

Theatre

Generated by Doxygen 1.9.8

1 Hierarchical Index	1
1.1 Class Hierarchy	1
2 Class Index	3
2.1 Class List	3
3 File Index	5
3.1 File List	5
4 Class Documentation	7
4.1 Actor Class Reference	7
4.1.1 Member Function Documentation	8
4.1.1.1 GetName()	8
4.1.1.2 GetRole()	8
4.1.1.3 operator==()	8
4.1.1.4 Play()	9
4.2 BalletDancer Class Reference	9
4.2.1 Member Function Documentation	10
4.2.1.1 Play()	10
4.3 BalletTheatre Class Reference	10
4.3.1 Member Function Documentation	11
4.3.1.1 Introduction()	11
4.4 Date Class Reference	11
4.4.1 Member Function Documentation	12
4.4.1.1 GetDate()	12
4.4.1.2 operator==()	12
4.5 DollActor Class Reference	13
4.5.1 Member Function Documentation	13
4.5.1.1 Play()	13
4.6 DollTheatre Class Reference	14
4.6.1 Member Function Documentation	14
4.6.1.1 Introduction()	14
4.7 Musician Class Reference	15
4.7.1 Member Function Documentation	16
4.7.1.1 Play()	16
4.8 OperaTheatre Class Reference	16
4.8.1 Member Function Documentation	17
4.8.1.1 Introduction()	17
4.9 OrderService Class Reference	17
4.9.1 Member Function Documentation	18
4.9.1.1 OrderTicket()	18
4.10 Performance Class Reference	18
4.10.1 Member Function Documentation	18

4.10.1.1 AddActor()	18
4.10.1.2 CreatePlaces()	19
4.10.1.3 GetActorsCount()	19
4.10.1.4 GetDate()	19
4.10.1.5 GetName()	19
4.10.1.6 GetPlace()	20
4.10.1.7 operator==()	20
4.10.1.8 RemoveActor()	20
4.11 Place Class Reference	21
4.11.1 Member Function Documentation	21
4.11.1.1 GetPlace()	21
4.11.1.2 IsEmpty()	21
4.11.1.3 Take()	22
4.12 Theatre Class Reference	22
4.12.1 Member Function Documentation	23
4.12.1.1 GetName()	23
4.12.1.2 Introduction()	23
4.12.1.3 StartPerformance()	23
4.13 Ticket Class Reference	24
4.13.1 Member Function Documentation	24
4.13.1.1 GetPerformance()	24
5 File Documentation	25
5.1 Actor.h	25
5.2 BalletDancer.h	25
5.3 BalletTheatre.h	25
5.4 Date.h	26
5.5 DollActor.h	26
5.6 DollTheatre.h	26
5.7 Musician.h	26
5.8 Opera.h	26
5.9 OrderService.h	27
5.10 pch.h	27
5.11 Performance.h	27
5.12 Place.h	28
5.13 Theatre.h	28
5.14 Ticket.h	29
Index	31

Chapter 1

Hierarchical Index

1.1 Class Hierarchy

This inheritance list is sorted roughly, but not completely, alphabetically:

Actor	7
BalletDancer	9
DollActor	13
Musician	15
Date	11
OrderService	17
Performance	18
Place	21
Theatre	22
BalletTheatre	10
DollTheatre	14
OperaTheatre	16
Ticket	24

Chapter 2

Class Index

2.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

Actor	7
BalletDancer	9
BalletTheatre	10
Date	11
DollActor	13
DollTheatre	14
Musician	15
OperaTheatre	16
OrderService	17
Performance	18
Place	21
Theatre	22
Ticket	24

Chapter 3

File Index

3.1 File List

Here is a list of all documented files with brief descriptions:

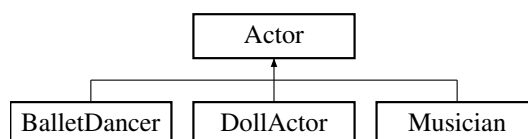
TheatreTest/ Actor.h	25
TheatreTest/ BalletDancer.h	25
TheatreTest/ BalletTheatre.h	25
TheatreTest/ Date.h	26
TheatreTest/ DollActor.h	26
TheatreTest/ DollTheatre.h	26
TheatreTest/ Musician.h	26
TheatreTest/ Opera.h	26
TheatreTest/ OrderService.h	27
TheatreTest/ pch.h	27
TheatreTest/ Performance.h	27
TheatreTest/ Place.h	28
TheatreTest/ Theatre.h	28
TheatreTest/ Ticket.h	29

