FACHHOCHSCHULE ERFURT UNIVERSITY OF APPLIED SCIENCES
Angewandte Informatik

Präsentation
Team 2 – Service 7 – Modul Persons

Milena Neumann
Luisa Oswald
Tran Anh Hoang

Gliederung

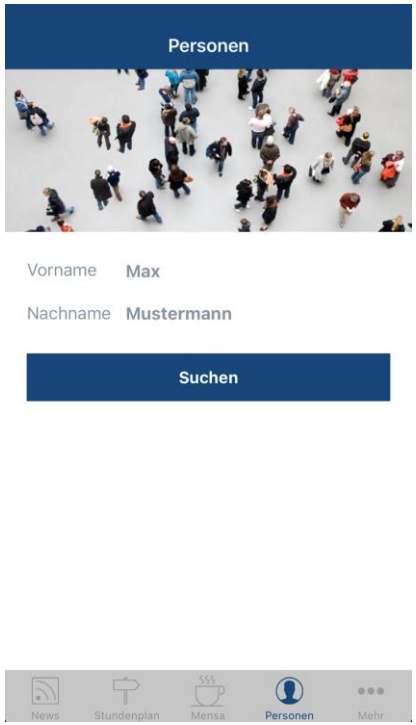
FACHHOCHSCHULE ERFURT UNIVERSITY OF APPLIED SCIENCES
Angewandte Informatik

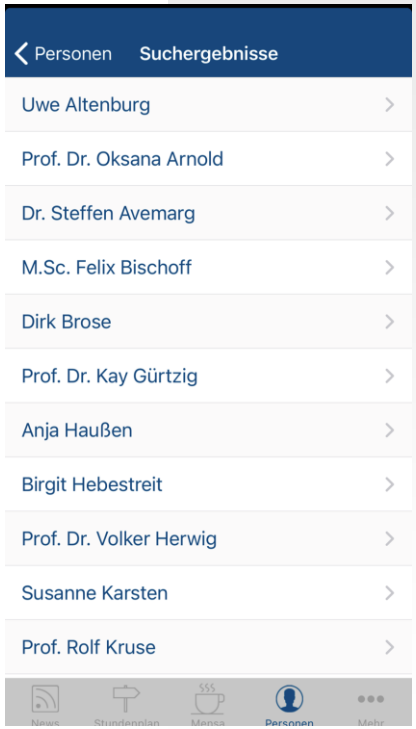
1. Themenbeschreibung
2. Tran Anh Hoang: CSearching Result, CSearchingHistory,
3. Milena Neumann: CInputField, CDeleting
4. Luisa Oswald: CNewPersonDataSet
5. Zukünftige Meilensteine
6. Dokumentation

1.Themenvorstellung

FACHHOCHSCHULE ERFURT UNIVERSITY OF APPLIED SCIENCES
Angewandte Informatik

- Modul „Persons“ – Suchvorgang aller Mitarbeiter und wichtige Rollen der Fachhochschule Erfurt





2.1 CSearchingResult - Hoang

FACHHOCHSCHULE ERFURT UNIVERSITY OF APPLIED SCIENCES
Angewandte Informatik

CSearchingResult

- ArrayList<String> SearchInputData
- Map<Integer, ArrayList <String>> InputDataHashMap
- Map<Integer, ArrayList <String>> PersonDataSet
- int HashMapKey

+ setSearchInputData: boolean
+ setListInputMap: void
+ setPersonDataSet(): void
+ getInputDataHashMap(): Map<Integer, ArrayList<String>>
+ getKey: <K, V>
+ compareDataInMaps(): ServiceCreateNewPerson

```
//Dataset von CInputField
private ArrayList<String> SearchInputData                    = new ArrayList<>();
private Map<Integer, ArrayList <String>> InputDataHashMap    = new HashMap<>();
//-----------------------------------------------------------------------------

//Dataset von Class CPersons
private Map<Integer, ArrayList <String>> PersonDataSet       = new HashMap<>();;
//-----------------------------------------------------------------------------

private int HashMapKey                                        = 0;
//-----------------------------------------------------------------------------
```

