High Level Design

High Level Design

- Moving from models of the real world toward software blueprints.
- Not all classes in the solution model correspond to "things" in the real world.
- Kinds of Classes
  - Entity
  - Boundary
  - Control

Kinds of Classes and Objects

- Entity Classes
  - Persistent information tracked by the system.
    - Caution: This term is used in several different ways in the software development literature.
  - Usually correspond to classes in our real world models.
- Boundary Classes
  - Responsible for interactions between the actors and the system.
    - User Interface classes.
  - Not in real world models.
- Control Classes
  - Top level control for use cases.
    - "Glue" for object methods.
  - Not in real world models.

Example: New Venue

- See posted solution for Project 6.

- Boundary class
  - Venue_from_User class.
    - No constructor. The class is never instantiated.
    - Static method Get_Venue_from_User()
    - Prompts user for input of venue information.
    - Creates a Venue object.
    - Returns Venue* pointer to caller.

- Could have just been a method in main.cpp
  - Put into a class to make program easier to read, understand, and modify.
  - Avoid large complex files.

Static Methods

- A method declared as static is associated with the class as a whole rather than with a specific object.
  - Invoked using the class name.
  - Has no "this" pointer.
  - Cannot refer to nonstatic member variables.
  - Cannot call nonstatic methods.

Static Classes

- A *class* declared as static can have only static methods and variables.
  - No constructor
  - Doesn't represent any "thing".
  - Just a home for the static methods and variables.

Venue_from_User.h

```
#pragma once

#include "Venue.h"

static class Venue_from_User
{
public:
    static Venue* Get_Venue_from_User();

private:
    static void Add_Seat_Rows(Venue* venue);
    static void Add_Seating_Sections(Venue* venue);
};
```

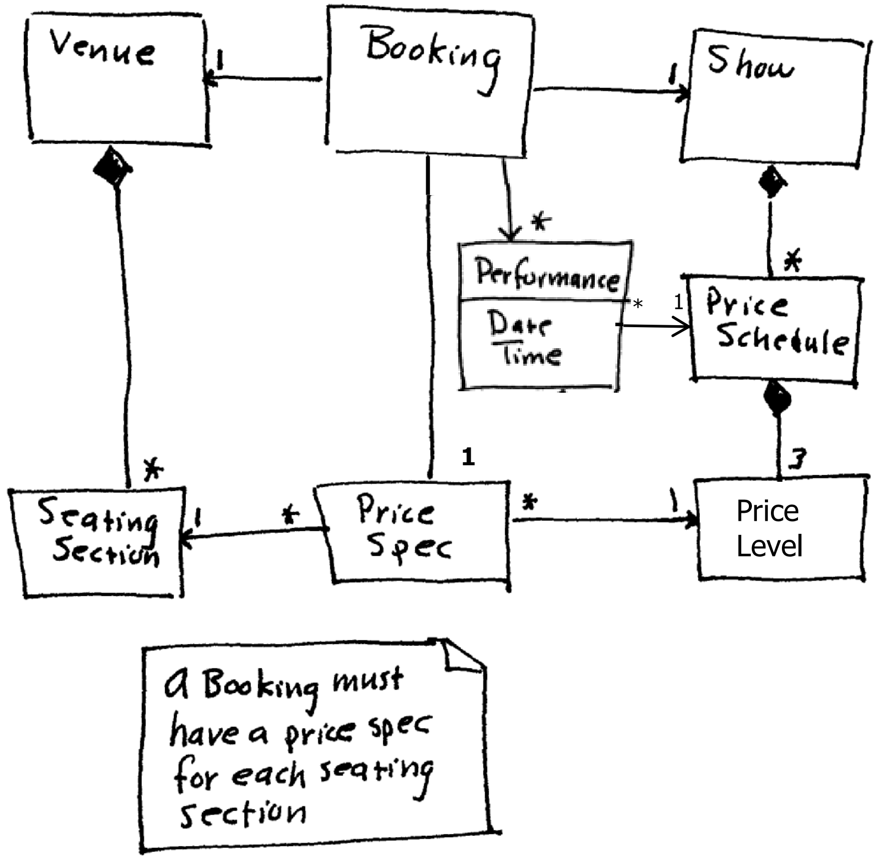
```
// Venue_from_User.cpp
include <iostream>
#include <string>
#include "Venue.h"
#include "Venue_from_User.h"
#include "Seating_Section.h"
using namespace std;

Venue* Venue_from_User::Get_Venue_from_User()
{
    string name;
    string street_address;
    string city;
    string state;
    int zip_code;

    // Get venue name and address from user
    ...
    Address adr(street_address, city, state, zip_code);
    Venue* new_venue = new Venue(name, adr);

    Add_Seat_Rows(new_venue);
    Add_Seating_Sections(new_venue);
    return new_venue;
}
```

Class Booking

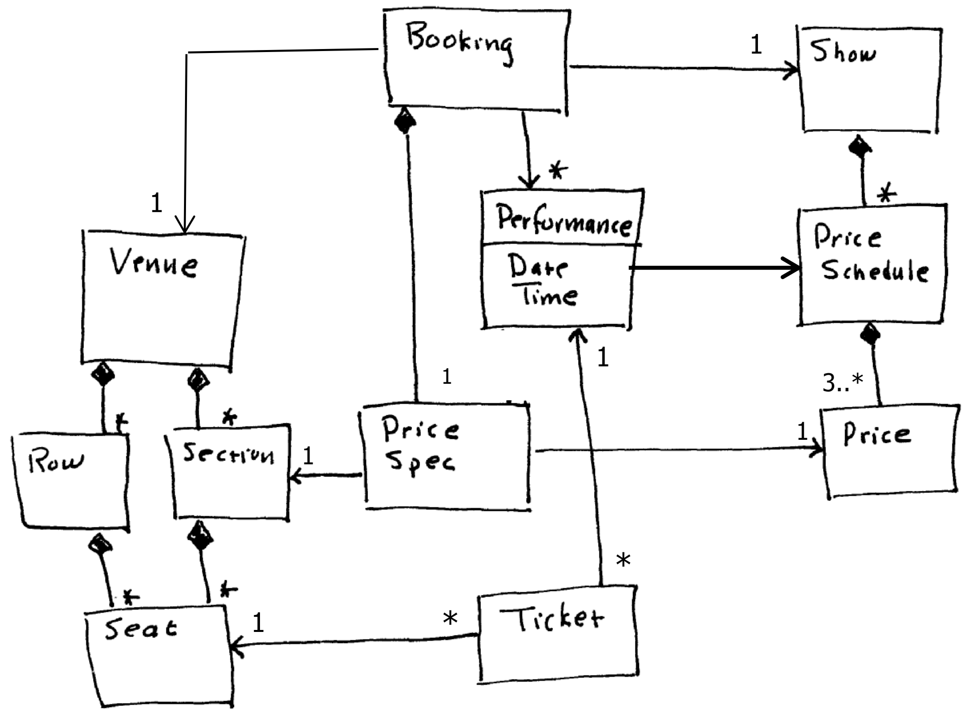
An administrator will book a show for some number of performances at a given venue.

- Specify venue.
- Specify show.
- Specify dates and times of performances.
- Specify a price schedule for each performance.
- Specify mapping of the venue's seating sections to the price levels in the show's price schedules.
    - Class Price_Specification
        - Applies to all performances

Class Booking



Ticket Booth Classes



Each Seat is
in one Row
and one section