**Sports Club**

**Глория и Виктория**

Структура

Картина, която съдържа екранна снимка, компютър, лаптоп, клавиатура

Описанието е генерирано автоматичноКартина, която съдържа екранна снимка, компютър, лаптоп, клавиатура

Описанието е генерирано автоматично

Папка Business

* Class PlayerBusiness

namespace SportsClub.Business

{

public class PlayerBusiness

{

private SportsClubContext sportsClubContext;

public List<Player> GetAll()

{

using (sportsClubContext = new SportsClubContext())

{

return sportsClubContext.Players.ToList();

}

}

public Player Get(int? id)

{

using (sportsClubContext = new SportsClubContext())

{

return sportsClubContext.Players.Find(id);

}

}

public void Add(Player player)

{

using (sportsClubContext = new SportsClubContext())

{

CheckIfTeamExists(player.TeamId);

CheckIfPlayerNameIsCorrect(player);

sportsClubContext.Players.Add(player);

sportsClubContext.SaveChanges();

}

}

public void Update(Player player)

{

using (sportsClubContext = new SportsClubContext())

{

var item = sportsClubContext.Players.Find(player.Id);

if (item != null)

{

CheckIfTeamExists(player.TeamId);

CheckIfPlayerNameIsCorrect(player);

sportsClubContext.Entry(item).CurrentValues.SetValues(player);

sportsClubContext.SaveChanges();

}

}

}

public void Delete(int id)

{

using (sportsClubContext = new SportsClubContext())

{

var item = sportsClubContext.Players.Find(id);

if (item != null)

{

try

{

sportsClubContext.Players.Remove(item);

}

catch

{

sportsClubContext.Players.Remove(item);

}

sportsClubContext.SaveChanges();

}

}

}

private void CheckIfTeamExists(int? teamId)

{

bool teamIdExists = false;

foreach (var team in sportsClubContext.Teams)

{

if (team.Id == teamId)

{

teamIdExists = true;

}

}

if (!teamIdExists)

{

throw new ArgumentException("No team with such id");

}

}

private void CheckIfPlayerNameIsCorrect(Player player)

{

if (player.Name == "" || player.Name == null)

{

throw new ArgumentException("Player name can't be empty");

}

}

}

}

* Class SportBusiness

namespace SportsClub.Business

{

public class SportBusiness

{

private SportsClubContext sportsClubContext;

public List<Sport> GetAll()

{

using(sportsClubContext = new SportsClubContext())

{

return sportsClubContext.Sports.ToList();

}

}

public Sport Get(int? id)

{

using (sportsClubContext = new SportsClubContext())

{

return sportsClubContext.Sports.Find(id);

}

}

public void Add(Sport sport)

{

using (sportsClubContext = new SportsClubContext())

{

CheckIfSportExists(sport);

CheckIfSportNameIsCorrect(sport);

sportsClubContext.Sports.Add(sport);

sportsClubContext.SaveChanges();

}

}

public void Update(Sport sport)

{

using (sportsClubContext = new SportsClubContext())

{

var item = sportsClubContext.Sports.Find(sport.Id);

if (item != null)

{

CheckIfSportExists(sport);

CheckIfSportNameIsCorrect(sport);

sportsClubContext.Entry(item).CurrentValues.SetValues(sport);

sportsClubContext.SaveChanges();

}

}

}

public void Delete(int id)

{

using (sportsClubContext = new SportsClubContext())

{

var item = sportsClubContext.Sports.Find(id);

if (item != null)

{

foreach (var team in sportsClubContext.Teams)

{

if (team.SportId == item.Id)

{

TeamBusiness teamBusiness = new TeamBusiness();

teamBusiness.Delete(team.Id);

}

}

try

{

sportsClubContext.Sports.Remove(item);

}

catch

{

sportsClubContext.Sports.Remove(item);

}

sportsClubContext.SaveChanges();

}

}

}

private void CheckIfSportExists(Sport sport)

{

foreach (var existentSport in sportsClubContext.Sports)

{

if (sport.Name == existentSport.Name)

{

throw new ArgumentException("Sport already exists");

