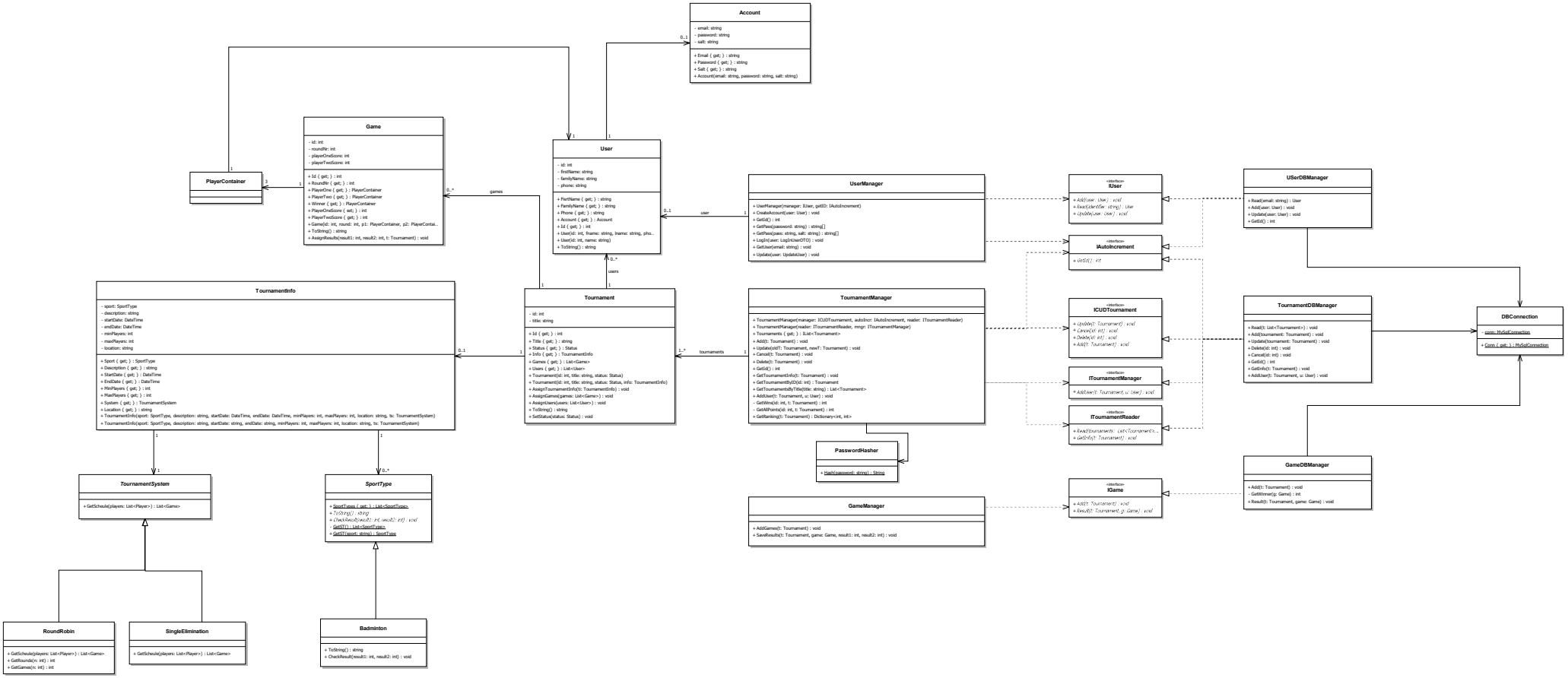
Account
- email: string
- password: string
- salt: string
+ Email { get; } : string
+ Password { get; } : string
+ Salt { get; } : string
+ Account(email: string, password: string, salt: string)
0..1
1
1
Game
- id: int
- roundNr: int
- playerOneScore: int
- playerTwoScore: int
+ Id { get; } : int
+ RoundNr { get; } : int
+ PlayerOne { get; } : PlayerContainer
+ PlayerTwo { get; } : PlayerContainer
+ Winner { get; } : PlayerContainer
+ PlayerOneScore { set; } : int
+ PlayerTwoScore { set; } : int
+ Game(id: int, round: int, p1: PlayerContainer, p2: PlayerContai...
+ ToString() : string
+ AssignResults(result1: int, result2: int, t: Tournament) : void
PlayerContainer
1
3
1
0..*
games
User
- id: int
- firstName: string
- familyName: string
- phone: string
+ FirstName { get; } : string
+ FamilyName { get; } : string
+ Phone { get; } : string
+ Account { get; } : Account
+ Id { get; } : int
+ User(id: int, fname: string, lname: string, pho...
