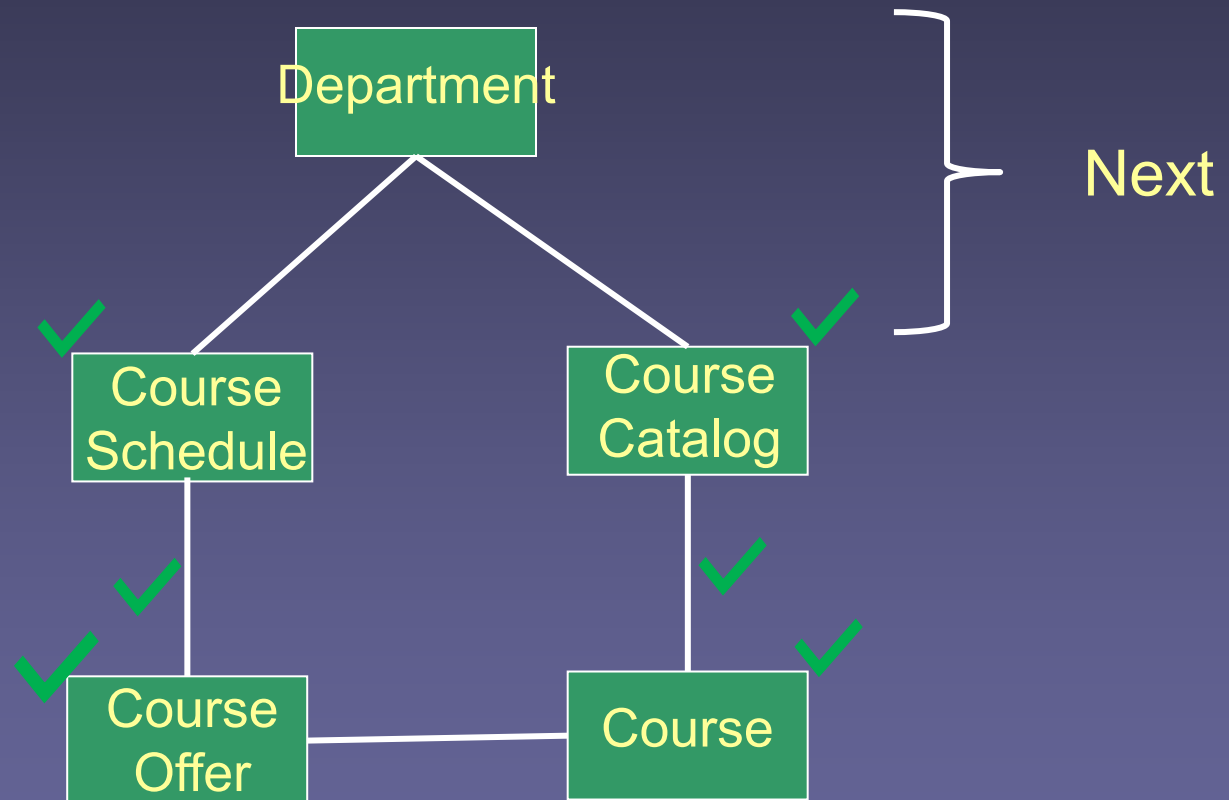

Fundamentals Of Software Engineering

Java Coding Patterns



Department

```
public class Department {

    String name;
    CourseCatalog coursecatalog;
    PersonDirectory persondirectory;
    StudentDirectory studentdirectory;
    FacultyDirectory facultydirectory;
    //EmployerDirectory employerdirectory;

    HashMap<String, CourseSchedule> mastercoursecatalog;

    public Department(String n) {
        name = n;
        mastercoursecatalog = new HashMap<String, CourseSchedule>();
        coursecatalog = new CourseCatalog(this);
        studentdirectory = new StudentDirectory(this); //pass the department
        persondirectory = new PersonDirectory();
    }
}
```