}

}

}

private void CheckIfSportNameIsCorrect(Sport sport)

{

if (sport.Name == "" || sport.Name == null)

{

throw new ArgumentException("Sport name can't be empty");

}

}

}

}

* Class TeamBusiness

namespace SportsClub.Business

{

public class TeamBusiness

{

private SportsClubContext sportsClubContext;

public List<Team> GetAll()

{

using (sportsClubContext = new SportsClubContext())

{

return sportsClubContext.Teams.ToList();

}

}

public Team Get(int? id)

{

using (sportsClubContext = new SportsClubContext())

{

return sportsClubContext.Teams.Find(id);

}

}

public void Add(Team team)

{

using (sportsClubContext = new SportsClubContext())

{

CheckIfSportExists(team.SportId);

CheckIfTrainerExists(team.TrainerId);

CheckIfTeamNameIsCorrect(team);

CheckIfTeamExists(team);

sportsClubContext.Teams.Add(team);

sportsClubContext.SaveChanges();

}

}

public void Update(Team team)

{

using (sportsClubContext = new SportsClubContext())

{

var item = sportsClubContext.Teams.Find(team.Id);

if (item != null)

{

CheckIfSportExists(team.SportId);

CheckIfTrainerExists(team.TrainerId);

CheckIfTeamNameIsCorrect(team);

CheckIfTeamExists(team);

sportsClubContext.Entry(item).CurrentValues.SetValues(team);

sportsClubContext.SaveChanges();

}

}

}

public void Delete(int id)

{

using (sportsClubContext = new SportsClubContext())

{

var item = sportsClubContext.Teams.Find(id);

if (item != null)

{

foreach (var player in sportsClubContext.Players)

{

if (player.TeamId == item.Id)

{

sportsClubContext.Players.Remove(player);

}

}

try

{

sportsClubContext.Teams.Remove(item);

}

catch

{

sportsClubContext.Teams.Remove(item);

}

sportsClubContext.SaveChanges();

}

}

}

private void CheckIfTeamExists(Team team)

{

bool teamExists = false;

foreach (var existingTeam in sportsClubContext.Teams)

{

if (existingTeam.Name == team.Name &&

existingTeam.SportId == team.SportId &&

existingTeam.TrainerId == team.TrainerId)

{

teamExists = true;

}

}

if (teamExists)

{

throw new ArgumentException("Team already exists");

}

}

private void CheckIfSportExists(int? sportId)

{

bool sportIdExists = false;

foreach (var sport in sportsClubContext.Sports)

{

if (sport.Id == sportId)

{

sportIdExists = true;

}

}

if (!sportIdExists)

{

throw new ArgumentException("No sport with such id");

}

}

private void CheckIfTrainerExists(int? trainerId)

{

bool trainerIdExists = false;

foreach (var trainer in sportsClubContext.Trainers)

{

if (trainer.Id == trainerId)

{

trainerIdExists = true;

}

}

if (!trainerIdExists)

{

throw new ArgumentException("No trainer with such id");

}

}

private void CheckIfTeamNameIsCorrect(Team team)

{

if (team.Name == "" || team.Name == null)

{

throw new ArgumentException("Team name can't be empty");

}

}

}

}

* Class TrainerBusiness

namespace SportsClub.Business

{

public class TrainerBusiness

{

private SportsClubContext sportsClubContext;

public List<Trainer> GetAll()

{

using (sportsClubContext = new SportsClubContext())

{

return sportsClubContext.Trainers.ToList();

}

}

public Trainer Get(int? id)

{

using (sportsClubContext = new SportsClubContext())

{

return sportsClubContext.Trainers.Find(id);

}

}

public void Add(Trainer trainer)

{

using (sportsClubContext = new SportsClubContext())

{

CheckIfTrainerNameIsCorrect(trainer);

sportsClubContext.Trainers.Add(trainer);

sportsClubContext.SaveChanges();

}

}

public void Update(Trainer trainer)

{

using (sportsClubContext = new SportsClubContext())

{

var item = sportsClubContext.Trainers.Find(trainer.Id);

if (item != null)