Chapter 4

Class Documentation

4.1 Actor Class Reference

Inheritance diagram for Actor:



Public Member Functions

- **Actor** (string name, string role)
 - string [GetName](#) ()
Implementation of the [Actor](#) class.
- string [GetRole](#) ()
Getter of Role.
- bool [operator==](#) ([Actor](#) actor)
Overloading == operator.
- virtual void [Play](#) ()
Play actor role.

Protected Attributes

- string **name**
- string **role**

4.1.1 Member Function Documentation

4.1.1.1 GetName()

```
string Actor::GetName ( )
```

Implementation of the [Actor](#) class.

Author

Dasha

Version

1

Date

November 2023

Contains a constructor; method for playing. Operator == for comparing the fields of two actors.

Getter of Name

Returns

Name

4.1.1.2 GetRole()

```
string Actor::GetRole ( )
```

Getter of Role.

Returns

Role

4.1.1.3 operator==()

```
bool Actor::operator== (
    Actor actor )
```

Overloading == operator.

Parameters

<i>actor</i>	other actor
--------------	-------------

Returns

True if yes, False if no

4.1.1.4 Play()

```
void Actor::Play ( ) [virtual]
```

Play actor role.

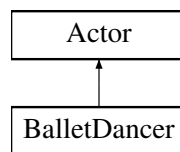
Reimplemented in [BalletDancer](#), [DollActor](#), and [Musician](#).

The documentation for this class was generated from the following files:

- TheatreTest/Actor.h
- TheatreTest/Actor.cpp

4.2 BalletDancer Class Reference

Inheritance diagram for BalletDancer:



Public Member Functions

- **BalletDancer** (string name, string role)
- void [Play](#) () override
Implementation of the [BalletDancer](#) class.

Public Member Functions inherited from [Actor](#)

- **Actor** (string name, string role)
- string [GetName](#) ()
Implementation of the [Actor](#) class.
- string [GetRole](#) ()
Getter of Role.
- bool [operator==](#) ([Actor](#) actor)
Overloading == operator.

Additional Inherited Members

Protected Attributes inherited from [Actor](#)

- string **name**
- string **role**

4.2.1 Member Function Documentation

4.2.1.1 Play()

```
void BalletDancer::Play ( ) [override], [virtual]
```

Implementation of the [BalletDancer](#) class.

Author

Dasha

Version

1

Date

November 2023

Contains a constructor; method for playing.

Play balletdancer role

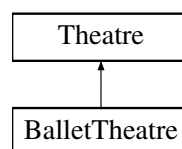
Reimplemented from [Actor](#).

The documentation for this class was generated from the following files:

- TheatreTest/BalletDancer.h
- TheatreTest/BalletDancer.cpp

4.3 BalletTheatre Class Reference

Inheritance diagram for BalletTheatre:



Public Member Functions

- **BalletTheatre** (string name, string adress, string director, vector< [Performance](#) > performances)
- void [Introduction](#) () override

Implementation of the [BalletTheatre](#) class.

Public Member Functions inherited from [Theatre](#)

- **Theatre** (string name, string address, string director, vector< [Performance](#) > performances)
- string [GetName](#) ()
Getter of Name.
- [Performance](#) **GetPerformance** (int i)
- void [StartPeroformance](#) ([Ticket](#) ticket)
Start of performance.
- void **ShowAfisha** ()
afisha output

Additional Inherited Members

4.3.1 Member Function Documentation

4.3.1.1 Introduction()

```
void BalletTheatre::Introduction ( ) [override], [virtual]
```

Implementation of the [BalletTheatre](#) class.

Author

Dasha

Version

1

Date

November 2023

Contains a constructor; method for start introdaction.

Start Introdaction

Reimplemented from [Theatre](#).