2

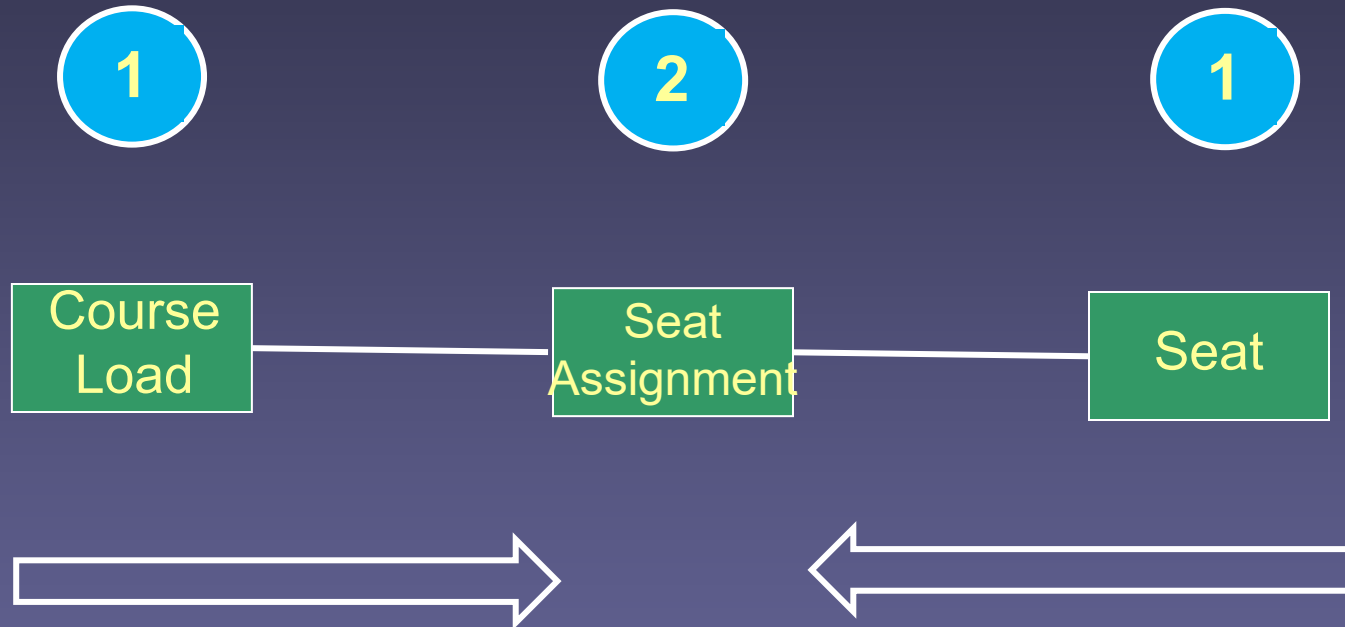
Course
Offer

1

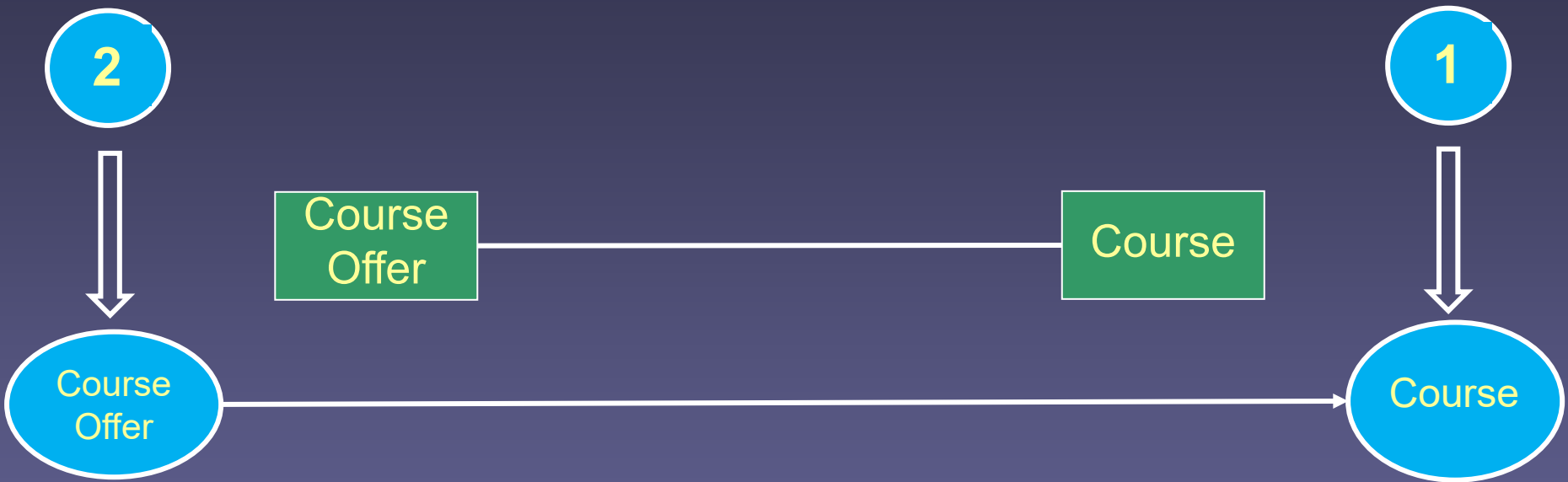
Course







1. Create Course First 2. Link CourseOffer to course

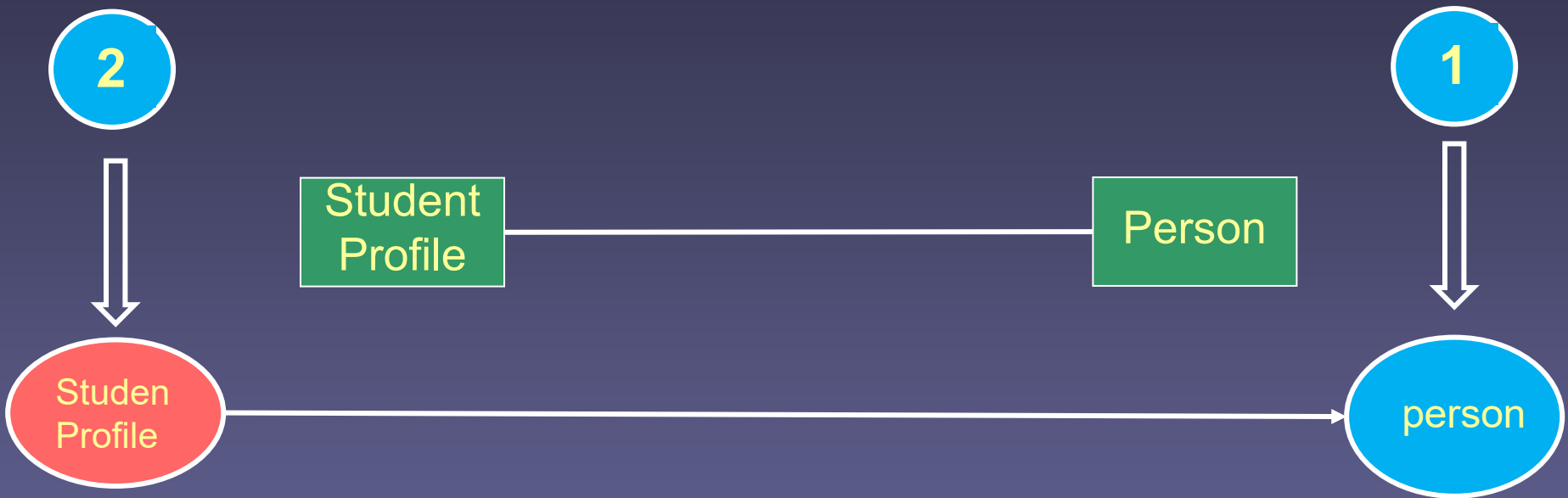


```
public class CourseOffer {
```

```
    Course course;  
    public CourseOffer(Course c){  
        course = c;  
    }  
    public String getCourseNumber(){  
        return course.getCourseNumber();  
    }  
}
```

```
Course c = coursecatalog.getCourseByNumber(n);  
CourseOffer co = new CourseOffer(c);  
schedule.add(co);
```

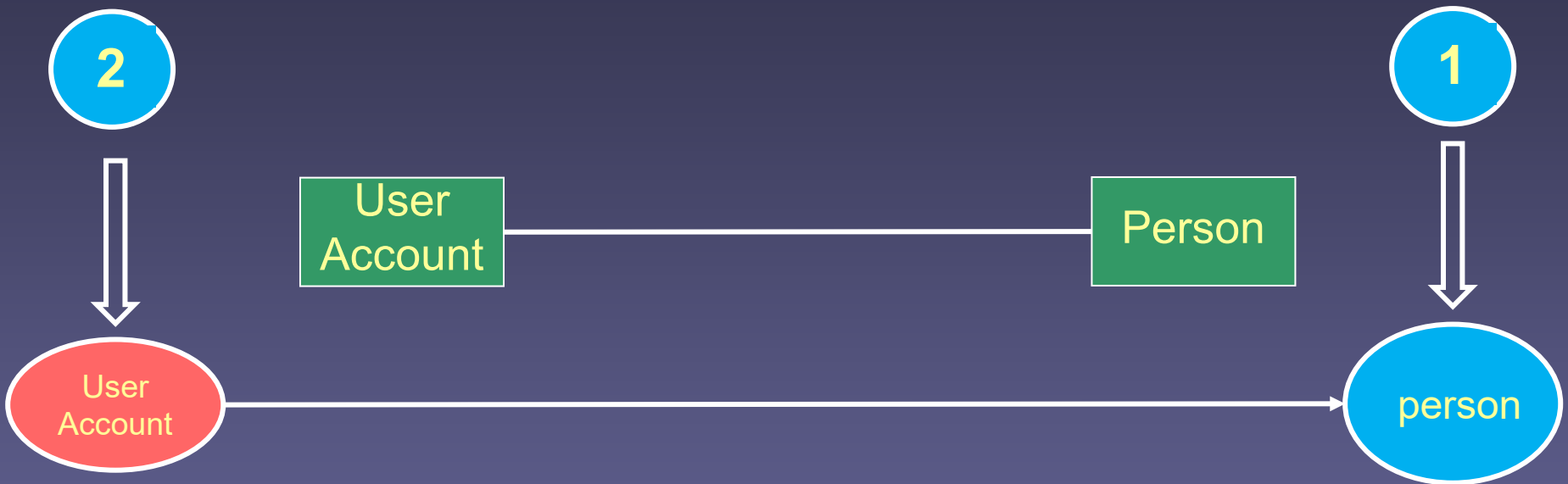

1. Create Person First 2. Link Link StudentProfile next



```
public class StudentProfile {  
    Person person;  
    public StudentProfile(Person p) {  
        person = p;  
    }  
}
```

```
PersonDirectory pd = department.getPersonDirectory();  
Person person = pd.newPerson("0112303");  
StudentDirectory sd = department.getStudentDirectory();  
StudentProfile student = sd.newStudentProfile(person);
```