{

CheckIfTrainerNameIsCorrect(trainer);

sportsClubContext.Entry(item).CurrentValues.SetValues(trainer);

sportsClubContext.SaveChanges();

}

}

}

public void Delete(int id)

{

using (sportsClubContext = new SportsClubContext())

{

var item = sportsClubContext.Trainers.Find(id);

if (item != null)

{

foreach (var team in sportsClubContext.Teams)

{

if(team.TrainerId==item.Id)

{

TeamBusiness teamBusiness = new TeamBusiness();

teamBusiness.Delete(team.Id);

}

}

try

{

sportsClubContext.Trainers.Remove(item);

}

catch

{

sportsClubContext.Trainers.Remove(item);

}

sportsClubContext.SaveChanges();

}

}

}

private void CheckIfTrainerNameIsCorrect(Trainer trainer)

{

if (trainer.Name == "" || trainer.Name == null)

{

throw new ArgumentException("Trainer name can't be empty");

}

}

}

}

Папка Data

* + Class SportsClubContext

namespace SportsClub.Data

{

public partial class SportsClubContext : DbContext

{

public SportsClubContext()

{

}

public SportsClubContext(DbContextOptions<SportsClubContext> options)

: base(options)

{

}

public virtual DbSet<Player> Players { get; set; }

public virtual DbSet<Sport> Sports { get; set; }

public virtual DbSet<Team> Teams { get; set; }

public virtual DbSet<Trainer> Trainers { get; set; }

protected override void OnConfiguring(DbContextOptionsBuilder optionsBuilder)

{

if (!optionsBuilder.IsConfigured)

{

optionsBuilder.UseSqlServer("Server = .\\SQLEXPRESS; Database= SportsClub; Integrated Security=True;");

}

}

protected override void OnModelCreating(ModelBuilder modelBuilder)

{

modelBuilder.Entity<Player>(entity =>

{

entity.Property(e => e.Name)

.IsRequired()

.HasMaxLength(50)

.IsUnicode(false);

entity.HasOne(d => d.Team)

.WithMany(p => p.Player)

.HasForeignKey(d => d.TeamId)

.HasConstraintName("FK\_\_Player\_\_TeamId\_\_47DBAE45");

});

modelBuilder.Entity<Sport>(entity =>

{

entity.Property(e => e.Name)

.IsRequired()

.HasMaxLength(50)

.IsUnicode(false);

});

modelBuilder.Entity<Team>(entity =>

{

entity.Property(e => e.Name)

.IsRequired()

.HasMaxLength(50)

.IsUnicode(false);

entity.HasOne(d => d.Sport)

.WithMany(p => p.Team)

.HasForeignKey(d => d.SportId)

.HasConstraintName("FK\_\_Team\_\_SportId\_\_3E52440B");

entity.HasOne(d => d.Trainer)

.WithMany(p => p.Team)

.HasForeignKey(d => d.TrainerId)

.HasConstraintName("FK\_\_Team\_\_TrainerId\_\_3F466844");

});

modelBuilder.Entity<Trainer>(entity =>

{

entity.Property(e => e.Name)

.IsRequired()

.HasMaxLength(50)

.IsUnicode(false);

});

OnModelCreatingPartial(modelBuilder);

}

partial void OnModelCreatingPartial(ModelBuilder modelBuilder);

}

}

* Папка Models
  + Class Player

namespace SportsClub.Data.Models

{

public class Player

{

public int Id { get; set; }

public string Name { get; set; }

public int? Age { get; set; }

public int? TeamId { get; set; }

public virtual Team Team { get; set; }

}

}

* + Class Sport

namespace SportsClub.Data.Models

{

public class Sport

{

public Sport()

{

Team = new HashSet<Team>();

}

public int Id { get; set; }

public string Name { get; set; }

public virtual ICollection<Team> Team { get; set; }

}

}

* + Class Team

namespace SportsClub.Data.Models

{

public class Team

{

public Team()

{

Player = new HashSet<Player>();