The documentation for this class was generated from the following files:

- TheatreTest/BalletTheatre.h
- TheatreTest/BalletTheatre.cpp

4.4 Date Class Reference

Public Member Functions

- **Date** (int hours, int minutes, int day, int month, int year)
- string [GetDate](#) ()
Implementation of the [Date](#) class.
- bool [operator==](#) ([Date](#) date)
Overloading == operator.

4.4.1 Member Function Documentation

4.4.1.1 `getDate()`

```
string Date::getDate ( )
```

Implementation of the [Date](#) class.

Author

Dasha

Version

1

Date

November 2023

Contains a constructor; method for showing information. Operator == for comparing the fields of two dates.

Getter of [Date](#)

Returns

date

4.4.1.2 `operator==()`

```
bool Date::operator== (
    Date date )
```

Overloading == operator.

Parameters

<i>date</i>	other date
-------------	------------

Returns

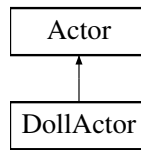
True if yes, False if no

The documentation for this class was generated from the following files:

- TheatreTest/Date.h
- TheatreTest/Date.cpp

4.5 DollActor Class Reference

Inheritance diagram for DollActor:



Public Member Functions

- **DollActor** (string name, string role)
- void **Play** () override
Play doll actor role.

Public Member Functions inherited from Actor

- **Actor** (string name, string role)
- string **GetName** ()
Implementation of the Actor class.
- string **GetRole** ()
Getter of Role.
- bool **operator==** (Actor actor)
Overloading == operator.

Additional Inherited Members

Protected Attributes inherited from Actor

- string **name**
- string **role**

4.5.1 Member Function Documentation

4.5.1.1 Play()

```
void DollActor::Play ( ) [override], [virtual]
```

Play doll actor role.

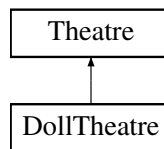
Reimplemented from [Actor](#).

The documentation for this class was generated from the following files:

- TheatreTest/DollActor.h
- TheatreTest/DollActor.cpp

4.6 DollTheatre Class Reference

Inheritance diagram for DollTheatre:



Public Member Functions

- **DollTheatre** (string name, string adress, string director, vector< [Performance](#) > performances)
- void [Introduction](#) () override
Implementation of the [DollTheatre](#) class.

Public Member Functions inherited from [Theatre](#)

- **Theatre** (string name, string address, string director, vector< [Performance](#) > performances)
- string [GetName](#) ()
Getter of Name.
- [Performance](#) [GetPerformance](#) (int i)
- void [StartPeroformance](#) ([Ticket](#) ticket)
Start of performance.
- void [ShowAfisha](#) ()
afisha output

Additional Inherited Members

4.6.1 Member Function Documentation

4.6.1.1 Introduction()

```
void DollTheatre::Introduction ( ) [override], [virtual]
```

Implementation of the [DollTheatre](#) class.

Author

Dasha

Version

1

Date

November 2023

Contains a constructor; method for start introdaction.

Start Introduction

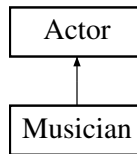
Reimplemented from [Theatre](#).

The documentation for this class was generated from the following files:

- TheatreTest/DollTheatre.h
- TheatreTest/DollTheatre.cpp

4.7 Musician Class Reference

Inheritance diagram for Musician:



Public Member Functions

- **Musician** (string name, string instrument)
- void **Play** () override

Implementation of the Musician class.

Public Member Functions inherited from **Actor**

- **Actor** (string name, string role)
- string **GetName** ()
Implementation of the Actor class.
- string **GetRole** ()
Getter of Role.
- bool **operator==** (**Actor** actor)
Overloading == operator.

Public Attributes

- string **instrument**

Additional Inherited Members

Protected Attributes inherited from **Actor**

- string **name**
- string **role**

4.7.1 Member Function Documentation

4.7.1.1 Play()

```
void Musician::Play ( ) [override], [virtual]
```

Implementation of the Musicion class.

Author

Dasha

Version

1

Date

November 2023

Contains a constructor; method for playing.

Play role

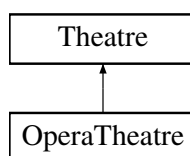
Reimplemented from [Actor](#).

