Java SE Programming Language

Lab guide

Document Code	**25e-BM/HR/HDCV/FSOFT**
Version	**1.1**
Effective Date	**20/11/2012**

Hanoi, 06/2019

RECORD OF CHANGES

No	Effective Date	Change Description	Reason	Reviewer	Approver
1.	01/Oct/2018	Create new	Draft		
2.	01/Jun/2019	Update template	Fsoft template	DieuNT1	VinhNV

Contents

FRESHER ACADEMY		
	CODE:	JPL.S.L701
	TYPE:	SHORT
	LOC:	70
	DURATION:	45 MINUTES

Unit 11: Exception Handling

Objectives:

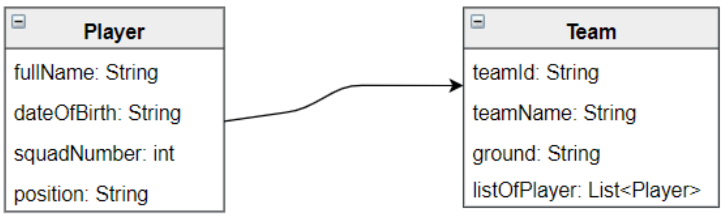
» Understand basic concept of Exception in Java.

» Understand how to create custom Exception type in Java.

» Understand how to handle Exception in Java.

Assignment Specifications:

For the class hierarchy is as follows, the trainee let's create the java classes install this class diagram to be able to relationship between it.



» The Player class contains the information about players. Each player has its **fullName**, **dateOfBirth**, **squadNumber**, **position**.

» The Team class has contains about teams. Each team has an **teamId**, **teamName**, **ground**, **listOfPlayer**.

Functional Requirements:

a. The program has a functions to validate data, that call in setter method.

b. Create custom exception named “**IncorrectFormatException**” that will be thrown when input data incorrect format.

c. The program has a function to create a team(s) and user can add player(s) for each team.

d. The program has a function to display list of team information.

Business Rules:

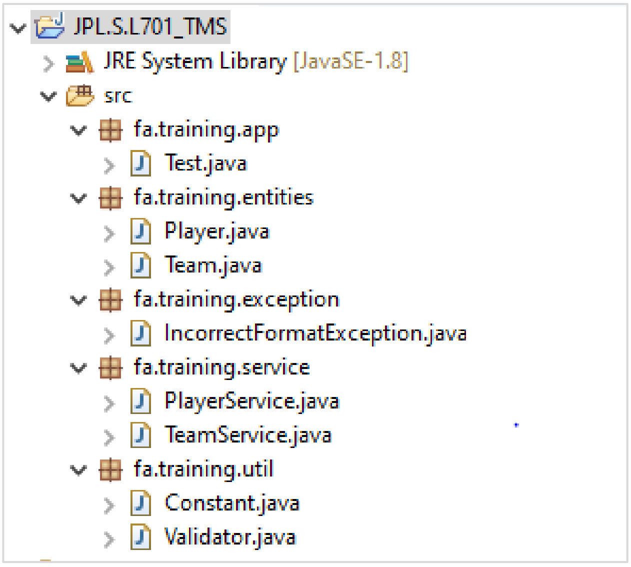
» Date of birth: must be date format.

» Squad number: must be a number.

» Position: only one of (“GK”, “CB”, “WB”, “CM”,”CF”).

Guidelines:

» Step1. Create a project named **JPL.S.L701_TMS** in Eclipse as below:



```
JPL.S.L701_TMS
  JRE System Library [JavaSE-1.8]
  src
    fa.training.app
      Test.java
    fa.training.entities
      Player.java
      Team.java
    fa.training.exception
      IncorrectFormatException.java
    fa.training.service
      PlayerService.java
      TeamService.java
    fa.training.util
      Constant.java
      Validator.java
```

» Step2. Create package **fa.training.entities** that contains **Player**, **Team** classes:

Player class

```java
package fa.training.entities;

import java.text.ParseException;
import java.text.SimpleDateFormat;

import fa.training.exception.IncorrectFormatException;
import fa.training.util.Constant;
import fa.training.util.Validator;

public class Player {
    private String fullName;
    private String dateOfBirth;
    private int squadNumber;
    private String position;

    /**
     * Constructor for Player class without Parameters.
     */
    public Player() {
        super();
    }

    /**
     * Constructor for Player class with Parameters.
     */
    public Player(String fullName, String dateOfBirth, int squadNumber, String position){
        super();
        this.fullName = fullName;
        this.dateOfBirth = dateOfBirth;
        this.squadNumber = squadNumber;
        this.position = position;
    }

    public String getFullName() {
        return fullName;
    }

    public void setFullName(String fullName) {
        this.fullName = fullName;
    }

    public String getDateOfBirth() {
        return dateOfBirth;
    }

    /**
     * @param dateOfBirth the dateOfBirth to set
     * @throws ParseException,IncorrectFormatException
     */
    public void setDateOfBirth(String dateOfBirth)
                                throws ParseException, IncorrectFormatException {
        SimpleDateFormat dateFormat = new SimpleDateFormat("dd/mm/yyy");
        try {
            dateOfBirth = dateFormat.format(dateFormat.parse(dateOfBirth));
            if (Validator.isDate(dateOfBirth)) {
                this.dateOfBirth = dateOfBirth;
            } else {
                throw new IncorrectFormatException(Constant.INCORRECT_DATE_MASSAGE);
            }
        } catch (ParseException e) {
            throw e;
        }
    }

    public int getSquadNumber() {
        return squadNumber;
    }

    public void setSquadNumber(String squadNumber) throws NumberFormatException {
        this.squadNumber = Integer.parseInt(squadNumber);

    }

    public String getPosition() {
        return position;
    }

    /**
     * @param position the position to set
     * @throws IncorrectFormatException
     */
    public void setPosition(String position) throws IncorrectFormatException {
        if (Validator.isPosition(position)) {
            this.position = position;
        } else {
            throw new IncorrectFormatException(Constant.INCORRECT_POSITION_MASSAGE);
        }
    }

    /*
     * Method to display Player information.
     */
    @Override
    public String toString() {
        return "\n fullName:" + fullName +  ", dateOfBirth:"      + dateOfBirth +
                ", squadNumber:" + squadNumber  +
                ", position:"  + position;
    }
}
```

Team class

```java
package fa.training.entities;

import java.util.List;

public class Team {

    private String teamId;
    private String teamName;
    private String ground;
    private List<Player> listOfPlayer;
    /**
     *  Constructor for Team class without Parameters.
     */
    public Team() {
        super();
    }

    /**
     * Constructor for Team class without Parameters.
     * @param teamId
     * @param teamName
     * @param ground
     * @param listOfPlayer
     */
    public Team(String teamId,String teamName, String ground, List<Player> listOfPlayer){
        super();
        this.teamId = teamId;
        this.teamName = teamName;
        this.ground = ground;
        this.listOfPlayer = listOfPlayer;
    }

    /**
     * @return the teamId
     */
    public String getTeamId() {
        return teamId;
    }

    /**
     * @param teamId the teamId to set
     */
    public void setTeamId(String teamId) {
        this.teamId = teamId;
    }

    /**
     * @return the teamName
     */
    public String getTeamName() {
        return teamName;
    }

    /**
     * @param teamName the teamName to set
     */
    public void setTeamName(String teamName) {
        this.teamName = teamName;
    }

    /**
     * @return the ground
     */
    public String getGround() {
        return ground;
    }
```

```java
     /**
      * @param ground the ground to set
      */
    public void setGround(String ground) {
        this.ground = ground;
    }

    /**
     * @return the listOfPlayer
     */
    public List<Player> getListOfPlayer() {
        return listOfPlayer;
    }

    /**
     * @param listOfPlayer the listOfPlayer to set
     */
    public void setListOfPlayer(List<Player> listOfPlayer) {
        this.listOfPlayer = listOfPlayer;
    }

    /*
     * Method to display Team information.
     */
    @Override
    public String toString() {
        return "teamId:" + teamId + ", teamName:" + teamName +
               ", ground:" + ground +"\n"+ ", listOfPlayer:"  + listOfPlayer;
    }
}
```

» Step3. Create package **fa.training.util** that contains **Constant, Validator** classes:

Constants class:

```java
package fa.training.util;

public class Constant {
    // message
    public static final String INCORRECT_DATE_MASSAGE = "Date value incorrect format!";
    public static final String INCORRECT_POSITION_MASSAGE =
                                                "Position value incorrect format!";
    // Regex pattern
    public static final String POSITION_PATTERN = "^GK|CB|WB|CM|CF$";
    public static final String DATE_PATTERN = "^([0-2][0-9]|(3)[0-1])(\\/)(((0)[0-
9])|((1)[0-2]))(\\/)\\d{4}$";
}
```

Validator class:

```java
package fa.training.util;

import java.util.regex.Matcher;
import java.util.regex.Pattern;

public class Validator {

    private static Matcher matcher = null;
    private static Pattern pattern = null;

    /**
     * This method check format of date value.
     *
     * @param String date.
     * @return boolean
     */
    public static boolean isDate(String date) {
        pattern = Pattern.compile(Constant.DATE_PATTERN);
        matcher = pattern.matcher(date);
        return matcher.matches();
    }

    /**
     * This method check format of "position" attribute.
     *
     * @param String position.
     * @return boolean
     */
    public static boolean isPosition(String position) {
        pattern = Pattern.compile(Constant.POSITION_PATTERN);
        matcher = pattern.matcher(position);
        return matcher.matches();
    }
}
```

» Step4. Create package **fa.tranning.exception** that contains **IncorrectFormatException** classes:

IncorrectFormatException class:

```java
package fa.training.exception;

public class IncorrectFormatException extends Exception {
    private static final long serialVersionUID = 1L;

    /**
     * Constructor for IncorrectFomartException without Parameters.
     */
    public IncorrectFormatException() {
        super();
    }

    /**
     * Constructor for IncorrectFomartException with message.
     */
    public IncorrectFormatException(String message) {
        super(message);
    }
 }
```

» Step5. Create package **fa.training.service** that contains **PlayerService, TeamService** classes:

PlayerService class:

```java
package fa.training.service;

import java.text.ParseException;
import java.util.Scanner;

import fa.training.entities.Player;
import fa.training.exception.IncorrectFormatException;

public class PlayerService {

    Player player = null;

    /**
     * This method take input player's information from keyboard.
     *
     * @param scanner
     * @return Player
     * @throws ParseException,IncorrectFormatException
     */
    public Player inputPlayer(Scanner scanner)
                                throws ParseException, IncorrectFormatException {

        // Create new player and set attributes.
        player = new Player();
        String fullName, dateOfBirth, position, squadNumber;
        System.out.print("\nEnter full name: ");
        fullName = scanner.nextLine();
        player.setFullName(fullName);
        System.out.print("\nEnter date of birth(dd/mm/yyyy): ");
        dateOfBirth = scanner.nextLine();
        player.setDateOfBirth(dateOfBirth);
        System.out.print("\nEnter squad number: ");
        squadNumber = scanner.nextLine();
        player.setSquadNumber(squadNumber);
        System.out.print("\nEnter position( GK|CB|WB|CM|CF):");
        position = scanner.nextLine();
        player.setPosition(position);
        return player;
    }
}
```

TeamService class:

```java
package fa.training.service;

import java.text.ParseException;
import java.util.ArrayList;
import java.util.List;
import java.util.Scanner;

import fa.training.entities.Player;
import fa.training.entities.Team;
import fa.training.exception.IncorrectFormatException;

public class TeamService {
    private String teamId;
    private String teamName;
    private String ground;
    private List<Player> listOfPlayer;
    private PlayerService playerService = new PlayerService();

    /**
     * This method create take input team's information from keyboard.
     *
     * @param scanner: Scanner
     * @return Team
     * @throws ParseException, IncorrectFormatException
     */
    public Team createNewTeam(Scanner scanner)
                                throws ParseException, IncorrectFormatException {

        System.out.println("Input team information!");
        String choice = "y";
        listOfPlayer = new ArrayList<>();
        System.out.print("Enter team ID:");
        teamId = scanner.nextLine();
        System.out.print("Enter full name of team:");
        teamName = scanner.nextLine();
        System.out.print("Enter ground:");
        ground = scanner.nextLine();
        System.out.print("Do you want to add new player for this team? (y/n)");
        choice = scanner.nextLine();
        Team team = new Team(teamId, teamName, ground, listOfPlayer);
        if (choice.equalsIgnoreCase("y")) {
            try {
                addNewPlayerToTeam(scanner, team);
            } catch (ParseException | IncorrectFormatException e) {
                System.out.println("Add player fail, because" + e.getMessage());
                e.printStackTrace();
                throw e;
            }
        }
        return team;
    }

    /**
     * This method add player(s) to exist team.
     *
     * @param scanner
     * @param team
     * @return Boolean
     * @throws ParseException, IncorrectFormatException
     */
    public Boolean addNewPlayerToTeam(Scanner scanner, Team team)
                                throws ParseException, IncorrectFormatException {
        Player player = null;
        System.out.print("\nInput new player information!");
        player = playerService.inputPlayer(scanner);
        return team.getListOfPlayer().add(player);
    }

    /**
     * This method display all team's information.
     *
     * @param teams:List<Team>.
     */
    public void displayTeam(List<Team> teams) {
        System.out.println("Team information!");
        for (Team team : teams) {
            System.out.println(team.toString());
        }
    }
}
```