1. Create Person First 2. Link Link User Account next



```
public class UserAccount {  
    Person person;  
    ``String username;  
    public UserAccount(Person p) {  
        person = p;  
    }  
}
```

```
PersonDirectory pd = department.getPersonDirectory();  
Person person = pd.newPerson("0112303");  
UserADirectory ud = department.getUserADirectory();  
UserAccount student = ud.newUserAccount(person);
```

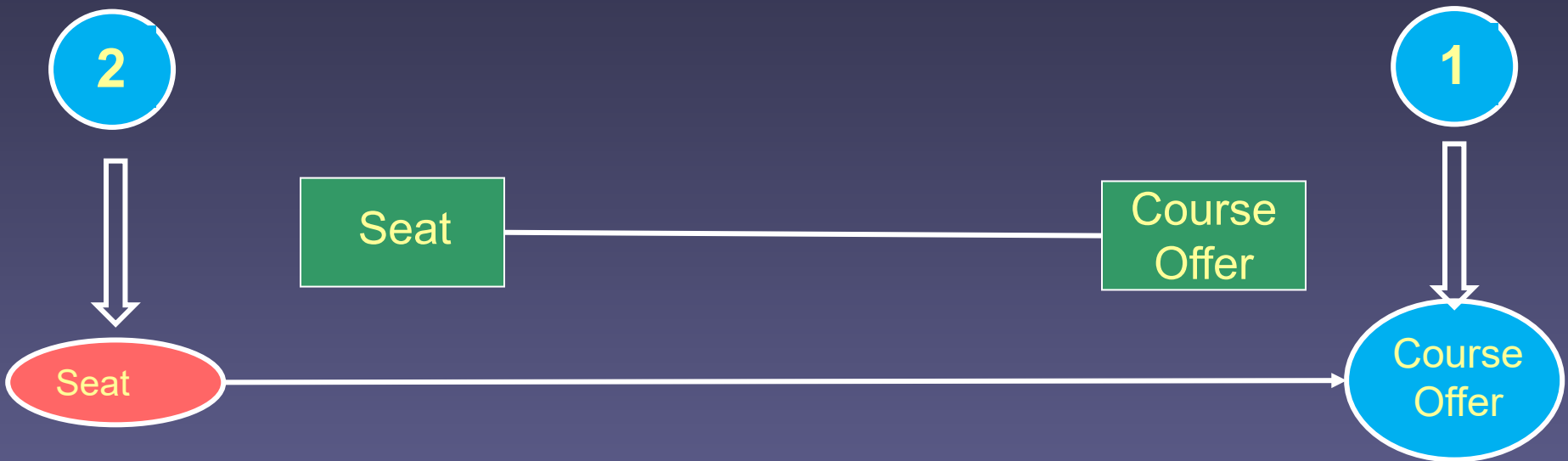
1. Create Person First 2. Link Link StudentProfile next



```
public class Patient {  
    Person person;  
    public Patient(Person p) {  
        person = p;  
    }  
}
```

```
Person person = new Person("0112303");  
Patient patient = new Patient(person);
```

1. Create Course Offer First 2. Link Seat next



```
public class Seat {  
    Boolean occupied;  
    int number;  
    CourseOffer courseoffer;  
    public Seat (CourseOffer co, int n){  
        courseoffer = co;  
        Number = n;  
    }  
}
```

```
Course course = new Course('info 5100",,,)  
CourseOffer co = new CourseOffer(course);  
Seat seat = new Seat(co, 1); //seat numner 1
```

Two ways of registering students

Registering a student
(student side)

Student



Transcript



Course Load

Assigning student to a class (Schedule Side)

Course Schedule



Course Offer



Must find an empty seat

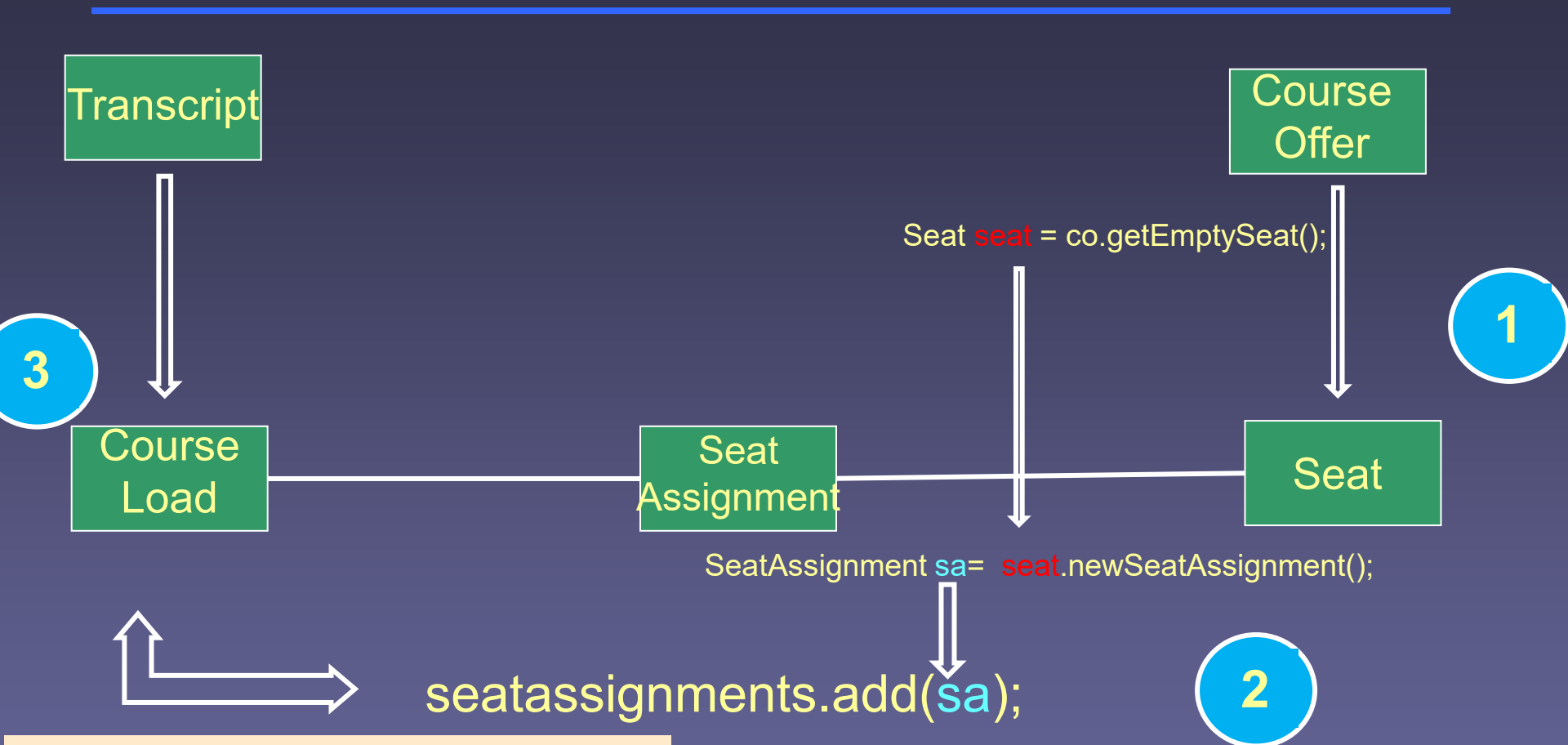
Seat

Student is registered for a class (a fact)