The documentation for this class was generated from the following files:

- TheatreTest/Musician.h
- TheatreTest/Musician.cpp

4.8 OperaTheatre Class Reference

Inheritance diagram for OperaTheatre:



Public Member Functions

- **OperaTheatre** (string name, string adress, string director, vector< [Performance](#) > performances)
- void [Introduction](#) () override

Implementation of the Opera class.

Public Member Functions inherited from [Theatre](#)

- **Theatre** (string name, string address, string director, vector< [Performance](#) > performances)
- string [GetName](#) ()
Getter of Name.
- [Performance](#) **GetPerformance** (int i)
- void [StartPeroformance](#) ([Ticket](#) ticket)
Start of performance.
- void **ShowAfisha** ()
afisha output

Additional Inherited Members

4.8.1 Member Function Documentation

4.8.1.1 Introduction()

```
void OperaTheatre::Introduction ( ) [override], [virtual]
```

Implementation of the Opera class.

Author

Dasha

Version

1

Date

November 2023

Contains a constructor; method for start introdaction.

Start Introduction

Reimplemented from [Theatre](#).

The documentation for this class was generated from the following files:

- TheatreTest/Opera.h
- TheatreTest/Opera.cpp

4.9 OrderService Class Reference

Public Member Functions

- **OrderService** (vector< [Theatre](#) > theatres)
- [Ticket](#) **OrderTicket** ()
Order [Ticket](#) to visiting performance.

4.9.1 Member Function Documentation

4.9.1.1 OrderTicket()

```
Ticket OrderService::OrderTicket ( )
```

Order [Ticket](#) to visiting performance.

Returns

ticket

The documentation for this class was generated from the following files:

- TheatreTest/OrderService.h
- TheatreTest/OrderService.cpp

4.10 Performance Class Reference

Public Member Functions

- **Performance** (string name, [Date](#) date)
- string [GetName](#) ()
Implementation of the Performmance class.
- string [GetDate](#) ()
Getter of [Date](#).
- int [GetActorsCount](#) ()
Getter of Actors count.
- string [GetPlace](#) (int i)
Take place.
- void **ShowPlaces** ()
Show list avaible places.
- bool [operator==](#) ([Performance](#) performance)
Overloading == operator.
- void **Start** ()
Start performance.
- void [AddActor](#) ([Actor](#) actor)
Add new actor.
- void [RemoveActor](#) ([Actor](#) actor)
Remove actor.
- void [CreatePlaces](#) (int row, int place)
Initialize size of hall.

4.10.1 Member Function Documentation

4.10.1.1 AddActor()

```
void Performance::AddActor (
    Actor actor )
```

Add new actor.

Parameters

<i>actor</i>	new actor
--------------	-----------

4.10.1.2 CreatePlaces()

```
void Performance::CreatePlaces (
    int row,
    int place )
```

Initialize size of hall.

Parameters

<i>row,place</i>	size
------------------	------

4.10.1.3 GetActorsCount()

```
int Performance::GetActorsCount ( )
```

Getter of Actors count.

Returns

actors count

4.10.1.4 GetDate()

```
string Performance::GetDate ( )
```

Getter of [Date](#).

Returns

date

4.10.1.5 GetName()

```
string Performance::GetName ( )
```

Implementation of the Performmance class.

Author

Dasha

Version

1

Date

November 2023

Contains a constructor; method for showing information and starting performances.

Getter of Name

Returns

Name

4.10.1.6 GetPlace()

```
string Performance::GetPlace (
    int i )
```

Take place.

Returns

place

4.10.1.7 operator==()

```
bool Performance::operator== (
    Performance performance )
```

Overloading == operator.

Parameters

<i>performance</i>	other performance
--------------------	-------------------

Returns

True if yes, False if no

4.10.1.8 RemoveActor()

```
void Performance::RemoveActor (
    Actor actor )
```

Remove actor.

Parameters

<i>actor</i>	removable actor
--------------	-----------------

The documentation for this class was generated from the following files:

- TheatreTest/Performance.h
- TheatreTest/Performance.cpp

4.11 Place Class Reference

Public Member Functions

- **Place** (int row, int place)
- void **Take** ()
Implementation of the [Place](#) class.
- void **Cancel** ()
Cancel taking place.
- string **GetPlace** ()
Getter of [Place](#).
- bool **IsEmpty** ()
[Place](#) is empty or not.