}

public int Id { get; set; }

public string Name { get; set; }

public int? SportId { get; set; }

public int? TrainerId { get; set; }

public virtual Sport Sport { get; set; }

public virtual Trainer Trainer { get; set; }

public virtual ICollection<Player> Player { get; set; }

}

}

* + Class Trainer

namespace SportsClub.Data.Models

{

public class Trainer

{

public Trainer()

{

Team = new HashSet<Team>();

}

public int Id { get; set; }

public string Name { get; set; }

public virtual ICollection<Team> Team { get; set; }

}

}

Папка Presentation

* + Class Display

namespace SportsClub.Presentation

{

public static class Display

{

private static int menuExitOperation = 6;

private static int tablesExitOperation = 5;

private static PlayerDisplay playerDisplay = new PlayerDisplay();

private static TeamDisplay teamDisplay = new TeamDisplay();

private static TrainerDisplay trainerDisplay = new TrainerDisplay();

private static SportDisplay sportDisplay = new SportDisplay();

public static void GetInput()

{

var operation = -1;

do

{

ShowMenu();

try

{

operation = GetIntNumber(operation);

switch (operation)

{

case 1:

{

ChooseTableToListAllFrom();

GetBackToMenu();

break;

}

case 2:

{

ChooseTableToAddEntryTo();

GetBackToMenu();

break;

}

case 3:

{

ChooseTableToUpdateEntryFrom();

GetBackToMenu();

break;

}

case 4:

{

ChooseTableToFetchEntryFrom();

GetBackToMenu();

break;

}

case 5:

{

ChooseTableToDeleteEntryFrom();

GetBackToMenu();

break;

}

default:

{

if (operation != menuExitOperation)

{

Console.WriteLine("Not an operation number");

GetBackToMenu();

}

break;

}

}

}

catch (ArgumentException exception)

{

Console.WriteLine(exception.Message);

GetBackToMenu();

}

} while (operation != menuExitOperation);

}

private static void ShowMenu()

{

Console.Clear();

Console.WriteLine("MENU:");

Console.WriteLine("1. List all entries");

Console.WriteLine("2. Add new entry");

Console.WriteLine("3. Update entry");

Console.WriteLine("4. Fetch entry by ID");

Console.WriteLine("5. Delete entry by ID");

Console.WriteLine("6. Exit");

Console.Write("Enter operation number: ");

}

private static void ShowTableNames()

{

Console.WriteLine("1. Players");

Console.WriteLine("2. Teams");

Console.WriteLine("3. Trainers");

Console.WriteLine("4. Sports");

Console.WriteLine("5. Exit");

Console.Write("Enter operation number: ");

}

private static void ChooseTableToListAllFrom()

{

var operation = -1;

do

{

Console.Clear();

Console.WriteLine("List all from:");

ShowTableNames();

try

{

operation = GetIntNumber(operation);

switch (operation)

{

case 1:

{

Console.Clear();

playerDisplay.ListAll();

GetBackToMenu();

break;

}

case 2:

{

Console.Clear();

teamDisplay.ListAll();

GetBackToMenu();

break;

}

case 3:

{

Console.Clear();

trainerDisplay.ListAll();

GetBackToMenu();

break;

}

case 4:

{

Console.Clear();

sportDisplay.ListAll();

GetBackToMenu();

break;

}

default:

{

if (operation != tablesExitOperation)

{

Console.WriteLine("Not an operation number");

GetBackToMenu();

}

break;

}

}

}

catch (ArgumentException exception)

{

Console.WriteLine(exception.Message);

GetBackToMenu();

}

} while (operation != tablesExitOperation);

}

private static void ChooseTableToAddEntryTo()

{

var operation = -1;

do

{

Console.Clear();

Console.WriteLine("Add to:");

ShowTableNames();

try

{

operation = GetIntNumber(operation);

switch (operation)

{

case 1:

{

Console.Clear();

playerDisplay.Add();

GetBackToMenu();

break;

}

case 2:

{

Console.Clear();

teamDisplay.Add();

GetBackToMenu();

break;

}

case 3:

{

Console.Clear();

trainerDisplay.Add();

GetBackToMenu();

break;

}

case 4:

{

Console.Clear();

sportDisplay.Add();

GetBackToMenu();

break;