2.1 CSearchingResult - Hoang

FACHHOCHSCHULE ERFURT UNIVERSITY OF APPLIED SCIENCES
Angewandte Informatik

CSearchingResult

- ArrayList<String> SearchInputData
- Map<Integer, ArrayList <String>> InputDataHashMap
- Map<Integer, ArrayList <String>> PersonDataSet
- int HashMapKey

+ setSearchInputData: boolean
+ setListInputMap: void
+ setPersonDataSet(): void
+ getInputDataHashMap(): Map<Integer, ArrayList<String>>
+ getKey: <K, V>
+ compareDataInMaps(): ServiceCreateNewPerson

```java
public boolean setSearchInputData()
{
    this.SearchInputData = CInputField.getListSearchInput();


    //überprüft ob Daten in SearchInputData gesetzt wurde
    return !SearchInputData.isEmpty();
}



public void setListInputMap()
{
    //setzt Key und SearchInputList in die HashMap
    this.InputDataHashMap.put(this.HashMapKey,this.SearchInputData);

    //Increment HashMapKey
    this.HashMapKey = this.HashMapKey + 1;

}


//lädt alle gespeichert Werte von der verkürzten Variante der HashMap einer PersonenDataSet
public void setPersonDataSet(int CountNumberOfValue, ArrayList<String> PersonData)
{
    CPerson PersonDataSet       = new CPerson();
    this.PersonDataSet = PersonDataSet.getPersonDataSet();
}
```

2.1 CSearchingResult - Hoang

FACHHOCHSCHULE ERFURT UNIVERSITY OF APPLIED SCIENCES Angewandte

CSearchingResult

- ArrayList<String> SearchInputData
- Map<Integer, ArrayList <String>> InputDataHashMap
- Map<Integer, ArrayList <String>> PersonDataSet
- int HashMapKey

+ setSearchInputData: boolean
+ setListInputMap: void
+ setPersonDataSet(): void
+ getInputDataHashMap(): Map<Integer, ArrayList<String>>
+ getKey: <K, V>
+ compareDataInMaps(): ServiceCreateNewPerson

```java
//Methode gibt den HashMapschlüssel zurück, welches den gesuchten Wert entspricht
public  <K, V> K getKey(Map<K, V> map, V value)
{
    for (Map.Entry<K, V> entry: map.entrySet())
    {
        if (value.equals(entry.getValue())) {
            return entry.getKey();
        }
    }
    return null;
}


/*
- Um einen schnellen Vergleich und eine kurze Datentransfer zu ermöglich, benötigen wir 2 verschiede
- Wir nutzen eine "verkürzte HashMap", welches nur diese folgenden Datenstruktur hat:
    --> ID/HashmapKey
    --> FirstName
    --> LastName
    --> Modul
    --> Faculty
 */
private ServiceCreateNewPerson compareDataInMaps()
{
    /*
        - zunächst wird mittels eines HashSet überprüft ob die Liste des UserInput auch in der Certa
        - Ist dieser vorhanden, so wird die Methode getKey() aufgerufen, welches aus den gematchten
        - dieser ist identisch zu den Key, welches den HashMap der gesamten Daten der Person enthält
        - zurück gegeben wird ein Objekt Person mit ihren gesamten Datensatz und Methoden
     */

    HashSet<ArrayList> set1 = new HashSet<>(InputDataHashMap.values());
    HashSet<ArrayList> set2 = new HashSet<>(CertainPersonDataHashMap.values());

    if (set1.equals(set2))
    {
        int PersonID = getKey(CertainPersonDataHashMap, SearchInputData);
        return getPersonDataSetByID(PersonID);
    }

    return null;
}


//Interface --> Gibt Object Person zurück, welches durch die PsersonID fest definiert wird
//-----------------------------------------------------------------------------------------
//Hier wird ein Object Person zurück gegeben, die die gesamte Datensätze besitzt und die dazugöriger
//-----------------------------------------------------------------------------------------


public ArrayList<String> SearchHistoryList()
{
    return SearchInputData;
}
```

2.1 CSearchingResult - Hoang

FACHHOCHSCHULE ERFURT UNIVERSITY OF APPLIED SCIENCES
Angewandte Informatik

<<Schnittstelle>>

PersonClient

getPersonDataSetByID(int PersonID): ServiceCreateNewPerson

```
public interface PersonClient
{
   ServiceCreateNewPerson getPersonDataSetByID(int PersonID);
}
```

```
@Override
public ServiceCreateNewPerson getPersonDataSetByID(int PersonID)
{
    return CPerson.searchDataSetByID(PersonID);
}
```