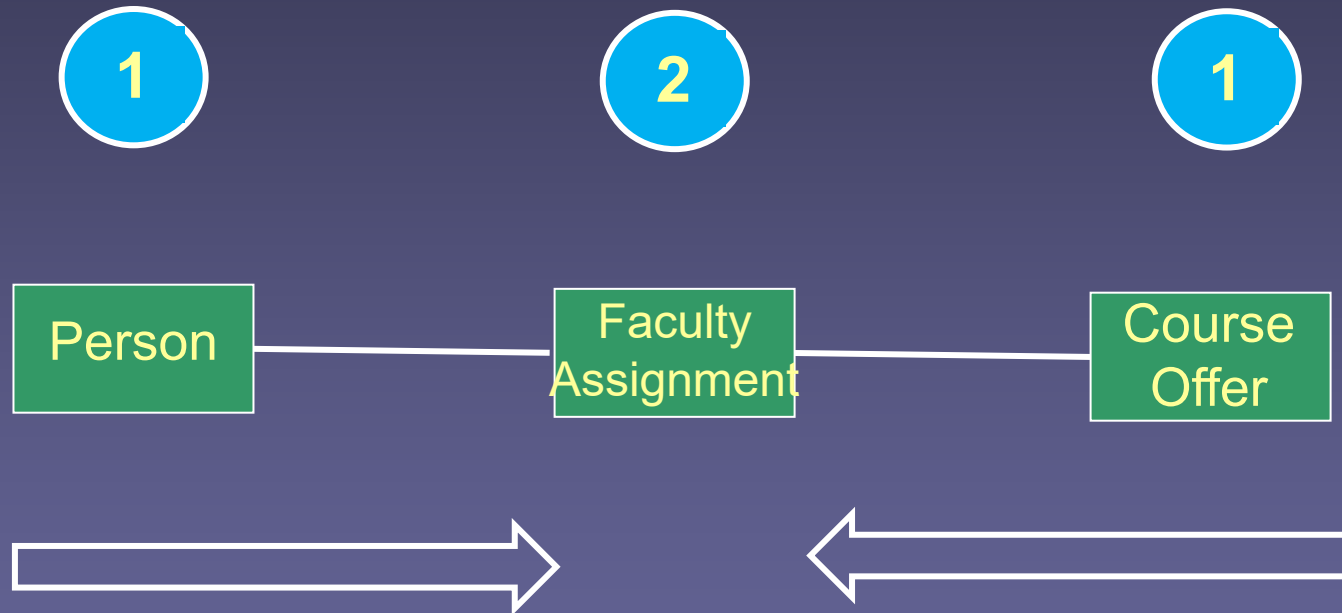
Seat Assignment



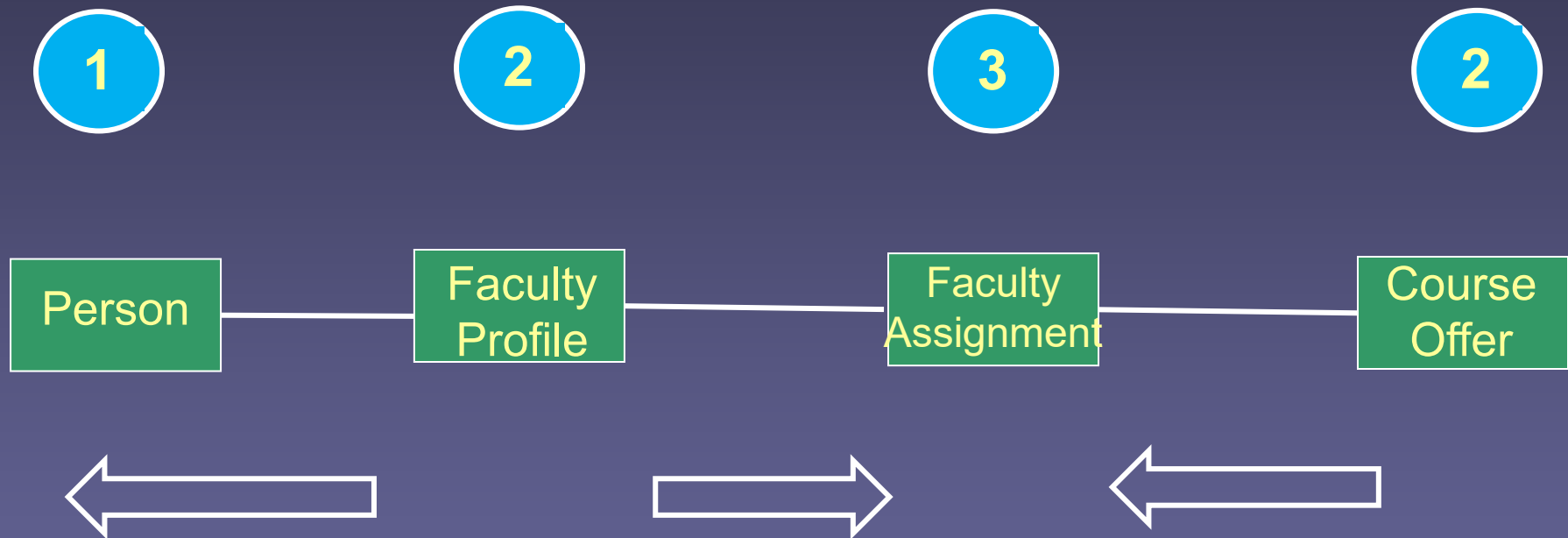


```
public SeatAssignment newSeatAssignment(CourseOffer co){  
  
    Seat seat = co.getEmptySeat(); // seat linked to courseoffer  
    if (seat==null) return null;  
    SeatAssignment sa = seat.newSeatAssignment();  
    seatassignments.add(sa); //add to students course load  
    return sa;  
}
```

