Валидация

Валидация на променими и непроменими типове данни

Учителски екип

Обучение за ИТ кариера

https://it-kariera.mon.bg/e-learning/

Съдържание

1. Валидация
2. Промениими типове данни
3. Непроменими типове данни

Валидация

- Валидацията на данни се случва в setters

```
public double Salary
{
  set
  {
    if (salary < 460)
      throw new ArgumentException("...");
    this.salary = value;
  }
}
```

По-добре е да се „хвърли“ изключение, отколкото да се извежда на екрана

- Сътрудник на вашия клас трябва да се грижи за обработка на изключенията

Валидация (2)

- Конструкторите използват private setter с валидационна логика

```
public Person(string firstName, string lastName,
              int age, double salary)
 {
  this.FirstName = firstName;
  this.LastName = lastName;
  this.Age = age;
  this.Salary = salary;
}
```

Валидацията се случва в setter-а

- Гарантират валидно състояние на обекта при неговото създаване

Задача: Валидация на данни

- Разширете Person с валидация за всяко поле
- Names трябва да са с не по-малко от 3 символа
- Age не може да е нула или отрицателно
- Salary да не е по-малко от 460

Person
-firstName : String -lastName : String -age : Integer -salary : Double
+Person() +FirstName(string fname) +LastName(string lname) +Age(int age) +Salary(double salary)

Задача: Валидация на данни

```
TODO: Add validation for firstName
TODO: Add validation for lastName
private void setAge(int age)
{
  if (age < 1)
    throw new ArgumentException("...");
  this.age = age;
}
TODO: Add validation for salary
```

Непроменими (Immutable) обекти

- Когато имате препратка (reference) към инстанция на обект, съдържанието, на която **не може** да бъде променяно

```
string myString = "old String"
Console.WriteLine( myString );
myString.replaceAll( "old", "new" );
Console.WriteLine( myString );
```

```
old String
old String
```

Промениими (Mutable) обекти

- Когато имате препратка (reference) към инстанция на обект, съдържанието, на която **може** да бъде променяно

```
Point myPoint = new Point( 0.0, 0.0 );
Console.WriteLine( myPoint );
myPoint.setLocation( 1.0, 0.0 );
Console.WriteLine( myPoint );
```

```
0.0, 0.0
1.0, 0.0
```

