Implementing Ticket Printer

Implementing Ticket Printer

- Download project from last class
- http://www.csee.usf.edu/~turnerr/Object_Oriented_Design/Downloads/2016_02_12_In_Class/
- File Ticket_Printer.zip
- Extract all
- Open project
- Build and run

Program Running

C:\Users\Rollins\Desktop\Ticket_Printer\Ticket_Printer\Debug\Ticket_Printer.exe

```
This is Ticket_Printer

19 Foster Street
Littleton, MA 01460

Row A Seat 1

Row Test has 4 seats
        Row Test Seat 1
        Row Test Seat 2
        Row Test Seat 3
        Row Test Seat 4

The Little Theater
19 Foster Street
Littleton, MA 01460
Row A has 4 seats
        Row A Seat 1
        Row A Seat 2
        Row A Seat 3
        Row A Seat 4
Row B has 4 seats
        Row B Seat 1
        Row B Seat 2
        Row B Seat 3
        Row B Seat 4
Row C has 4 seats
        Row C Seat 1
        Row C Seat 2
        Row C Seat 3
        Row C Seat 4
```

Continue Development

- Let's continue evolving this project into the real Ticket Printer program.

- We have our Venue.
- Before we can implement Ticket_Book we need class Performance

- Add class Performance to the project.

Performance.h

```
#pragma once
#include <string>
#include "Venue.h"

struct Date_Time
{
    int Day;
    int Month;
    int Year;

    int Hour;
    int Minute;
};
```

Performance.h (continued)

```
class Performance
{
private:
    const string show_name;
    const Venue* venue;
    const Date_Time when;

public:
    Performance(const string&     Show_Name,
                const Venue*      Venue_,
                const Date_Time&  When);

    const Venue* Get_Venue() const { return venue; };

    void Display() const;
};
```

Performance.cpp

```
#include <iostream>
#include <string.h>
#include "Performance.h"
#include "Venue.h"
using namespace std;

Performance::Performance(const string&      Show_Name,
                         const Venue*       Venue_,
                         const Date_Time&   When) :
    show_name(Show_Name), venue(Venue_), when(When)
{}
```

Performance.cpp (continued)

```
void Performance::Display() const
{
    cout.fill('0');
    cout << "Performance: " << show_name << endl;
    cout << when.Day << "/" << when.Month << "/" << when.Year;
    cout << " at ";
    cout.width(2);
    cout << when.Hour << ":";
    cout.width(2);
    cout << when.Minute << endl;
    venue->Display();
}
```

main.cpp

```
#include "Performance.h"
...

Performance* Create_Performance(Venue* venue)
{
    Date_Time when = {4, 2, 2016, 20, 0};
    Performance* p = new Performance("Billy Elliot", venue, when);
    return p;
}
...

    Performance* performance = Create_Performance(venue);
    performance->Display();
```