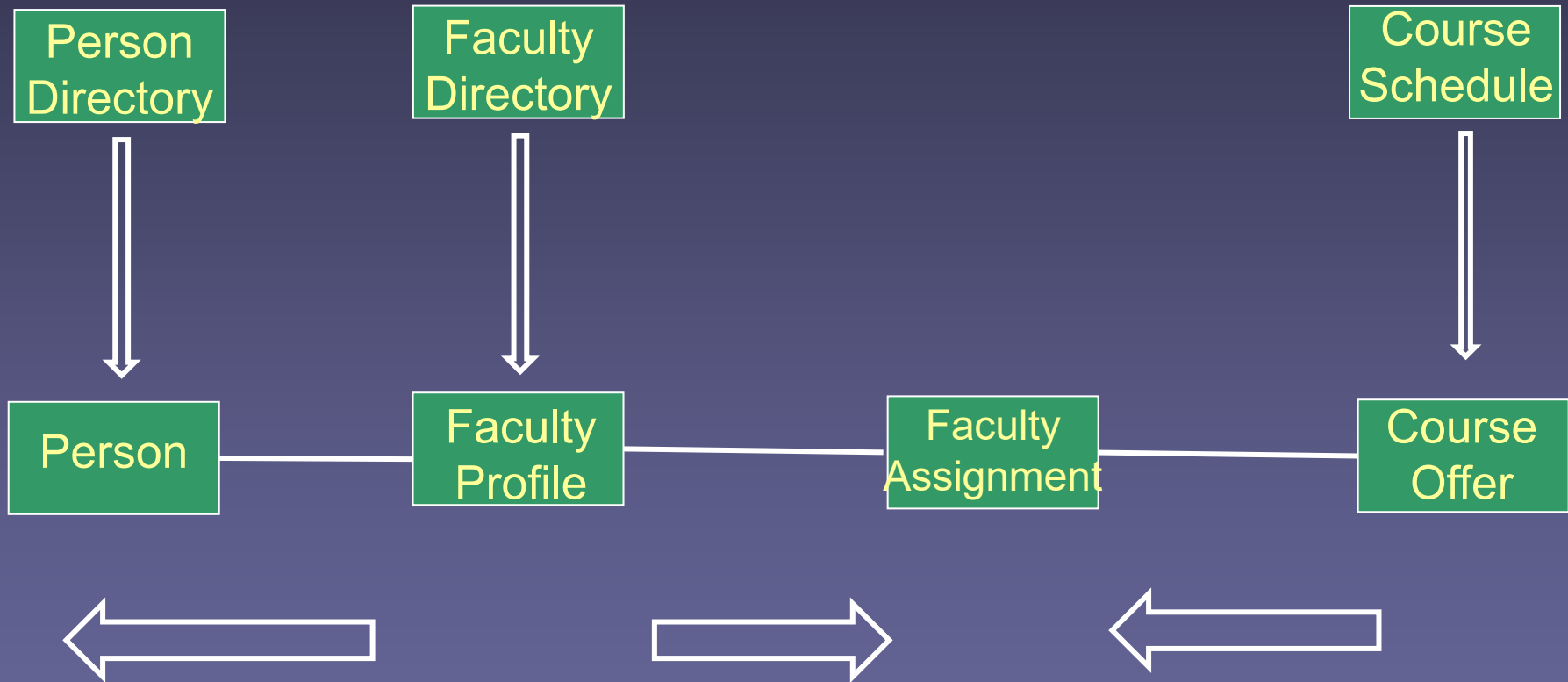
How to assign a teacher to a class



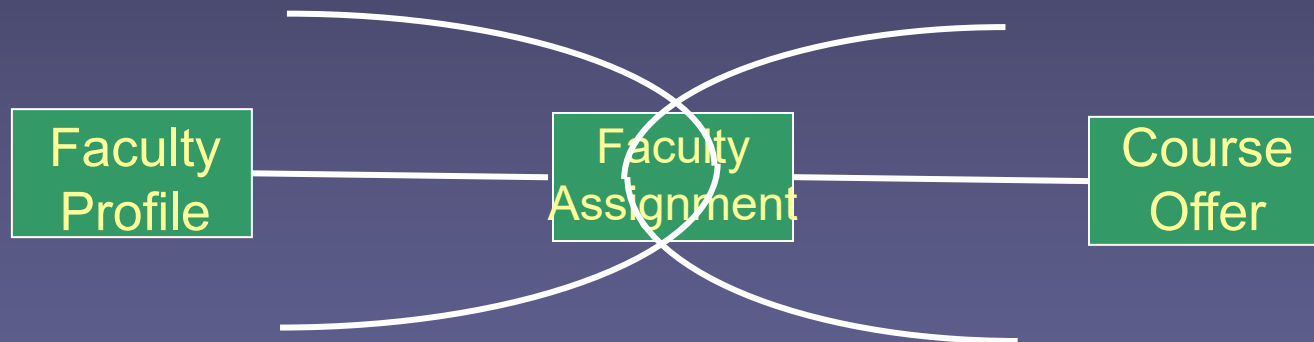
How to assign a teacher to a class



How to assign a teacher to a class

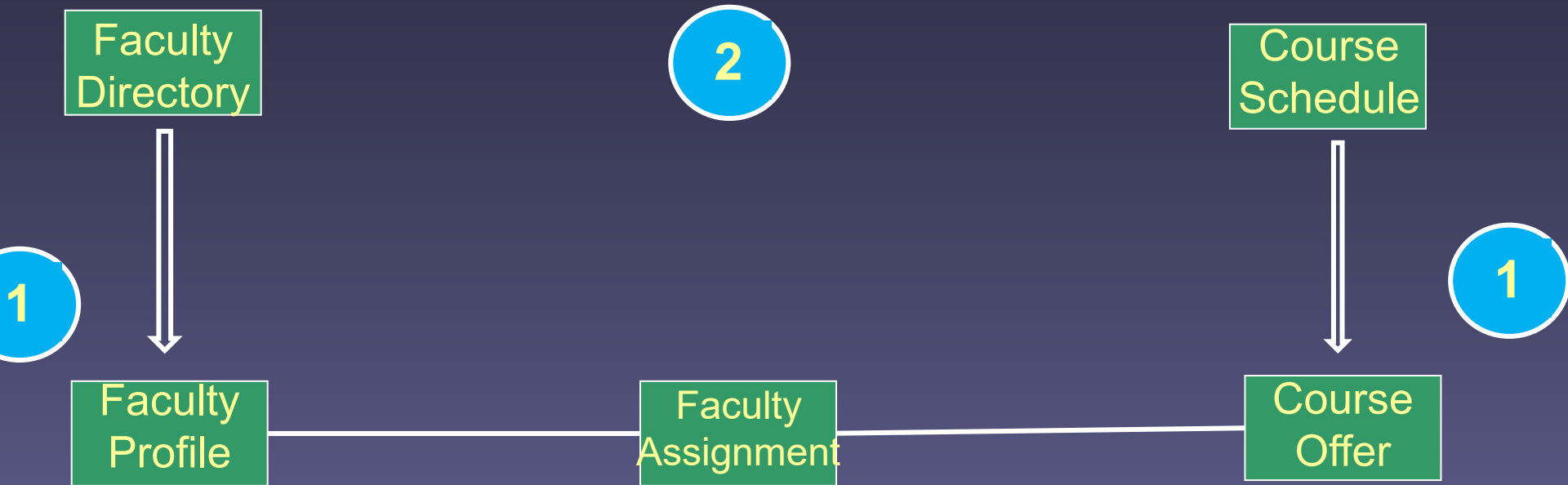


The binding object is shared by both. The faculty assignment knows the teacher and the course



```
public class FacultyAssignment {  
  
    CourseOffer courseoffer;  
    FacultyProfile facultyprofile;  
    public FacultyAssignment(FacultyProfile fp, CourseOffer co){  
        courseoffer = co;  
        facultyprofile = fp;  
    }  
}
```

the teacher for the class



Find teacher => faculty profile

Find course offer => course offer

From Course offer assign teacher "course offer has faculty assignment attribute

Courseoffer.assignteacher(teacher profile)