2.2 CSearchingHistory - Hoang

FACHHOCHSCHULE ERFURT UNIVERSITY OF APPLIED SCIENCES
Angewandte Informatik

CSearchingHistory

- ArrayList <String> UserInputList
- LinkedHashMap<Integer, String> SearchHistoryMap
- SubSringFirstName: String
- SubSringLastName: String
- DisplayName: String

+ setSubStringLastNameFromArrayList
+ setSubSringFirstNameFromArrayList: void
+ setSearchHistoryMap(): void
- chechSizeOfSearchHistoryMap(): void
- clearEntireMap(): void

```
private ArrayList <String> UserInputList                    = new ArrayList<>();
private LinkedHashMap<Integer, String> SearchHistoryMap     = new LinkedHashMap<>();

private String SubStringFirstName;
private String SubStringLastName;
private String DisplayName;

public int FirstElementOfArrayList = 0;
public int SecondElementOfArrayList = 1;

private int HashMapKey = 0;
```

2.2 CSearchingHistory - Hoang

CSearchingHistory

- ArrayList <String> UserInputList
- LinkedHashMap<Integer, String> SearchHistoryMap
- SubSringFirstName: String
- SubSringLastName: String
- DisplayName: String

+ setSubStringLastNameFromArrayList
+ setSubSringFirstNameFromArrayList: void
+ setSearchHistoryMap(): void
- chechSizeOfSearchHistoryMap(): void
- clearEntireMap(): void

```java
//Set the Searching Input from the User in the UserInputList - ArrayList
public void setSearchHistoryList()
{
    CSearchingResult result = new CSearchingResult();
    UserInputList = result.SearchHistoryList();
}

//Set the first Element from the ArrayList in SubStringFirstName
public void setSubStringFirstNameFromArrayList()
{
    this.SubStringFirstName = UserInputList.get(FirstElementOfArrayList);
}

//Set the second Element from the ArrayList in SubStringLastName
public void setSubStringLastNameFromArrayList()
{
    this.SubStringLastName = UserInputList.get(SecondElementOfArrayList);
}


//concat SubStringLastName and SubStringFirstName to DisplayName
public void setDisplayName()
{
    DisplayName = SubStringFirstName + SubStringFirstName;
}

//Set the Key and Values for the SearchHistoryMap
public void setSearchHistoryMap()
{
    SearchHistoryMap.put(HashMapKey, DisplayName);
    this.HashMapKey = HashMapKey + 1;
}

/*
    - Check size of SearchHistoryMap
    - If size of SearchHistoryMap --> delete the first Element of the Map
 */
public void checkSizeOfSearchHistoryMap()
{
    if (SearchHistoryMap.size() > 5)
    {
        SearchHistoryMap.remove(FirstMapKey);
    }
}

//Clear entire Map
private void clearEntireMap()
{
    SearchHistoryMap.clear();
}
```

3.1 CInputField -Milena

FACHHOCHSCHULE ERFURT UNIVERSITY OF APPLIED SCIENCES
Angewandte

CInputField

- inFirstName:String
- inLastName:String
- inModul: String
- inFaculty: EnumFaculty
- List<String> PersonInputData:

+ checkFieldInputValid(): boolean
+ checkFacultyValid(): string
+ setListSearchInput(): void
+ getListSearchInput():ArrayList<String>
+ checkStringValid(): boolean

```java
public boolean checkStringValid(String StringToCheck)   // public damit auch andere auf d
{
    char FirstCharacterOfTheString = StringToCheck.charAt(0);

    if (StringToCheck.length() > 30) {
        System.out.println("Der eingegebene Name ist zu lang!");
        return false;
    }

    if (FirstCharacterOfTheString == ' ') {
        System.out.println("Das erste Zeichen darf kein Leerzeichen sein, bitte prüfen");
        return false;
    }

    if (StringToCheck == null || StringToCheck.trim().isEmpty())
    {
        System.out.println("Die Zeichenkette hat ein falsches Format");
        return false;
    }
    Pattern p = Pattern.compile("[^A-Za-z0-9]");
    Matcher m = p.matcher(StringToCheck);
    // boolean b = m.matches();
    boolean b = m.find();
    if (b)
    {
        System.out.println("Es ist ein nicht erlaubtes Zeichen in der Eingabe");
        return false;
    }
    return true;
```