Program Running

C:\Users\Rollins\Desktop\Ticket_Printer\Ticket_Printer\Debug\Ticket_Printer.exe

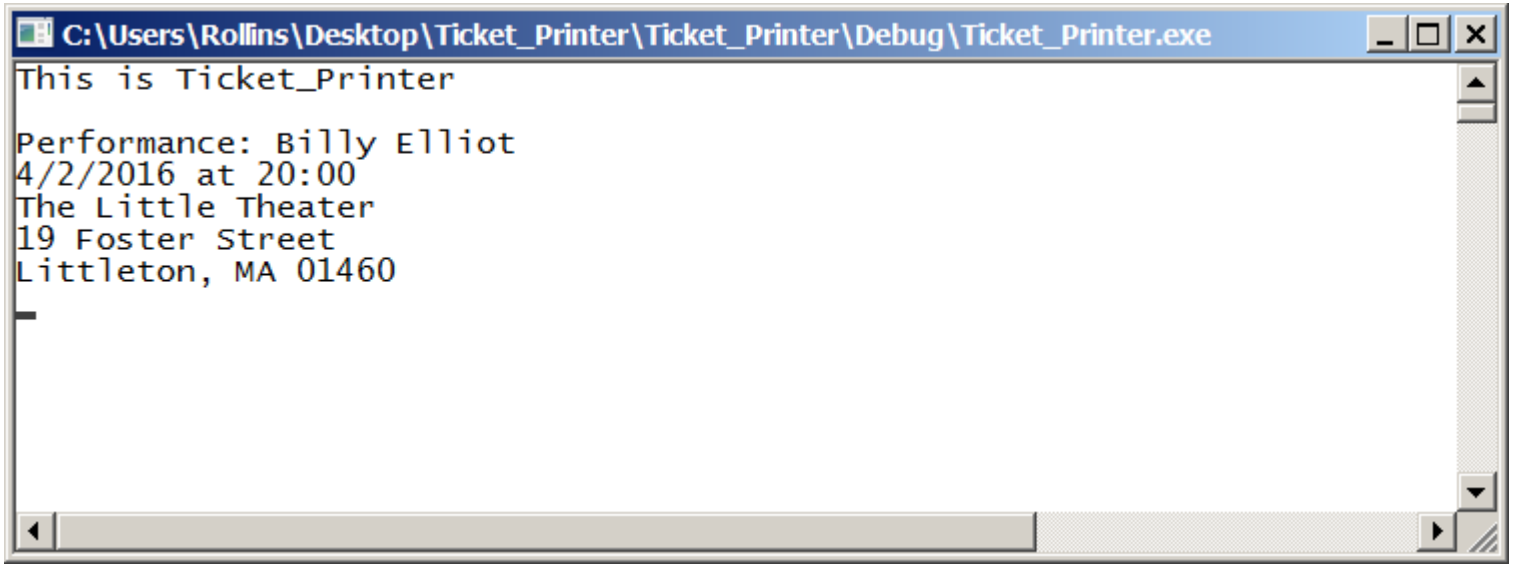
```
        Row Test Seat 3
        Row Test Seat 4

The Little Theater
19 Foster Street
Littleton, MA 01460
Row A has 4 seats
        Row A Seat 1
        Row A Seat 2
        Row A Seat 3
        Row A Seat 4
Row B has 4 seats
        Row B Seat 1
        Row B Seat 2
        Row B Seat 3
        Row B Seat 4
Row C has 4 seats
        Row C Seat 1
        Row C Seat 2
        Row C Seat 3
        Row C Seat 4
Performance: Billy Elliot
4/2/2016 at 20:00
The Little Theater
19 Foster Street
Littleton, MA 01460
Row A has 4 seats
        Row A Seat 1
        Row A Seat 2
        Row A Seat 3
        Row A Seat 4
Row B has 4 seats
        Row B Seat 1
        Row B Seat 2
        Row B Seat 3
        Row B Seat 4
Row C has 4 seats
        Row C Seat 1
        Row C Seat 2
        Row C Seat 3
        Row C Seat 4
```

Clean Up

- Comment out the earlier outputs in main.cpp
  - We know those classes are working now.
- When we print a ticket, we will use the Display method of Performance to output the name of the show and Venue
  - But we don't want to show all of the seats in the venue, just the name and address.
  - Change the Display method in class Venue to just output what we want.
- Build and run

Program Running



Looks like our Performance class is OK.

Continue Development

- We have all the classes now except Ticket_Book and Ticket.
- Ticket_Book depends on Ticket
  - "Has a" relationship
- Let's implement class Ticket next.
- Add class Ticket to the project.

Ticket.h

```
#pragma once
#include "Performance.h"
#include "Seat.h"

class Ticket
{
private:
    const Performance* performance;
    const Seat* seat;
    bool sold;

public:
    Ticket(const Performance* perf, const Seat* s);
    void Display() const;
};
```

Ticket.cpp

```
#include "Ticket.h"
#include "Performance.h"
#include "Seat.h"

Ticket::Ticket(const Performance* perf, const Seat* s) :
    performance(perf), seat(s), sold(false)
{}

void Ticket::Display() const
{
    performance->Display();
    seat->Display();
}
```

Class Ticket

- Class Ticket *delegates* its Display functionality to class Performance and class Seat.

Testing Class Ticket

main.cpp

```
#include "Ticket.h"
...
    //performance->Display();

    Seat* seat = new Seat("A", 1);
    Ticket* ticket = new Ticket(performance, seat);
    ticket->Display();
```

Build and run

Program Running

C:\Users\Rollins\Desktop\Ticket_Printer\Ticket_Printer\Debug\Ticket_Printer.exe

```
This is Ticket_Printer

Performance: Billy Elliot
4/2/2016 at 20:00
The Little Theater
19 Foster Street
Littleton, MA 01460
Row A Seat 1
```

Continue Development

We have all the classes now except Ticket_Book.

Add class Ticket_Book to the project.

Implementing Ticket_Book

- Our first task is to create the tickets.
  - Do this in the constructor.