}

default:

{

if (operation != tablesExitOperation)

{

Console.WriteLine("Not an operation number");

GetBackToMenu();

}

break;

}

}

}

catch (ArgumentException exception)

{

Console.WriteLine(exception.Message);

GetBackToMenu();

}

} while (operation != tablesExitOperation);

}

private static void ChooseTableToUpdateEntryFrom()

{

var operation = -1;

do

{

Console.Clear();

Console.WriteLine("Update from:");

ShowTableNames();

try

{

operation = GetIntNumber(operation);

switch (operation)

{

case 1:

{

Console.Clear();

playerDisplay.Update();

GetBackToMenu();

break;

}

case 2:

{

Console.Clear();

teamDisplay.Update();

GetBackToMenu();

break;

}

case 3:

{

Console.Clear();

trainerDisplay.Update();

GetBackToMenu();

break;

}

case 4:

{

Console.Clear();

sportDisplay.Update();

GetBackToMenu();

break;

}

default:

{

if (operation != tablesExitOperation)

{

Console.WriteLine("Not an operation number");

GetBackToMenu();

}

break;

}

}

}

catch (ArgumentException exception)

{

Console.WriteLine(exception.Message);

GetBackToMenu();

}

} while (operation != tablesExitOperation);

}

private static void ChooseTableToFetchEntryFrom()

{

var operation = -1;

do

{

Console.Clear();

Console.WriteLine("Fetch from:");

ShowTableNames();

try

{

operation = GetIntNumber(operation);

switch (operation)

{

case 1:

{

Console.Clear();

playerDisplay.Fetch();

GetBackToMenu();

break;

}

case 2:

{

Console.Clear();

teamDisplay.Fetch();

GetBackToMenu();

break;

}

case 3:

{

Console.Clear();

trainerDisplay.Fetch();

GetBackToMenu();

break;

}

case 4:

{

Console.Clear();

sportDisplay.Fetch();

GetBackToMenu();

break;

}

default:

{

if (operation != tablesExitOperation)

{

Console.WriteLine("Not an operation number");

GetBackToMenu();

}

break;

}

}

}

catch (ArgumentException exception)

{

Console.WriteLine(exception.Message);

GetBackToMenu();

}

} while (operation != tablesExitOperation);

}

private static void ChooseTableToDeleteEntryFrom()

{

var operation = -1;

do

{

Console.Clear();

Console.WriteLine("Delete from:");

ShowTableNames();

try

{

operation = GetIntNumber(operation);

switch (operation)

{

case 1:

{

Console.Clear();

playerDisplay.Delete();

GetBackToMenu();

break;

}

case 2:

{

Console.Clear();

teamDisplay.Delete();

GetBackToMenu();

break;

}

case 3:

{

Console.Clear();

trainerDisplay.Delete();

GetBackToMenu();

break;

}

case 4:

{

Console.Clear();

sportDisplay.Delete();

GetBackToMenu();

break;

}

default:

{

if (operation != tablesExitOperation)

{

Console.WriteLine("Not an operation number");

GetBackToMenu();

}

break;

}

}

}

catch (ArgumentException exception)

{

Console.WriteLine(exception.Message);

GetBackToMenu();

}

} while (operation != tablesExitOperation);

}

public static int GetIntNumber(int number)

{

try

{

number = int.Parse(Console.ReadLine());

return number;

}

catch

{

throw new ArgumentException("This should be written with numbers only");

}

}

public static int? GetIntNumber(int? number)

{

try

{

number = int.Parse(Console.ReadLine());

return number;

}

catch

{

throw new ArgumentException("This should be written with numbers only");

}

}

private static void GetBackToMenu()

{

Console.WriteLine();

Console.WriteLine("Press enter to get back to menu...");

Console.ReadLine();

}

}

}

* + Class PlayerDisplay

namespace SportsClub.Presentation

{

public class PlayerDisplay

{

private PlayerBusiness playerBusiness = new PlayerBusiness();

private TeamBusiness teamBusiness = new TeamBusiness();

public void ListAll()

{

Console.WriteLine("Players:");