3.1 CInputField -Milena

FACHHOCHSCHULE ERFURT UNIVERSITY OF APPLIED SCIENCES
Angewandte

```java
private String checkFacultyValid(Faculty inFaculty)
{
    String result;
    switch(inFaculty)
    {
        case GEAI:
            result = "Gebäudetechnik und Informatik";
        break;
        case LAGAF:
            result = "Landschaftsarchitektur, Gartenbau und Forst";
        break;
        case WLV:
            result = "Wirtschaft-Logistik-Verkehr";
        break;
        case BKR:
            result = "Bauingenieurwesen und Konservierung/Restaurierung";
        break;
        case ASP:
            result = "Architektur und Stadtplanung";
        break;
        case AS:
            result = "Angewandte Sozialwissenschaften";
            break;

        default:
            System.out.println("Die eingegebene Fakultät existiert nicht od
            result = " ";
        break;

    }
    return result;
}
```

```java
private void setListSearchInput()
{
    PersonInputData.add(this.inFirstName);
    PersonInputData.add(this.inLastName);
    PersonInputData.add(this.inModul);
    PersonInputData.add(checkFacultyValid(inFaculty));
}
```

```java
private boolean checkFieldInputValid()
{
    if    ((checkStringValid(this.inFirstName))
        && (checkStringValid(this.inLastName))
        && (checkStringValid(this.inModul)))
    {
        setListSearchInput();  //Parameter werde
        return true;
    }
    return false;
}
```

```java
ArrayList<String> getListSearchInput()
{
    return PersonInputData;
}
```

3.2 CDeleting -Milena

FACHHOCHSCHULE ERFURT UNIVERSITY OF APPLIED SCIENCES
Angewandte Informatik

CDeleting

- HashMap< Boolean , Integer> PersonToDelete
- HashMapKey: int
- PersonID: int

+ setDeletedPerson(): void
+ removePersonFromTableToDelete(): void

```java
public class CDeleting
{
    private int HashMapKey;
    private HashMap<Integer, Integer>PersonsToDelete = new HashMap<>();
    private int PersonID;

    public CDeleting(int HashMapKey, int PersonID) // Konstruktor
    {
        this.PersonID = PersonID;
        this.HashMapKey = HashMapKey;
    }

    void setDeletedPerson()
    {
        PersonsToDelete.put(this.HashMapKey, this.PersonID);
    }

    void removePersonFromTableToDelete()
    {
       PersonsToDelete.remove(HashMapKey);
    }
}
```

4. CNewPersonDataSet - Luisa

FACHHOCHSCHULE ERFURT UNIVERSITY OF APPLIED SCIENCES
Angewandte Informatik

CNewPersonDataSet

- FirstName: String
- LastName: String
- Address: String
- Email: String
- Phonenumber: short
- Title: String
- HireDate: String
- Faculty: Enum
- TeachingFlag: Boolean
- Major: String
- ImmatriculationDate: String
- ExmatriculationDate: String
- TutorFlag: Boolean
- ScientificWorkerFlag: Boolean
- Picture: String
- JobTitle: String
- Room: String

```java
public class CNewPersonDataSet {

    private final String firstname;
    private final String lastname;
    private final String address;
    private final String email;
    private String phonenumber;
    private String title;
    private String hireDate;
    private Enum faculty;
    private Boolean teachingFlag;
    private String room;
    private String major;
    private String immatriculationDate;
    private String exmatriculationDate;
    private Boolean tutorFlag;
    private Boolean scientificWorkerFlag;
    private String jobTitle;
```