- Given the Venue and the Performance
  - Iterate over the seat rows.
    - For each seat row
      - Iterate over the seats.
        - For each seat
          - Create a ticket

Ticket_Book.h

```
#pragma once
#include "Performance.h"
#include "Ticket.h"

class Ticket_Book
{
private:
    Ticket** tickets;   // A dynamically allocated array of
                        // pointers to Tickets
    int number_of_tickets;

public:
    Ticket_Book(Performance* p);

    void Print_Tickets() const;
};
```

Ticket_Book.cpp

```
#include <iostream>
#include "Ticket_Book.h"
#include "Ticket.h"
#include "Venue.h"
using namespace std;

Ticket_Book::Ticket_Book(Performance* p)
{
    const Venue* venue = p->Get_Venue();
    number_of_tickets = venue->Capacity();
    tickets = new Ticket*[number_of_tickets];

}
```

Build and run

Error!

Error List

Entire Solution | 2 Errors | 0 Warnings | 0 Messages | Build + IntelliSense

Search Error List

	Code	Description	Project	File	Line	Supp...
	LNK2019	unresolved external symbol "public: int __thiscall Venue::Capacity(void)const " (?Capacity@Venue@@QBEHXZ) referenced in function "public: __thiscall Ticket_Book::Ticket_Book(class Performance *)" (??0Ticket_Book@@QAE@PAVPerformance@@@Z)	Ticket_Printer	Ticket_Book.obj	1	
	LNK1120	1 unresolved externals	Ticket_Printer	Ticket_Printer.exe	1	

We have not implemented the Capacity method in class Venue.

Add to Venue.cpp

```
// Return number of seats
int Venue::Capacity() const
{
    int count = 0;
    for (int i = 0; i < number_of_seat_rows; ++i)
    {
        count += seat_rows[i]->Number_of_Seats();
    }
    return count;
}
```

Visual Studio tells us that Number_of_Seats does not exist.

Seat_Row.h

Add accessor method for number_of_seats

```
int Number_of_Seats() const {return number_of_seats;};
```

Build and run

Remember our objective

- Given the Venue and the Performance
  - Iterate over the seat rows.
    - For each seat row
      - Iterate over the seats.
        - For each seat
          - Create a ticket

Add to Ticket_Book.cpp

```
int n = 0;  // index for array tickets

// Create a ticket for each seat in the venue.
int nr_rows = venue->Number_of_Seat_Rows();
for (int i = 0; i < nr_rows; ++i)
{
    const Seat_Row* row = venue->Get_Seat_Row(i);
    int nr_seats_in_row = row->Number_of_Seats();
    for (int j = 0; j < nr_seats_in_row; ++j)
    {
        const Seat* seat = row->Get_Seat(j);
        tickets[n++] = new Ticket(p, seat);
    }
}
```

Missing Accessor Methods

- Visual Studio tells us that some accessor methods that we need are missing:
  - `venue->Number_of_Seat_Rows();`
  - `venue->Get_Seat_Row(i);`
  - `row->Get_Seat(j);`

Add to Venue.h

```
int Number_of_Seat_Rows() const
{
    return number_of_seat_rows;
};

const Seat_Row* Get_Seat_Row(int index) const
{
    return seat_rows[index];
}
```

Add to Seat_Row.h

```
const Seat* Get_Seat(int idx) const { return seats[idx]; };
```

Build and run

Implementing Print_Tickets

- We have created an array of Tickets.
- Iterate over the array and ask each Ticket to print itself.

Add to Ticket_Book.cpp

```
void Ticket_Book::Print_Tickets() const
{
    for (int i = 0; i < number_of_tickets; ++i)
    {
        tickets[i]->Display();
        cout << "----------------------------\n";
    }
}
```

Testing Ticket_Book

```
#include "Ticket_Book.h"
...
    //Seat* seat = new Seat("A", 1);
    //Ticket* ticket = new Ticket(performance, seat);
    //ticket->Display();

    Ticket_Book* ticket_book = new Ticket_Book(performance);
    ticket_book->Print_Tickets();
```

Build and run

Program Running

C:\Users\Rollins\Desktop\Ticket_Printer\Ticket_Printer\Debug\Ticket_Printer.exe

```
This is Ticket_Printer

Performance: Billy Elliot
4/2/2016 at 20:00
The Little Theater
19 Foster Street
Littleton, MA 01460
Row A Seat 1
----------------------------
Performance: Billy Elliot
4/2/2016 at 20:00
The Little Theater
19 Foster Street
Littleton, MA 01460
Row A Seat 2
----------------------------
Performance: Billy Elliot
4/2/2016 at 20:00
The Little Theater
19 Foster Street
Littleton, MA 01460
Row A Seat 3
----------------------------
Performance: Billy Elliot
4/2/2016 at 20:00
The Little Theater
19 Foster Street
Littleton, MA 01460
Row A Seat 4
----------------------------
Performance: Billy Elliot
4/2/2016 at 20:00
The Little Theater
19 Foster Street
Littleton, MA 01460
Row B Seat 1
----------------------------
```