» Step6. Create package **fa.training.app** that contains **Test** classes:

Test class:

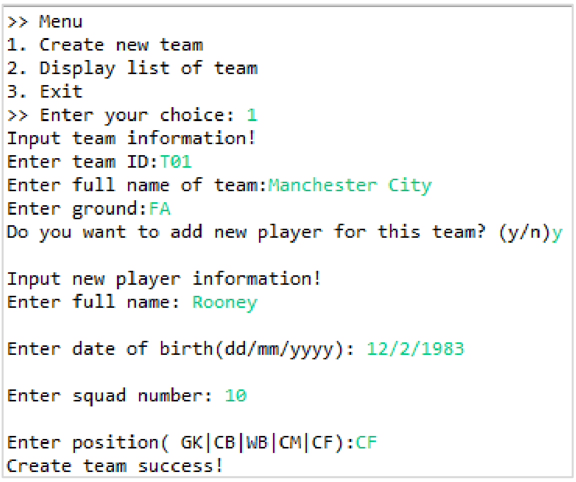
```
package fa.training.app;

import java.util.ArrayList;
import java.util.List;
import java.util.Scanner;

import fa.training.entities.Team;
import fa.training.service.TeamService;

public class Test {
    public static List<Team> teams = new ArrayList<>();
    static TeamService teamService = new TeamService();
    static Scanner scanner = new Scanner(System.in);

    public static void main(String[] args) {
        Team team = null;
        String key = "3";
        Loop: do {
            showMenu();
            key = scanner.nextLine();
            switch (key) {
            case "1":
                try {
                    team = teamService.createNewTeam(scanner);
                    teams.add(team);
                    System.out.println("Create team success!");
```

```java
                        continue Loop;
                    } catch (Exception e) {
                        System.err.println("Create team fail!");
                        break Loop;
                    }
                case "2":
                    teamService.displayTeam(teams);
                    break;
                default:
                    break Loop;
                }
            } while (true);
    }

    public static void showMenu() {
        System.out.println(">> Menu");
        System.out.println("1. Create new team");
        System.out.println("2. Display list of team");
        System.out.println("3. Exit");
        System.out.print(">> Enter your choice: ");
    }
}
```

Outputs:

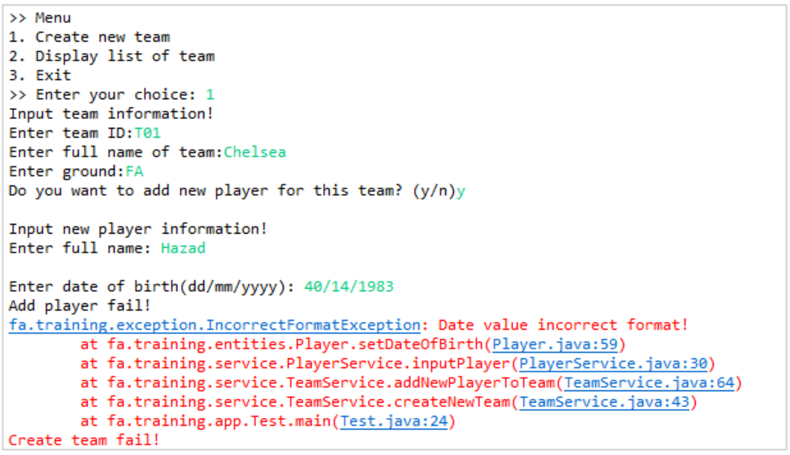
Select 1: Create new team.

