary: WalletManager
+ artManager: ArtManager

Operations
+ Market(UserRepository, ArtManager, WalletManager):
+ makeTradeWithArt(UUID, UUID): void
+ getNamesMerchandiseForSale(): List<String>
+ removeItem(Merchandise): void
+ checkItem(Merchandise): boolean
+ addArtToMarket(UUID): boolean
+ getItemsForSale(): List<Merchandise>
+ makeTradeWithCash(UUID, UUID): void

WalletManager

Attributes
+ registry: PublicWalletRegistry
+ wallets: Map<UUID, Wallet>
+ userFinder: FindUser
+ DEFAULT_INIT_CURRENCY: double

Operations
+ WalletManager(UserRepository):
+ createWallet(User, String, boolean, UUID, double): Wallet
+ createWallet(User): Wallet
+ getUserWalletByID(String, UUID): Wallet
+ makeWalletPublic(Wallet): void
+ getWalletById(UUID): Wallet
+ getWalletByID(UUID): Wallet
+ changeOwner(Wallet, String): void
+ retrievePublicWallets(): List<UUID>
+ createWallet(User, String, boolean): Wallet
+ getUserWallets(String): List<Wallet>
+ getUserWalletIDs(String): List<UUID>
+ transferWallet(String, String, UUID): void
+ makeWalletPrivate(Wallet): void

Wallet

Attributes
+ publicAccess: boolean
+ netWorth: double
+ id: UUID
+ currency: double
+ walletName: String
+ arts: HashMap<UUID, Art>
+ owner: User

Operations
+ Wallet(User, String):
+ Wallet(User, String, UUID, double):
+ addArt(Art): void
+ getNetWorth(): double
+ setWalletName(String): void
+ isPublicAccess(): boolean
+ getId(): UUID
+ getName(): String
+ getTypeString(): String
+ getOwnerObj(): User
+ getNameOrTitle(): String
+ setOwner(User): void
+ getIsTradeable(): boolean
+ getIsEmpty(): boolean
+ calcNetWorth(): void
+ getCurrency(): double
+ getArts(): HashMap<UUID, Art>
+ containsArt(UUID): boolean
+ getArt(UUID): Art
+ removeCurrency(double): void
+ getOwner(): String
+ addCurrency(double): void
+ setPublic(boolean): void
+ removeArt(Art): void
+ getAllArt(): HashMap<UUID, Art>

WalletFacade

Attributes
+ artManager: ArtManager
+ walletManager: WalletManager
+ wallet: Wallet

Operations
+ WalletFacade(Wallet, WalletManager, ArtManager):
+ getCurrency(): double
+ getIsTradeable(): boolean
+ getId(): UUID
+ initializeWalletByID(String, UUID): void
+ getWalletArtTitles(): Map<UUID, String>
+ getNetWorth(): double
+ getWalletArtPrices(): Map<UUID, Float>
+ getName(): String
+ getTradeableArtNames(): HashMap<UUID, String>
+ getAllWalletArt(): HashMap<UUID, Art>
+ addArtToWallet(Art): void
+ getWalletArts(): Map<UUID, String>

TradingUtil

Attributes
+ tradingTo: Wallet
+ tradingFrom: Wallet
+ userRepository: UserRepository

Operations
+ TradingUtil(Wallet, Wallet, UserRepository):
+ makeTrade_Art_Money(Art): boolean
+ makeTrade_Wallet_Wallet(): boolean
+ makeTrade_Art_Art(Art, Art): boolean

Art

Attributes
+ isTradable: boolean
+ asciiString: String
+ wallet: Wallet
+ id: UUID
+ price: float
+ title: String

Operations
+ Art(String, String):
+ Art(String, String, UUID, float):
+ getId(): UUID
+ getIsTradeable(): boolean
+ setWallet(Wallet): void
+ getNameOrTitle(): String
+ getTitle(): String
+ getWallet(): Wallet
+ getArt(): String
+ getOwner(): String
+ getTypeString(): String
+ getPrice(): float
+ setisTradable(boolean): void
+ setPrice(float): void

ArtGenerator

Attributes
+ asciiCategories: Map<String, String>

Operations
+ ArtGenerator():
+ generateArt(String): String
+ getRandomArt(String): String
+ determineClosestCategory(String): String

Appraiser

Attributes

Operations
+ Appraiser():
+ appraiseArt(ArtManager, Art): double

ArtManager

Attributes
+ library: HashMap<UUID, Art>
+ walletManager: WalletManager

Operations
+ ArtManager(WalletManager):
+ getArt(UUID): Art
+ createNewArt(String, String, float, UUID): void
+ artExists(Art): boolean
+ addArt(Art, UUID): void
+ getAllArt(): Collection<Art>
+ getArtByWallet(UUID): Map<UUID, ArtFacade>

ArtFacade

Attributes
+ artManager: ArtManager
+ art: Art

Operations
+ ArtFacade(Art, ArtManager):
+ getAsciiArt(): String
+ getTradeable(): boolean
+ getPrice(): double
+ getId(): UUID
+ getTitle(): String