Computer Science 22: Object Oriented Programming

Lecture #3: Objects and Classes

About This Lecture

- Define: Object
  - Attributes
  - Behavior
  - State
  - Identity
- Define: Class

Object

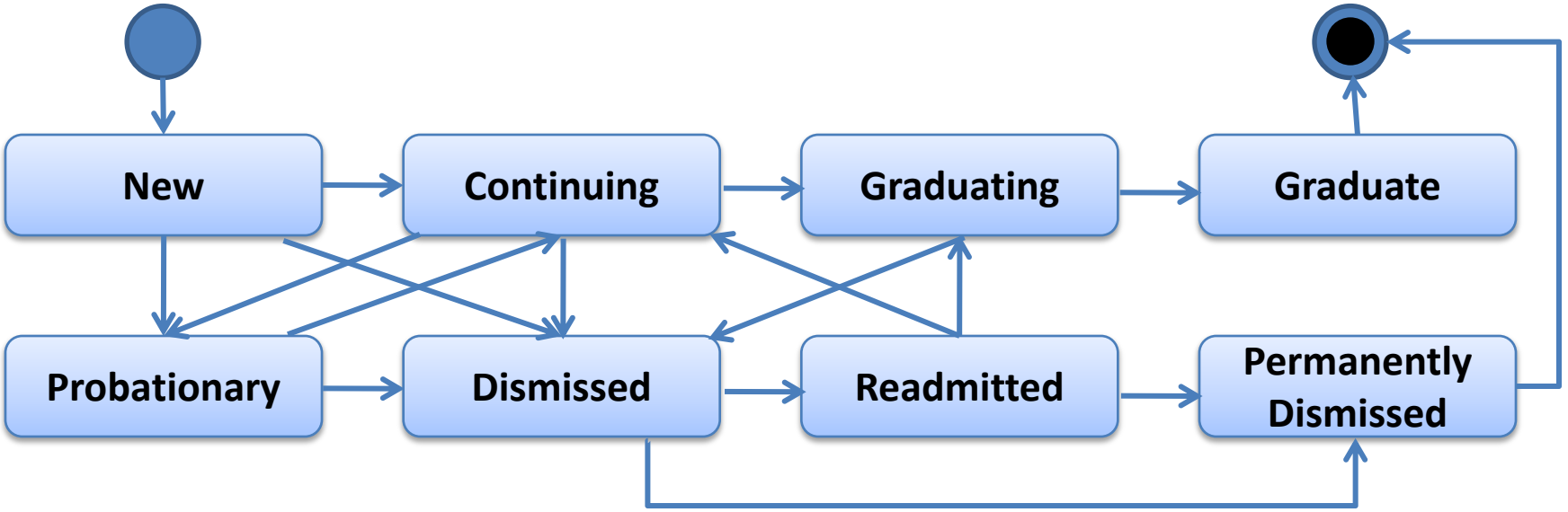
- An entity with well defined boundary that ***encapsulates*** **state** and **behavior**
- Any thing, **abstract** or real (**concrete**), with which we store data and the operations that manipulate those data.

Properties of Objects

- Objects have (maintain) **states**
- Objects exhibit **behavior**
- Objects have **identity**
- Objects have **relationships** with other objects

State

- The state of an object encompasses all the properties (static) of the object AND the current (dynamic) values of each of these properties.
- **Attributes** = properties



Example

- Object: Student
  - **Name**: “Juan Dela Cruz”
  - **Student Number**: “2008-12345”
  - **Birth Date**: January 1, 2000
  - **Gender**: Male
  - **Schools Attended**: {“Sakawalan Elementary School”, “Kungsansan High School”}
  - **Blood Type**: O