4.11.1 Member Function Documentation

4.11.1.1 GetPlace()

```
string Place::GetPlace ( )
```

Getter of [Place](#).

Returns

place

4.11.1.2 IsEmpty()

```
bool Place::IsEmpty ( )
```

[Place](#) is empty or not.

Returns

True if yes, False if no

4.11.1.3 Take()

```
void Place::Take ( )
```

Implementation of the [Place](#) class.

Author

Dasha

Version

1

Date

November 2023

Contains a constructor; method for showing information and working with places.

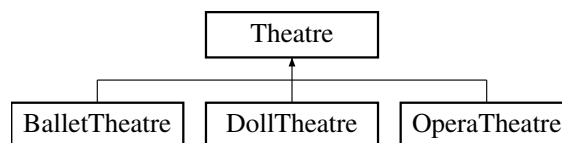
Take this place

The documentation for this class was generated from the following files:

- TheatreTest/Place.h
- TheatreTest/Place.cpp

4.12 Theatre Class Reference

Inheritance diagram for Theatre:



Public Member Functions

- **Theatre** (string name, string address, string director, vector< [Performance](#) > performances)
- string [GetName](#) ()
Getter of Name.
- [Performance](#) **GetPerformance** (int i)
- void [StartPeroformance](#) ([Ticket](#) ticket)
Start of performance.
- void **ShowAfisha** ()
afisha output

Protected Member Functions

- virtual void [Introduction](#) ()
Implementation of the Opera class.

4.12.1 Member Function Documentation

4.12.1.1 GetName()

```
string Theatre::GetName ( )
```

Getter of Name.

Returns

Name

4.12.1.2 Introduction()

```
void Theatre::Introduction ( ) [protected], [virtual]
```

Implementation of the Opera class.

Author

Dasha

Version

1

Date

November 2023

Contains a constructor; methods for start introduction and showing information.

Introduction output

Reimplemented in [BalletTheatre](#), [DollTheatre](#), and [OperaTheatre](#).

4.12.1.3 StartPerformance()

```
void Theatre::StartPerformance (
    Ticket ticket )
```

Start of performance.

Parameters

<i>ticket</i>	ticket of performance
---------------	-----------------------

The documentation for this class was generated from the following files:

- TheatreTest/Theatre.h
- TheatreTest/Theatre.cpp

4.13 Ticket Class Reference

Public Member Functions

- **Ticket** ([Performance](#) performance, string place, int price)
- [Performance](#) **GetPerformance** ()
Implementation of the [Ticket](#) class.
- void **ShowInfo** ()
Show information about performance.

4.13.1 Member Function Documentation

4.13.1.1 GetPerformance()

```
Performance Ticket::GetPerformance ( )
```

Implementation of the [Ticket](#) class.

Author

Dasha

Version

1

Date

November 2023

Contains a constructor; method for showing ticket information.

Getter of [Performance](#)

Returns

performance

The documentation for this class was generated from the following files:

- TheatreTest/Ticket.h
- TheatreTest/Ticket.cpp

Chapter 5

File Documentation

5.1 Actor.h

```
00001 #pragma once
00002
00003 #include <iostream>
00004
00005 using namespace std;
00006
00007 class Actor {
00008 protected:
00009     string name;
00010     string role;
00011 public:
00012     Actor(string name, string role) : name(name), role(role) {}
00013
00014     string GetName();
00015     string GetRole();
00016
00017
00018     bool operator ==(Actor actor);
00019
00020     void virtual Play();
00021 };
```

5.2 BalletDancer.h

```
00001 #pragma once
00002 #include "Actor.h"
00003
00004 class BalletDancer : public Actor {
00005 public:
00006
00007     BalletDancer(string name, string role) : Actor(name, role) {}
00008
00009     void Play() override;
00010 };
```

5.3 BalletTheatre.h

```
00001 #pragma once
00002 #include "Theatre.h"
00003
00004 class BalletTheatre : public Theatre {
00005 public:
00006     BalletTheatre(string name, string address, string director, vector<Performance> performances) :
00007         Theatre(name, address, director, performances) {}
00007
00008     void Introduction() override;
00009 };
```