List<Player> players = playerBusiness.GetAll();

if (players.Count != 0)

{

foreach (var item in players)

{

Console.WriteLine();

Console.WriteLine($"Id: {item.Id}");

Console.WriteLine($"Name: {item.Name}");

Console.WriteLine($"Age: {item.Age}");

var team = teamBusiness.Get(item.TeamId);

Console.WriteLine($"Team: {team.Name}");

}

}

else

{

Console.WriteLine("No players");

}

}

public void Add()

{

Player player = new Player();

Console.WriteLine("Enter Name:");

player.Name = Console.ReadLine();

Console.WriteLine("Enter Age:");

player.Age = Display.GetIntNumber(player.Age);

Console.WriteLine("Enter TeamId:");

player.TeamId = Display.GetIntNumber(player.TeamId);

playerBusiness.Add(player);

}

public void Update()

{

Console.WriteLine("Enter Id:");

int id = 0;

id = Display.GetIntNumber(id);

Player player = playerBusiness.Get(id);

if (player != null)

{

Console.WriteLine("Enter new name:");

player.Name = Console.ReadLine();

Console.WriteLine("Enter new age:");

player.Age = Display.GetIntNumber(player.Age);

Console.WriteLine("Enter new team id:");

player.TeamId = Display.GetIntNumber(player.TeamId);

playerBusiness.Update(player);

}

else

{

Console.WriteLine("Player not found");

}

}

public void Fetch()

{

Console.WriteLine("Enter id:");

int id = 0;

id = Display.GetIntNumber(id);

Player player = playerBusiness.Get(id);

if (player != null)

{

Console.WriteLine($"Id: {player.Id}");

Console.WriteLine($"Name: {player.Name}");

Console.WriteLine($"Age: {player.Age}");

var team = teamBusiness.Get(player.TeamId);

Console.WriteLine($"Team: {team.Name}");

}

else

{

Console.WriteLine("Player not found");

}

}

public void Delete()

{

Console.WriteLine("Enter id:");

int id = 0;

id = Display.GetIntNumber(id);

playerBusiness.Delete(id);

}

}

}

* + Class SportDisplay

namespace SportsClub.Presentation

{

public class SportDisplay

{

private SportBusiness sportBusiness = new SportBusiness();

public void ListAll()

{

Console.WriteLine("Sports:");

List<Sport> sports = sportBusiness.GetAll();

if (sports.Count != 0)

{

foreach (var item in sports)

{

Console.WriteLine();

Console.WriteLine($"Id: {item.Id}");

Console.WriteLine($"Name: {item.Name}");

}

}

else

{

Console.WriteLine("No sports");

}

}

public void Add()

{

Sport sport = new Sport();

Console.WriteLine("Enter Name:");

sport.Name = Console.ReadLine();

sportBusiness.Add(sport);

}

public void Update()

{

Console.WriteLine("Enter Id:");

int id = 0;

id = Display.GetIntNumber(id);

Sport sport = sportBusiness.Get(id);

if (sport != null)

{

Console.WriteLine("Enter new name:");

sport.Name = Console.ReadLine();

sportBusiness.Update(sport);

}

else

{

Console.WriteLine("Sport not found");

}

}

public void Fetch()

{

Console.WriteLine("Enter id:");

int id = 0;

id = Display.GetIntNumber(id);

Sport sport = sportBusiness.Get(id);

if (sport != null)

{

Console.WriteLine($"Id: {sport.Id}");

Console.WriteLine($"Name: {sport.Name}");

}

else

{

Console.WriteLine("Sport not found");

}

}

public void Delete()

{

Console.WriteLine("Enter id:");

int id = 0;

id = Display.GetIntNumber(id);

sportBusiness.Delete(id);

}

}

}

* + Class TeamDisplay

namespace SportsClub.Presentation

{

public class TeamDisplay

{

private TeamBusiness teamBusiness = new TeamBusiness();

private TrainerBusiness trainerBusiness = new TrainerBusiness();

private SportBusiness sportBusiness = new SportBusiness();

public void ListAll()

{

Console.WriteLine("Teams:");

List<Team> teams = teamBusiness.GetAll();

if (teams.Count != 0)