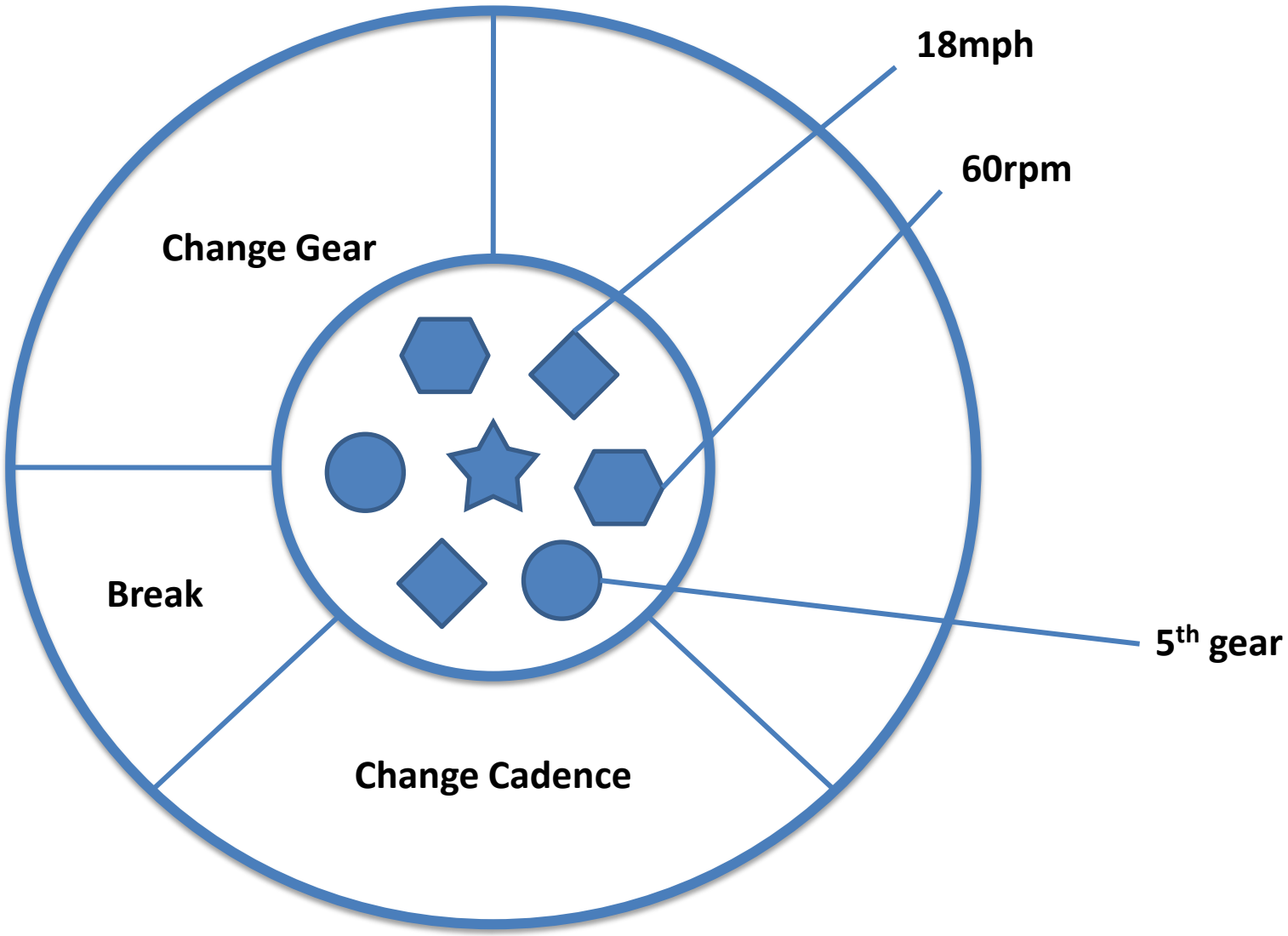
Behavior

- How an object acts and reacts, in terms of its state changes and message passing.
- The state of an object represents the cumulative results of its behavior.
- Defined actions of objects are called: **operations**
  - a.k.a. **Methods**
  - a.k.a. **Message** (i.e., defined messages that objects respond to or understand)

Example

- Operations/Methods on a radio:
  - Turn On/Turn Off
  - Select Modulation (AM/FM)
  - Change Frequency
  - ???

Object



Abstraction

- An abstraction denotes the essential characteristics of an object that distinguish it from all other kinds of objects and thus provide a crisply defined conceptual boundaries, relative to the perspective of the viewer.
- A simplified description, or specification of a system that emphasizes some of the system's details while suppressing the others.