Промени полета

- Променимите `private` полета все още не са капсулирани

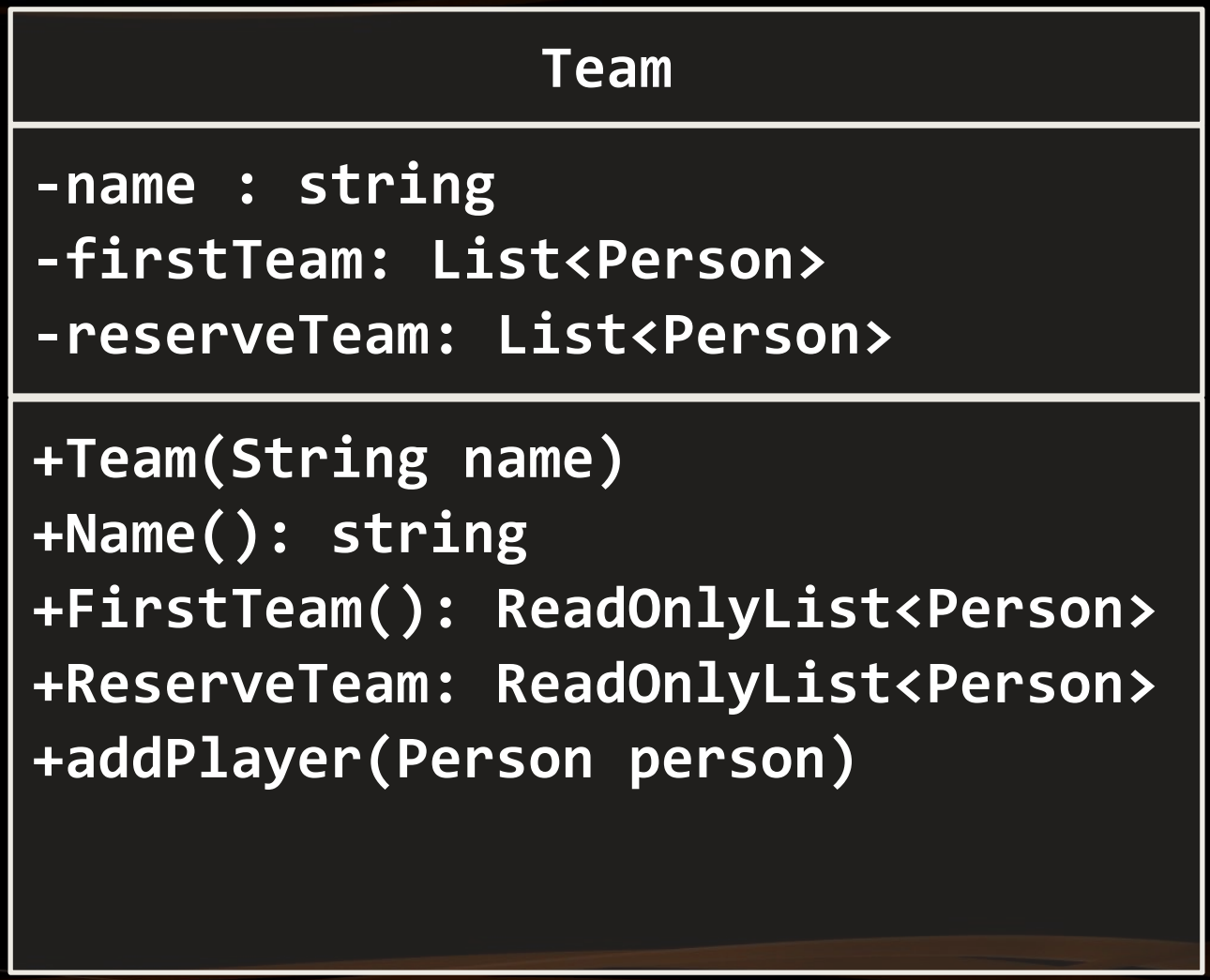
```
class Team {
  private string name;
  private List<Person> players;

  public List<Person> Players
  {
    get { return this.players; }
  }
}
```

- Тогава getter-а е също и setter

Задача: Първи и резервен отбор

- Разширете вашия проект с клас Team
- Team трябва да има два комплекта отбори първи отбор и втори отбор
- Въведете persons от клавиатурата и ги добавете към отбора
- Ако те са по-млади от 40, тогава ги добавете към първи отбор
- Изведете броя на играчите на всеки отбор



Решение: Валидиране на данни

```
private string name;
private List<Person> firstTeam;
private List<Person> reserveTeam;

public Team(string name)
{
  this.name = name;
  this.firstTeam = new List<Person>();
  this.reserveTeam = new List<Person>();
}
```

Решение: Валидиране на данни

```
public IReadOnlyCollection<Person> FirstTeam
{
 get { return this.firstTeam.AsReadOnly(); }
}
//TODO: add getter for reserve team
public void AddPlayer(Person player)
{
  if (player.Age < 40)
    firstTeam.Add(player);
  else
    reserveTeam.Add(player);
}
```

Обобщение

- С помощта на модификаторите за достъп можем да извършваме валидация на данните
- При замяна на непроменими типове с променими в private полета (с цел бързодействие и пестене на ресурс), трябва да знаем, че с private се гарантира защитен достъп само до адресите, в които се пазят данните, но не и самите данни