5.4 Date.h

```
00001 #pragma once
00002 #include <string>
00003
00004 using namespace std;
00005
00006 class Date
00007 {
00008 private:
00009     int hours;
00010     int minutes;
00011     int day;
00012     int month;
00013     int year;
00014
00015 public:
00016     Date(int hours, int minutes, int day, int month, int year) : hours(hours), minutes(minutes),
        day(day), month(month), year(year) {}
00017
00018     string GetDate();
00019
00020     bool operator ==(Date date);
00021 };
```

5.5 DollActor.h

```
00001 #pragma once
00002 #include "Actor.h"
00003
00004 class DollActor : public Actor {
00005 public:
00006
00007     DollActor(string name, string role) : Actor(name, role) {}
00008
00009     void Play() override;
00010 };
```

5.6 DollTheatre.h

```
00001 #pragma once
00002 #include "Theatre.h"
00003
00004 class DollTheatre : public Theatre {
00005 public:
00006     DollTheatre(string name, string adress, string director, vector<Performance> performances) :
        Theatre(name, adress, director, performances) {}
00007
00008     void Introduction() override;
00009 };
```

5.7 Musician.h

```
00001 #pragma once
00002 #include "Actor.h"
00003
00004 class Musician : public Actor {
00005 public:
00006     string instrument;
00007
00008     Musician(string name, string instrument) : Actor(name, instrument), instrument(instrument) {}
00009
00010     void Play() override;
00011 };
```

5.8 Opera.h

```
00001 #pragma once
00002 #include "Theatre.h"
00003
```

```

00004 class OperaTheatre : public Theatre {
00005 public:
00006     OperaTheatre(string name, string address, string director, vector<Performance> performances) :
Theatre(name, address, director, performances) {}
00007
00008     void Introduction() override;
00009 };

```

5.9 OrderService.h

```

00001 #pragma once
00002 #include <vector>
00003 #include "Performance.h"
00004 #include "Ticket.h"
00005 #include "Theatre.h"
00006
00007 class OrderService {
00008 private:
00009     vector<Theatre> theatres;
00010     int ticketPrices[3] = { 5, 20, 15 };
00011
00012     void ShowTheatres();
00013
00014     void ShowPrice();
00015
00016     Theatre ChooseTheatre();
00017
00018     Performance ChoosePerformance(Theatre theatre);
00019
00020     string ChoosePlace(Performance performance);
00021
00022     int ChoosePrice();
00023
00024 public:
00025     OrderService(vector<Theatre> theatres) : theatres(theatres) {}
00026     Ticket OrderTicket();
00027 };

```

5.10 pch.h

```

00001 //
00002 // pch.h
00003 //
00004
00005 #pragma once
00006
00007 #include "gtest/gtest.h"
00008
00009 #include <iostream>
00010 #include <string>
00011
00012 using namespace std;
00013
00014
00015
00016
00017
00018
00019
00020
00021
00022
00023
00024
00025
00026

```

5.11 Performance.h

```

00001 #pragma once
00002 #include <string>
00003 #include "Date.h"
00004 #include <vector>
00005 #include "Place.h"
00006 #include "Actor.h"

```

```

00007
00008
00009 using namespace std;
00010
00011 class Performance {
00012 private:
00013     string name;
00014     Date date;
00015     vector<Actor> actors;
00016     vector<Place> places;
00017 public:
00018     Performance(string name, Date date) : name(name), date(date) {}
00019
00020     string GetName();
00021     string GetDate();
00022     int GetActorsCount();
00023     string GetPlace(int i);
00024
00025     void ShowPlaces();
00026
00027     bool operator ==(Performance performance);
00028
00029     void Start();
00030
00031     void AddActor(Actor actor);
00032
00033     void RemoveActor(Actor actor);
00034
00035     void CreatePlaces(int row, int place);
00036 };

```

5.12 Place.h

```

00001 #pragma once
00002 #include <string>
00003
00004 using namespace std;
00005
00006 class Place
00007 {
00008 private:
00009     int row;
00010     int place;
00011     bool isEmpty = true;
00012 public:
00013     Place(int row, int place) :row(row), place(place) {}
00014
00015     void Take();
00016     void Cancel();
00017
00018     string GetPlace();
00019
00020     bool IsEmpty();
00021 };

