user

+

+ transitdroid.domain.entities.user

---

user::UserFactory

+ createClean(UUID, int, List<UserRole>, String, String, String, byte[], byte[]) : User
+ createNew(List<UserRole>, String, String, String, byte[], byte[]) : User
+ createNew(UUID, int, List<UserRole>, String, String, String, byte[], byte[]) : User

---

BaseOutputMapper
user::UserOutputMapper

+ insert(User) : boolean
+ remove(User) : boolean
+ update(User) : boolean

«interface»
user::IUser

+ getFirstName() : String
+ getLastName() : String
+ getPassword() : byte[]
+ getRoles() : List<UserRole>
+ getSalt() : byte[]
+ getUsername() : String
+ setFirstName(String) : void
+ setLastName(String) : void
+ setPassword(byte[]) : void
+ setRoles(List<UserRole>) : void
+ setSalt(byte[]) : void
+ setUsername(String) : void

DomainObject
user::User

- firstName: String
- lastName: String
- password: byte ([])
- roles: List<UserRole>
- salt: byte ([])
- username: String

+ getFirstName() : String
+ getLastName() : String
+ getPassword() : byte[]
+ getRoles() : List<UserRole>
+ getSalt() : byte[]
+ getUsername() : String
+ setFirstName(String) : void
+ setLastName(String) : void
+ setPassword(byte[]) : void
+ setRoles(List<UserRole>) : void
+ setSalt(byte[]) : void
+ setUsername(String) : void
~ User(UUID, int, List<UserRole>, String, String, String, byte[], byte[])
~ User()

BaseInputMapper
user::UserInputMapper

+ find(UUID) : User
+ find(String, String) : User
# map(ResultSet) : User

«enumeration»
user::UserRole

ADMINISTRATOR
PASSENGER

Attributes
- index: int {readOnly}

~ UserRole(int)
+ index() : int

«enumeration»
user::UserType

REGULAR
SENIOR
STUDENT
AID

Attributes
- index: int {readOnly}

~ UserType(int)
+ index() : int

BaseProxy
user::UserProxy

+ equals(Object) : boolean
+ getFirstName() : String
# getFromMapper(UUID) : User
+ getId() : UUID
+ getLastName() : String
+ getPassword() : byte[]
+ getRoles() : List<UserRole>
+ getSalt() : byte[]
+ getUsername() : String
+ hashCode() : int
+ setFirstName(String) : void
+ setLastName(String) : void
+ setPassword(byte[]) : void
+ setRoles(List<UserRole>) : void
+ setSalt(byte[]) : void
+ setUsername(String) : void
+ toString() : String
# UserProxy(UUID)

---

user::UserIdentityMap

- instance: UserIdentityMap = null
- map: IdentityHashMap<UUID, User> = new IdentityHas...

~ get(UUID) : User
~ getUniqueInstance() : UserIdentityMap
~ put(UUID, User) : void
- UserIdentityMap()

-instance