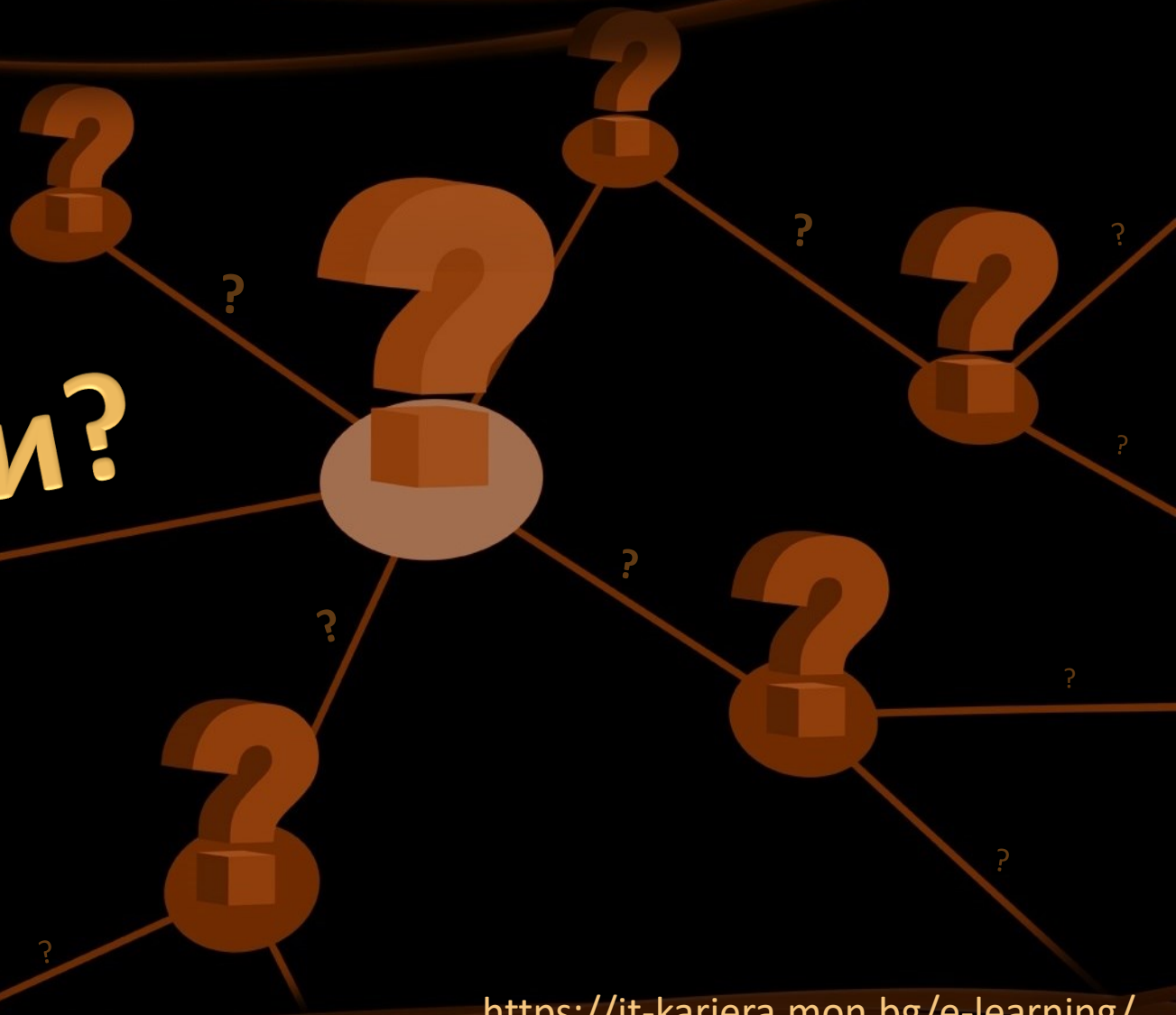
Капсулация – ползи

- Позволяват валидации и data binding
- Променими обекти
- Непромениими обекти

```
public void addPlayer(Person person) {
    if (person.getAge() < 40) {
        firstTeam.add(person);
    } else {
        reserveTeam.add(person);
    }
}
```

Валидация

Въпроси?



https://it-kariera.mon.bg/e-learning/

Министерство на образованието и науката (МОН)

- Настоящият курс (презентации, примери, задачи, упражнения и др.) е разработен за нуждите на Национална програма "**Обучение за ИТ кариера**" на МОН за подготовка по професия "Приложен програмист"

- Курсът е базиран на учебно съдържание и методика, предоставени от **фондация "Софтуерен университет"** и се разпространява под свободен лиценз **CC-BY-NC-SA**