```

5.13 Theatre.h

```

00001 #pragma once
00002 #include <string>
00003 #include <vector>
00004 #include "Performance.h"
00005 #include "Ticket.h"
00006
00007 using namespace std;
00008
00009 class Theatre {
00010 private:
00011     string name;
00012     string address;
00013     string director;
00014     vector<Performance> performances;
00015
00016 protected:
00017     void virtual Introduction();
00018 public:
00019     Theatre(string name, string address, string director, vector<Performance> performances)
00020         :name(name), address(address), director(director), performances(performances) {}
00021
00022     string GetName();

```



```
00022     Performance GetPerformance(int i);
00023
00024     void StartPerformance(Ticket ticket);
00025
00026     void ShowAfisha();
00027 };
```

5.14 Ticket.h

```
00001 #pragma once
00002 #include <string>
00003 #include "Performance.h"
00004 #include "Place.h"
00005 using namespace std;
00006
00007 class Ticket {
00008 private:
00009     Performance performance;
00010     string place;
00011     double price;
00012
00013 public:
00014     Ticket(Performance performance, string place, int price) :performance(performance), place(place),
        price(price) {}
00015
00016     Performance GetPerformance();
00017
00018     void ShowInfo();
00019 };
```


Index

Actor, [7](#)
 GetName, [8](#)
 GetRole, [8](#)
 operator==, [8](#)
 Play, [9](#)
AddActor
 Performance, [18](#)

BalletDancer, [9](#)
 Play, [10](#)
BalletTheatre, [10](#)
 Introduction, [11](#)

CreatePlaces
 Performance, [19](#)

Date, [11](#)
 GetDate, [12](#)
 operator==, [12](#)
DollActor, [13](#)
 Play, [13](#)
DollTheatre, [14](#)
 Introduction, [14](#)

GetActorsCount
 Performance, [19](#)
GetDate
 Date, [12](#)
 Performance, [19](#)
GetName
 Actor, [8](#)
 Performance, [19](#)
 Theatre, [23](#)
GetPerformance
 Ticket, [24](#)
GetPlace
 Performance, [20](#)
 Place, [21](#)
GetRole
 Actor, [8](#)

Introduction
 BalletTheatre, [11](#)
 DollTheatre, [14](#)
 OperaTheatre, [16](#)
 Theatre, [23](#)
IsEmpty
 Place, [21](#)

Musician, [15](#)
 Play, [16](#)

OperaTheatre, [16](#)
 Introduction, [17](#)
operator==
 Actor, [8](#)
 Date, [12](#)
 Performance, [20](#)
OrderService, [17](#)
 OrderTicket, [18](#)
OrderTicket
 OrderService, [18](#)

Performance, [18](#)
 AddActor, [18](#)
 CreatePlaces, [19](#)
 GetActorsCount, [19](#)
 GetDate, [19](#)
 GetName, [19](#)
 GetPlace, [20](#)
 operator==, [20](#)
 RemoveActor, [20](#)
Place, [21](#)
 GetPlace, [21](#)
 IsEmpty, [21](#)
 Take, [21](#)
Play
 Actor, [9](#)
 BalletDancer, [10](#)
 DollActor, [13](#)
 Musician, [16](#)

RemoveActor
 Performance, [20](#)

StartPerformance
 Theatre, [23](#)

Take
 Place, [21](#)
Theatre, [22](#)
 GetName, [23](#)
 Introduction, [23](#)
 StartPerformance, [23](#)
TheatreTest/Actor.h, [25](#)
TheatreTest/BalletDancer.h, [25](#)
TheatreTest/BalletTheatre.h, [25](#)
TheatreTest/Date.h, [26](#)
TheatreTest/DollActor.h, [26](#)
TheatreTest/DollTheatre.h, [26](#)
TheatreTest/Musician.h, [26](#)
TheatreTest/Opera.h, [26](#)

TheatreTest/OrderService.h, [27](#)
TheatreTest/pch.h, [27](#)
TheatreTest/Performance.h, [27](#)
TheatreTest/Place.h, [28](#)
TheatreTest/Theatre.h, [28](#)
TheatreTest/Ticket.h, [29](#)
Ticket, [24](#)
 GetPerformance, [24](#)