+ User(id: int, name: string)
+ ToString() : string
0..1
user
1
0..*
users
1
1
UserManager
+ UserManager(manager: IUser, getID: IAutoIncrement)
+ CreateAccount(user: User) : void
+ GetId() : int
+ GetPass(password: string) : string[]
+ GetPass(pass: string, salt: string) : string[]
+ LogIn(user: LogInUserDTO) : void
+ GetUser(email: string) : void
+ Update(user: UpdateUser) : void
«interface»
IUser
+ Add(user: User) : void
+ Read(identifier: string) : User
+ Update(user: User) : void
«interface»
IAutoIncrement
+ GetID() : int
USerDBManager
+ Read(email: string) : User
+ Add(user: User) : void
+ Update(user: User) : void
+ GetId() : int
TournamentInfo
- sport: SportType
- description: string
- startDate: DateTime
- endDate: DateTime
- minPlayers: int
- maxPlayers: int
- location: string
+ Sport { get; } : SportType
+ Description { get; } : string
+ StartDate { get; } : DateTime
+ EndDate { get; } : DateTime
+ MinPlayers { get; } : int
+ MaxPlayers { get; } : int
+ System { get; } : TournamentSystem
+ Location { get; } : string
+ TournamentInfo(sport: SportTypo, description: string, startDate: DateTime, endDate: DateTime, minPlayers: int, maxPlayers: int, location: string, ts: TournamentSystem)
+ TournamentInfo(sport: SportTypo, description: string, startDate: string, endDate: string, minPlayers: int, maxPlayers: int, location: string, ts: TournamentSystem)
0..1
1
Tournament
- id: int
- title: string
+ Id { get; } : int
+ Title { get; } : string
+ Status { get; } : Status
+ Info { get; } : TournamentInfo
+ Games { get; } : List<Game>
+ Users { get; } : List<User>
+ Tournament(id: int, title: string, status: Status)
+ Tournament(id: int, title: string, status: Status, info: TournamentInfo)
+ AssignTournamentInfo(ti: TournamentInfo) : void
+ AssignGames(games: List<Game>) : void
+ AssignUsers(users: List<User>) : void
+ ToString() : string
+ SetStatus(status: Status) : void
1..*
tournaments
1
TournamentManager
+ TournamentManager(manager: ICUDTournament, autoIncr: IAutoIncrement, reader: ITournamentReader)
+ TournamentManager(reader: ITournamentReader, mngr: ITournamentManager)
+ Tournaments { get; } : IList<Tournament>
+ Add(t: Tournament) : void
+ Update(oldT: Tournament, newT: Tournament) : void
+ Cancel(t: Tournament) : void
+ Delete(t: Tournament) : void
+ GetId() : int
+ GetTournamentInfo(t: Tournament) : void
+ GetTournamentById(id: int) : Tournament
+ GetTournamentsByTitle(title: string) : List<Tournament>
+ AddUser(t: Tournament, u: User) : void
- GetWins(id: int, t: Tournament) : int
- GetAllPoints(id: int, t: Tournament) : int
+ GetRanking(t: Tournament) : Dictionary<int, int>
«interface»
ICUDTournament
+ Update(t: Tournament) : void
+ Cancel(id: int) : void
+ Delete(id: int) : void
+ Add(t: Tournament) : void
«interface»
ITournamentManager
+ AddUser(t: Tournament, u: User) : void
«interface»
ITournamentReader
+ ReadTournaments: List<Tournament>..
+ getInfo(t: Tournament) : void
TournamentDBManager
+ Read(t: List<Tournament>) : void
+ Add(tournament: Tournament) : void
+ Update(tournament: Tournament) : void
+ Delete(id: int) : void
+ Cancel(id: int) : void
+ GetId() : int
+ GetInfo(t: Tournament) : void
+ AddUser(t: Tournament, u: User) : void
DBConnection
- conn: MySqlConnection
+ Conn { get; } : MySqlConnection
PasswordHasher
+ Hash(password: string) : String
GameManager
+ AddGames(t: Tournament) : void
+ SaveResults(t: Tournament, game: Game, result1: int, result2: int) : void
«interface»
IGame
+ Add(t: Tournament) : void
+ Result(t: Tournament, g: Game) : void
GameDBManager
+ Add(t: Tournament) : void
- GetWinner(g: Game) : int
+ Result(t: Tournament, game: Game) : void
TournamentSystem
+ GetSchedule(players: List<Player>) : List<Game>
SportType
+ SportTypes { get; } : List<SportType>
+ ToString() : string
+ CheckResult(result1: int, result2: int) : void
- GetST() : List<SportType>
+ GetST(sport: string) : SportType
1
1
1
0..*
RoundRobin
+ GetSchedule(players: List<Player>) : List<Game>
+ GetRounds(n: int) : int
+ GetGames(n: int) : int
SingleElimination
+ GetSchedule(players: List<Player>) : List<Game>
Badminton
+ ToString() : string
+ CheckResult(result1: int, result2: int) : void