{

foreach (var item in teams)

{

Console.WriteLine();

Console.WriteLine($"Id: {item.Id}");

Console.WriteLine($"Name: {item.Name}");

var sport = sportBusiness.Get(item.SportId);

Console.WriteLine($"Sport: {sport.Name}");

var trainer = trainerBusiness.Get(item.TrainerId);

Console.WriteLine($"Trainer: {trainer.Name}");

}

}

else

{

Console.WriteLine("No teams");

}

}

public void Add()

{

Team team = new Team();

Console.WriteLine("Enter Name:");

team.Name = Console.ReadLine();

Console.WriteLine("Enter SportId:");

team.SportId = Display.GetIntNumber(team.SportId);

Console.WriteLine("Enter TrainerId:");

team.TrainerId = Display.GetIntNumber(team.TrainerId);

teamBusiness.Add(team);

}

public void Update()

{

Console.WriteLine("Enter Id:");

int id = 0;

id = Display.GetIntNumber(id);

Team team = teamBusiness.Get(id);

if (team != null)

{

Console.WriteLine("Enter new name:");

team.Name = Console.ReadLine();

Console.WriteLine("Enter new sport id:");

team.SportId = Display.GetIntNumber(team.SportId);

Console.WriteLine("Enter new trainer id:");

team.TrainerId = Display.GetIntNumber(team.TrainerId);

teamBusiness.Update(team);

}

else

{

Console.WriteLine("Team not found");

}

}

public void Fetch()

{

Console.WriteLine("Enter id:");

int id = 0;

id = Display.GetIntNumber(id);

Team team = teamBusiness.Get(id);

if (team != null)

{

Console.WriteLine($"Id: {team.Id}");

Console.WriteLine($"Name: {team.Name}");

var sport = sportBusiness.Get(team.SportId);

Console.WriteLine($"Sport: {sport.Name}");

var trainer = trainerBusiness.Get(team.TrainerId);

Console.WriteLine($"Trainer: {trainer.Name}");

}

else

{

Console.WriteLine("Team not found");

}

}

public void Delete()

{

Console.WriteLine("Enter id:");

int id = 0;

id = Display.GetIntNumber(id);

teamBusiness.Delete(id);

}

}

}

* + Class TrainerDisplay

namespace SportsClub.Presentation

{

public class TrainerDisplay

{

private TrainerBusiness trainerBusiness = new TrainerBusiness();

public void ListAll()

{

Console.WriteLine("Trainers:");

List<Trainer> trainers = trainerBusiness.GetAll();

if (trainers.Count != 0)

{

foreach (var item in trainers)

{

Console.WriteLine();

Console.WriteLine($"Id: {item.Id}");

Console.WriteLine($"Name: {item.Name}");

}

}

else

{

Console.WriteLine("No trainers");

}

}

public void Add()

{

Trainer trainer = new Trainer();

Console.WriteLine("Enter Name:");

trainer.Name = Console.ReadLine();

trainerBusiness.Add(trainer);

}

public void Update()

{

Console.WriteLine("Enter Id:");

int id = 0;

id = Display.GetIntNumber(id);

Trainer trainer = trainerBusiness.Get(id);

if (trainer != null)

{

Console.WriteLine("Enter new name:");

trainer.Name = Console.ReadLine();

trainerBusiness.Update(trainer);

}

else

{

Console.WriteLine("Trainer not found");

}

}

public void Fetch()

{

Console.WriteLine("Enter id:");

int id = 0;

id = Display.GetIntNumber(id);

Trainer trainer = trainerBusiness.Get(id);

if (trainer != null)

{

Console.WriteLine($"Id: {trainer.Id}");

Console.WriteLine($"Name: {trainer.Name}");

}

else

{

Console.WriteLine("Trainer not found");

}

}

public void Delete()

{

Console.WriteLine("Enter id:");

int id = 0;

id = Display.GetIntNumber(id);

trainerBusiness.Delete(id);

}

}

}

Class Program

namespace SportsClub

{

class Program

{

static void Main(string[] args)

{

Display.GetInput();

}

}

}