Faculty
Directory

Course
Schedule

1

2

1

Faculty
Profile

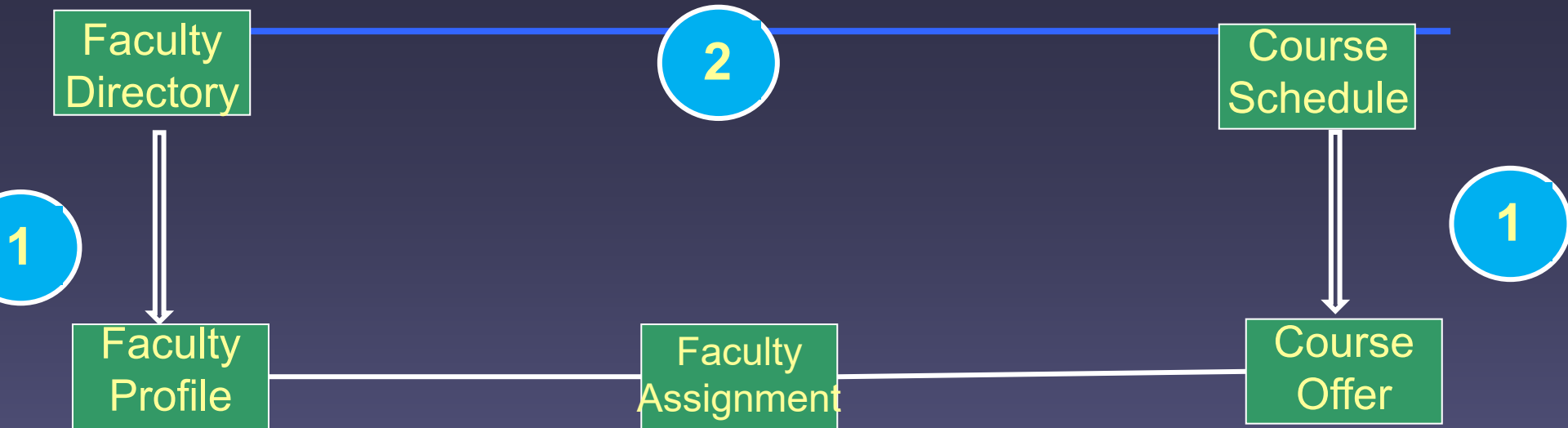
Faculty
Assignment

Course
Offer

- ▼ Persona
 - ▼ EmploymentHistory
 - Employment.java
 - EmploymentHistroy.java
 - ▼ Faculty
 - FacultyAssignment.java
 - FacultyDirectory.java
 - FacultyProfile.java
 - Person.java
 - PersonDirectory.java
 - StudentAccount.java
 - StudentDirectory.java
 - StudentProfile.java
 - Transcript.java



```
public class FacultyProfile {  
    Person person;  
    ArrayList <FacultyAssignment> facultyassignments;  
  
    public FacultyProfile(Person p) {  
        person = p;  
        facultyassignments = new ArrayList();  
    }  
  
    public FacultyAssignment AssignAsTeacher(CourseOffer co) {  
        FacultyAssignment fa = new FacultyAssignment(this, co);  
        facultyassignments.add(fa);  
        return fa;  
    }  
}
```



```
public class CourseOffer {

    Course course;
    ArrayList<Seat> seatlist;
    FacultyAssignment facultyassignment;

    public CourseOffer(Course c) {
        course = c;
        seatlist = new ArrayList();
    }
    public void AssignAsTeacher(FacultyProfile fp) {

        facultyassignment = new FacultyAssignment(fp, this);
    }

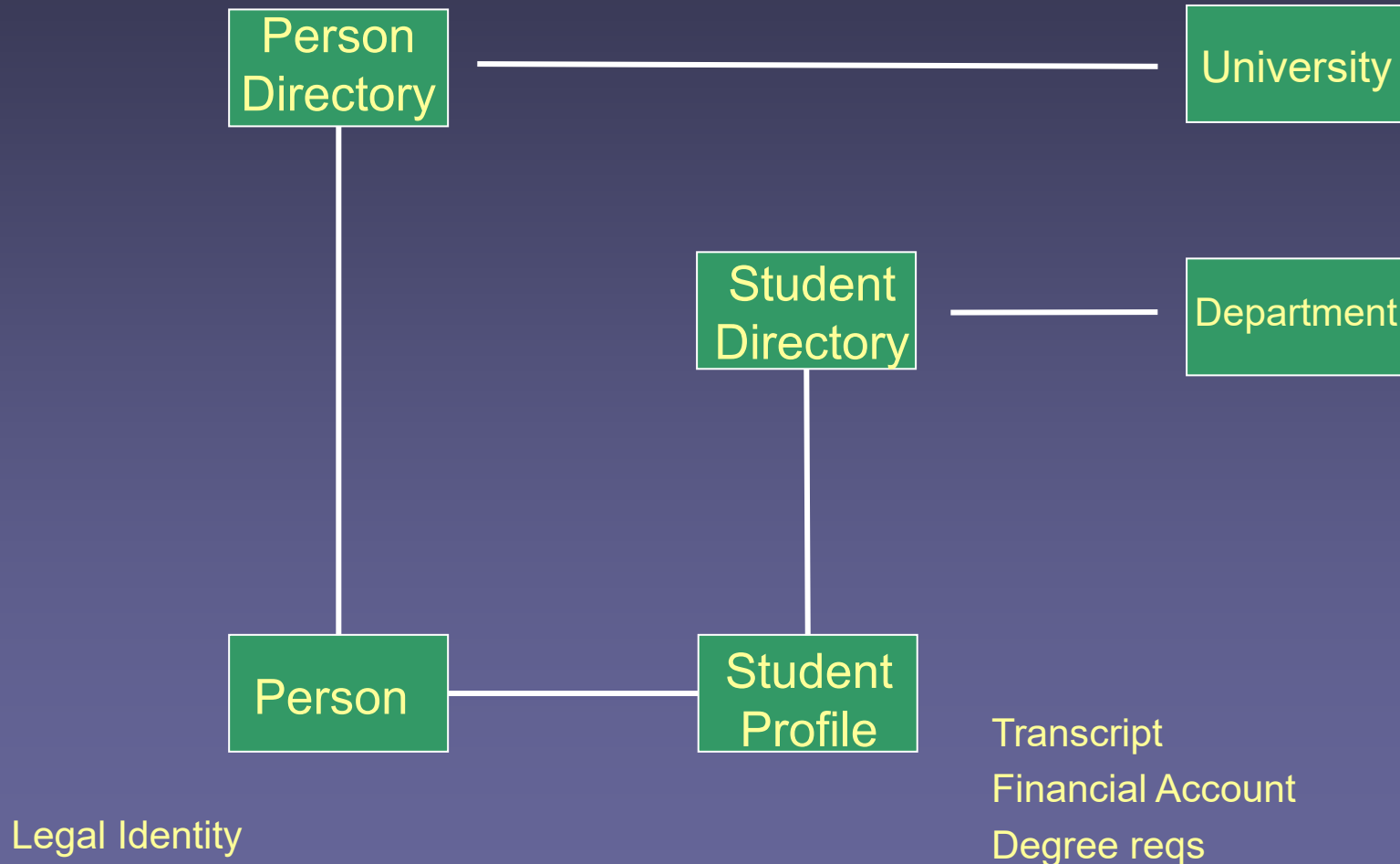
    public FacultyProfile getFacultyProfile() {
        return facultyassignment.getFacultyProfile();
    }
}
```