» Trường hợp valid data



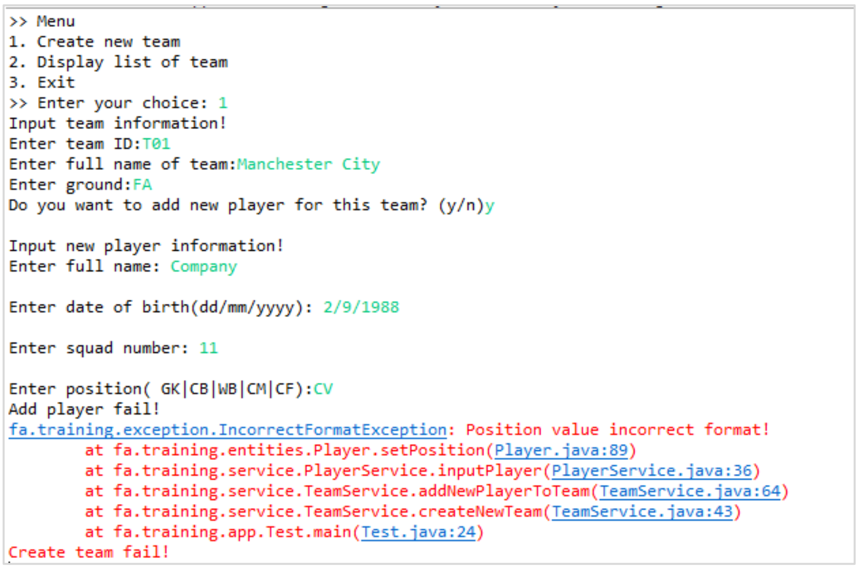
```
>> Menu
1. Create new team
2. Display list of team
3. Exit
>> Enter your choice: 1
Input team information!
Enter team ID:T01
Enter full name of team:Manchester City
Enter ground:FA
Do you want to add new player for this team? (y/n)y

Input new player information!
Enter full name: Rooney

Enter date of birth(dd/mm/yyyy): 12/2/1983

Enter squad number: 10

Enter position( GK|CB|WB|CM|CF):CF
Create team success!
```

» Invalid birth of date



```
>> Menu
1. Create new team
2. Display list of team
3. Exit
>> Enter your choice: 1
Input team information!
Enter team ID:T01
Enter full name of team:Chelsea
Enter ground:FA
Do you want to add new player for this team? (y/n)y

Input new player information!
Enter full name: Hazad

Enter date of birth(dd/mm/yyyy): 40/14/1983
Add player fail!
fa.training.exception.IncorrectFormatException: Date value incorrect format!
	at fa.training.entities.Player.setDateOfBirth(Player.java:59)
	at fa.training.service.PlayerService.inputPlayer(PlayerService.java:30)
	at fa.training.service.TeamService.addNewPlayerToTeam(TeamService.java:64)
	at fa.training.service.TeamService.createNewTeam(TeamService.java:43)
	at fa.training.app.Test.main(Test.java:24)
Create team fail!
```

» Invalid birth of position



```
>> Menu
1. Create new team
2. Display list of team
3. Exit
>> Enter your choice: 1
Input team information!
Enter team ID:T01
Enter full name of team:Manchester City
Enter ground:FA
Do you want to add new player for this team? (y/n)y

Input new player information!
Enter full name: Company

Enter date of birth(dd/mm/yyyy): 2/9/1988

Enter squad number: 11

Enter position( GK|CB|WB|CM|CF):CV
Add player fail!
fa.training.exception.IncorrectFormatException: Position value incorrect format!
        at fa.training.entities.Player.setPosition(Player.java:89)
        at fa.training.service.PlayerService.inputPlayer(PlayerService.java:36)
        at fa.training.service.TeamService.addNewPlayerToTeam(TeamService.java:64)
        at fa.training.service.TeamService.createNewTeam(TeamService.java:43)
        at fa.training.app.Test.main(Test.java:24)
Create team fail!
```

-- THE END --