4. CNewPersonDataSet - Luisa

FACHHOCHSCHULE ERFURT UNIVERSITY OF APPLIED SCIENCES
Angewandte Informatik

```
//using a builder pattern
private CNewPersonDataSet(CPersonBuilder builder) {

    this.firstname = builder.firstname;
    this.lastname = builder.lastname;
    this.address = builder.address;
    this.email = builder.email;
    this.phonenumber = builder.phonenumber;
    this.title = builder.title;
    this.hireDate = builder.hireDate;
    this.faculty = builder.faculty;
    this.teachingFlag = builder.teachingFlag;
    this.room = builder.room;
    this.major = builder.major;
    this.immatriculationDate = builder.immatriculationDate;
    this.exmatriculationDate = builder.exmatriculationDate;
    this.tutorFlag = builder.tutorFlag;
    this.scientificWorkerFlag = builder.scientificWorkerFlag;
    this.jobTitle = builder.jobTitle;
}
```

4. CNewPersonDataSet - Luisa

FACHHOCHSCHULE ERFURT UNIVERSITY OF APPLIED SCIENCES
Angewandte Informatik

CPersonBuilder

- FirstName: String
- LastName: String
- Address: String
- Email: String
- Phonenumber: short
- Title: String
- HireDate: String
- Faculty: Enum
- TeachingFlag: Boolean
- Major: String
- ImmatriculationDate: String
- ExmatriculationDate: String
- TutorFlag: Boolean
- ScientificWorkerFlag: Boolean
- Picture: String
- AcademicTitle: String
- JobTitle: String
- Room: String

+ PersonBuilder()
+ PersonBuilder otherEmployee()
+ PersonBuilder lecturer()
+ PersonBuilder teachingFlag()
+ PersonBuilder student()
+ CNewPersonDataSet build()

```java
public static class CPersonBuilder {

    private final String firstname;       //required
    private final String lastname;        //required
    private final String address;         //required
    private final String email;           //required
    private String phonenumber;           //optional
    private String title;                 //optional
    private String hireDate;              //optional
    private Enum faculty;                 //optional
    private Boolean teachingFlag;         //optional
    private String room;                  //optional
    private String major;                 //optional
    private String immatriculationDate;   //optional
    private String exmatriculationDate;   //optional
    private Boolean tutorFlag;            //optional
    private Boolean scientificWorkerFlag; //optional
    private String jobTitle;              //optional
```

4. CNewPersonDataSet - Luisa

FACHHOCHSCHULE ERFURT UNIVERSITY OF APPLIED SCIENCES
Angewandte Informatik

```
//attributes that are not optional, they are needed in every new person instance
public CPersonBuilder(String firstname, String lastname, String address, String email){
    this.firstname = firstname;
    this.lastname = lastname;
    this.address = address;
    this.email = email;
}
```

```
//only needed for lecturers
public CPersonBuilder lecturer(String phonenumber, String title, String hireDate, Enum faculty, Boolean teachingFlag, String room){
    this.phonenumber = phonenumber;
    this.title = title;
    this.hireDate = hireDate;
    this.faculty = faculty;
    this.teachingFlag = teachingFlag;
    this.room = room;
    return this;
}
```

4. CNewPersonDataSet - Luisa

FACHHOCHSCHULE ERFURT UNIVERSITY OF APPLIED SCIENCES
Angewandte Informatik

```
//only needed for other employees
public CPersonBuilder otherEmployee(String jobTitle){
    this.jobTitle = jobTitle;
    return this;
}
```

```
//only needed for students
public CPersonBuilder student(String major, String immatriculationDate, String exmatriculationDate, Boolean tutorFlag, Boolean scientificWorkerFlag){
    this.major = major;
    this.immatriculationDate = immatriculationDate;
    this.exmatriculationDate = exmatriculationDate;
    this.tutorFlag = tutorFlag;
    this.scientificWorkerFlag = scientificWorkerFlag;
    return this;
}
```

5. zukünftige Meilensteine

- Fertigstellungen restlichen Klassen
- Fertigstellungen aller Interfaces
- Testszenarien mittels Mokito Test - Framework
- Fertigstellung der Dokumentation im Git Repo

6. Dokumentation

FACHHOCHSCHULE ERFURT UNIVERSITY OF APPLIED SCIENCES
Angewandte Informatik

die Dokumentation befindet sich im ReadMe im Repository und beinhaltet:

- Vorstellung unseres Projektes
- Aufteilung der Arbeit bzw Klassen auf die Gruppenmitglieder
- kurze Beschreibung jeder Klasse und deren Funktion
- kurze Beschreibung und Arbeitsfortschitt der Meilensteine

Vielen Dank für Ihre Aufmerksamkeit