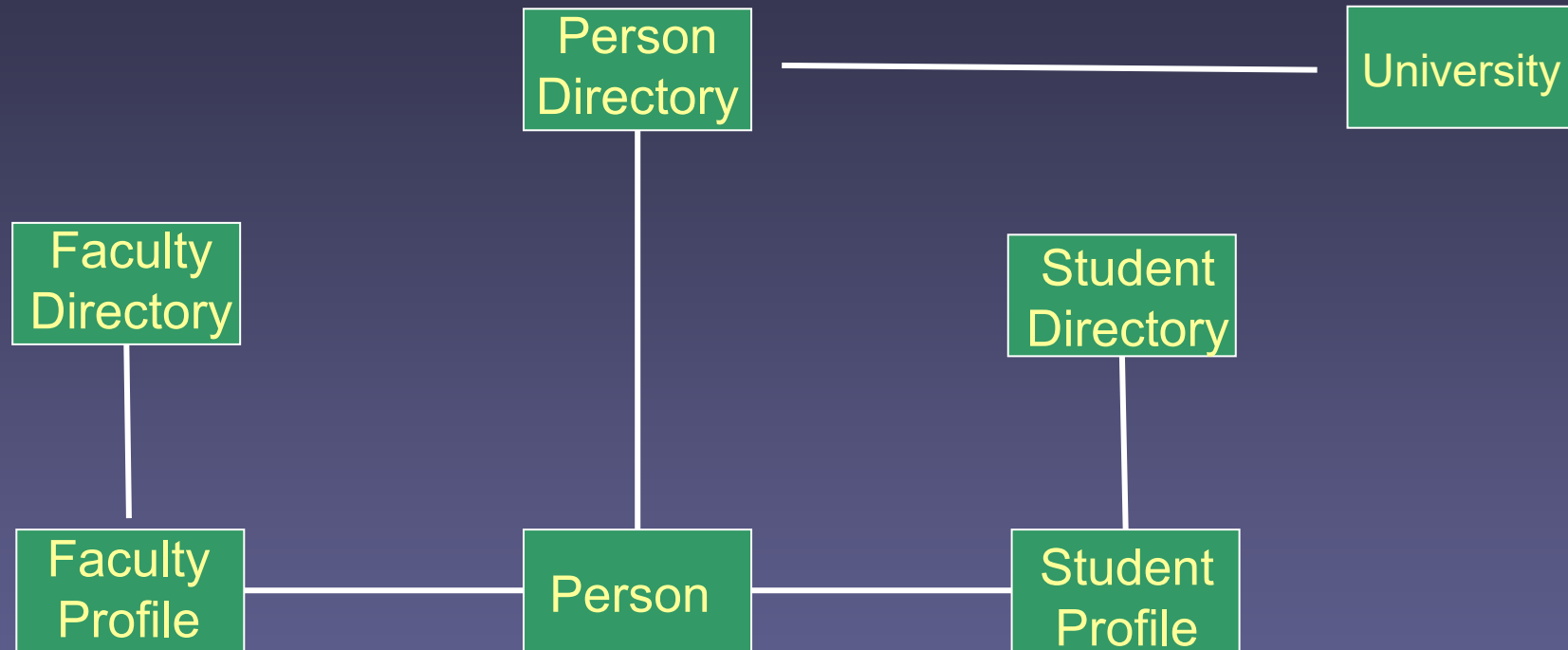
```
*/
public class FacultyAssignment {

    CourseOffer courseoffer;
    FacultyProfile facultyprofile;
    public FacultyAssignment(FacultyProfile fp, CourseOffer co){
        courseoffer = co;
        facultyprofile = fp;
    }
    public FacultyProfile getFacultyProfile(){
        return facultyprofile;
    }
}
```