Example:

- Key Abstractions in a Student Registration System

Student

A person who enlists in sections of courses

Recommended Courses

List of allowed courses for a student

Course

A college degree program

Section

A grouping composed of students who are taking a course scheduled at certain days and time of day

Slot

Represents one seat in section

Example:

- Key Abstractions for Library Computer/ Internet Usage Tracking

Student

A user who is given 20 hours worth of Internet access at the library

User Activity

Represents an instance where the user is logged into the internet

Ban List

List of students who committed “abuses”

Abstraction

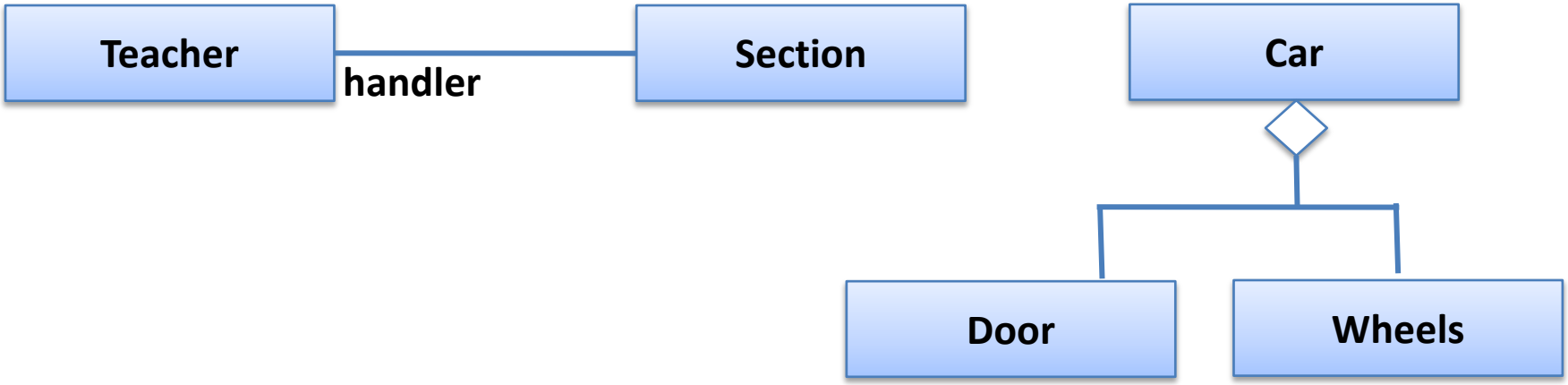
- Helps us deal with complexity
- Provides a basis for deciding which properties of objects are important and which are not given the problem context.
  - Compare Student in the two systems described
    - What pertinent properties and operations “define” the object in each context?

Back to Objects: Identity

- Identity is the property of an object which distinguishes it from all other objects
  - *More of this later on*

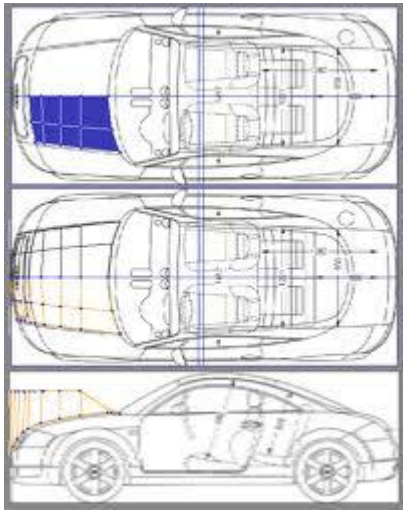
Relationships

- Kinds of relations
  - Association (Link)
  - Aggregation
  - Superclass/Subclass relationship



The Class

- A class is a set of objects that share a common **structure** and a common **behavior**.
- A single object is an **instance** of a class
- Analogy: class = template/blueprint



The Class

- Defines what properties and operations of objects that belong to it
  - e.g., All car instances have a steering wheel
  - e.g., All car instances can change their gears
- Objects in a class can be arranged in a hierarchy
  - e.g., Training cars are special kind of car (it has two steering wheels)
    - *More on this later when we discuss "Inheritance"*

Code Feature: C++ Class

```
class Account {
   private:
       int account_number;
       string fname, lname;
       float balance;
   public:
       Account( ) { }
       ~Account( ) { }
       void process { }
       int update(float balance);
       int liquidate();
}
```

Code Feature: Java

```
public class Student{
   private String studentNumber;
   private String firstName, lastName;
   public String allowedUnits;
   public Student () { }
   public void setAllowedUnits(int
   allowedUnits){
   }
   public void getAllowedUnits(){
   }
}
```

Reading Assignment

- What are the “parts” of a class?