How to model a person role – step 1



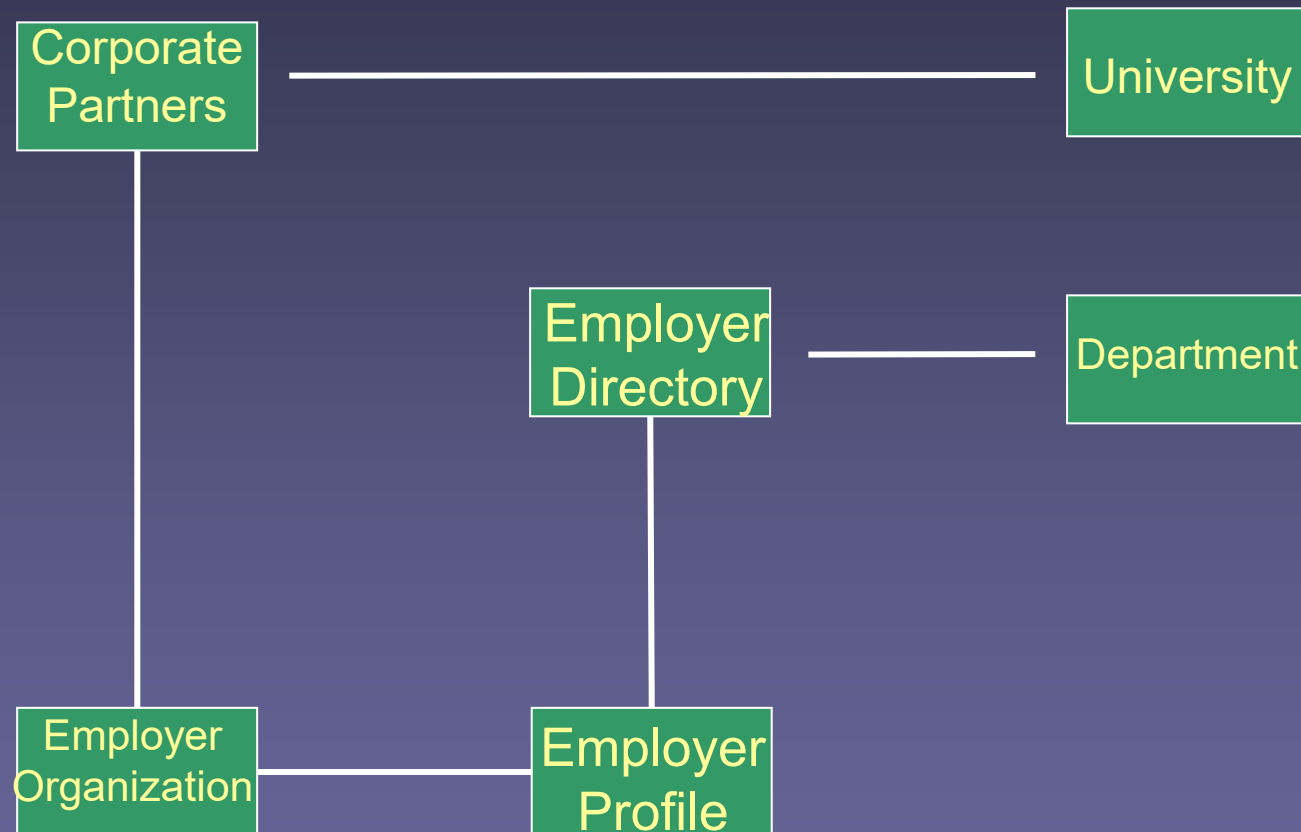
How to model a person role – step 1



Department
Classes
Performance evaluations

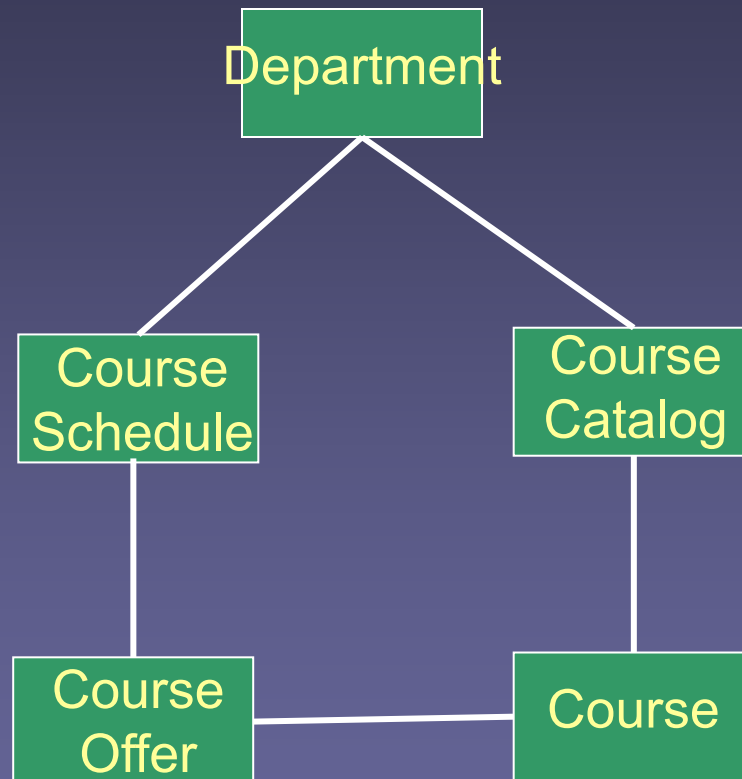
Transcript
Financial Account
Degree reqs

How to model a person role – step 1



Legal Identity

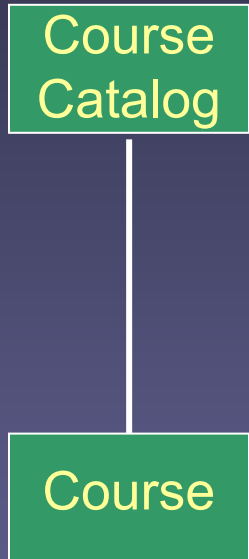
*Goal is to define the java code for
this component of the department*



Course is the first building block

Course

```
public class Course {  
    String number;  
    String name;  
    int credits;  
    int price = 1250; //per credit hour  
    public Course(String n, String numb, int ch, int p){  
        name = n;  
        number = numb;  
        credits = ch;  
        price = p;  
    }  
    public String getCOurseNumber(){  
        return number;  
    }  
  
    public int getCoursePrice() {  
        return price*credits;  
    }  
}
```



```
public class CourseCatalog {

    String lastupdated;
    ArrayList<Course> courselist;
    public CourseCatalog(){
        courselist = new ArrayList();
    }

    public ArrayList<Course> getCourseList(){
        return courselist;
    }
    public Course newCourse(String n, String nm, int cr){
        Course c = new Course(n, nm, cr);
        courselist.add(c);
        lastupdated = todaysdate();
        return c;
    }
    public Course getCourseByNumber(String n){
        for( Course c: courselist){

            if(c.getCourseNumber().equals(n)) return c;
        }
        return null;
    }
}
```

Order of generating the data



Every Semester there is a schedule

Course offerings

0

Course	Fall 2020	Spring 2021	Summer 2021	Fall 2021
Info 5001				
Info 5100				
Info 6210				
Info 6101				

```
HashMap<String, CourseSchedule> mastercoursecatalog;
```

```
public Department(String n) {  
    name = n;  
    mastercoursecatalog = new HashMap<String, CourseSchedule>();  
}
```

```
public class CourseOffer {
```

```
    Course course;  
    ArrayList<Seat> seatlist;
```

```
    public CourseOffer(Course c){  
        course =c;  
        seatlist = new ArrayList();
```

```
    }  
    public String getCourseNumber(){  
        return course.getCourseNumber();  
    }
```

```
    public void generatSeats(int n){  
        for (int i=0; i<n; i++){  
            seatlist.add(new Seat(this, i));  
        }  
    }
```

```
    public Seat getEmptySeat(){  
        for(Seat s: seatlist){  
  
            if(s.isOccupied()) return s;  
        }  
        return null;  
    }
```

```
    public int getTotalCourseRevenues(){  
        int sum = 0;  
        for(Seat s: seatlist){  
            if(s.isOccupied()==true) sum = sum + course.getCoursePrice();  
        }  
        return sum;  
    }  
}
```

Course
Offer

Seat


```

public class CourseOffer {

    Course course;
    ArrayList<Seat> seatlist;

    public CourseOffer(Course c){
        course = c;
        seatlist = new ArrayList();
    }
    public String getCourseNumber(){
        return course.getCourseNumber();
    }
    public void generatSeats(int n){
        for (int i=0; i<n; i++){
            seatlist.add(new Seat(this, i));
        }
    }
    public Seat getEmptySeat(){
        for(Seat s: seatlist){

            if(s.isOccupied()) return s;
        }
        return null;
    }
    public int getTotalCourseRevenues(){
        int sum = 0;
        for(Seat s: seatlist){
            if(s.isOccupied()==true) sum = sum + course.getCoursePrice();
        }
        return sum;
    }
}

```

Course
Offer

Seat



Seats

This means Current courseoffer

Course

Course
Offer

Seat

Course
Offer

Course

Course
Offer

Seat

```
public class CourseOffer {  
  
    Course course;  
    ArrayList<Seat> seatlist;  
  
    public CourseOffer(Course c){  
        course =c;  
        seatlist = new ArrayList();  
    }  
    public String getCourseNumber(){  
        return course.getCourseNumber();  
    }  
    public void generatSeats(int n){  
        for (int i=0; i<n; i++){  
            seatlist.add(new Seat(this, i));  
        }  
    }  
    public Seat getEmptySeat(){  
        for(Seat s: seatlist){  
  
            if(s.isOccupied()) return s;  
        }  
        return null;  
    }  
    public int getTotalCourseRevenues(){  
        int sum = 0;  
        for(Seat s: seatlist){  
            if(s.isOccupied()==true) sum = sum + course.getCoursePrice();  
        }  
        return sum;  
    }  
}
```