

# TP - Sistema da Locadora

## 1.0

Generated by Doxygen 1.9.8



<b>1 Hierarchical Index</b>	<b>1</b>
1.1 Class Hierarchy	1
<b>2 Class Index</b>	<b>3</b>
2.1 Class List	3
<b>3 File Index</b>	<b>5</b>
3.1 File List	5
<b>4 Class Documentation</b>	<b>7</b>
4.1 Bluray Class Reference	7
4.1.1 Detailed Description	8
4.1.2 Constructor & Destructor Documentation	8
4.1.2.1 Bluray()	8
4.1.3 Member Function Documentation	9
4.1.3.1 alugarAparelho()	9
4.1.3.2 calcularValorLocacao()	9
4.1.3.3 devolverAparelho()	10
4.1.3.4 getQuantidadeAparelhosDisponiveis()	10
4.1.3.5 setQuantidadeAparelhos()	10
4.2 Cliente Class Reference	11
4.2.1 Detailed Description	11
4.2.2 Constructor & Destructor Documentation	12
4.2.2.1 Cliente()	12
4.2.2.2 ~Cliente()	12
4.2.3 Member Function Documentation	12
4.2.3.1 adicionarFilmeAlugado()	12
4.2.3.2 adicionarPontos()	12
4.2.3.3 devolverFilmesAlugados()	13
4.2.3.4 getCpf()	13
4.2.3.5 getFilmesAlugados()	13
4.2.3.6 getNome()	13
4.2.3.7 getPontos()	13
4.2.3.8 mostrarInfo()	14
4.2.3.9 zerarPontos()	14
4.3 DVD Class Reference	14
4.3.1 Detailed Description	15
4.3.2 Constructor & Destructor Documentation	15
4.3.2.1 DVD()	15
4.3.3 Member Function Documentation	16
4.3.3.1 calcularValorLocacao()	16
4.3.4 Member Data Documentation	16
4.3.4.1 categoria	16

4.4 Filme Class Reference	17
4.4.1 Detailed Description	17
4.4.2 Member Enumeration Documentation	18
4.4.2.1 Tipo	18
4.4.3 Constructor & Destructor Documentation	18
4.4.3.1 Filme()	18
4.4.3.2 ~Filme()	19
4.4.4 Member Function Documentation	19
4.4.4.1 adicionarUnidadesDisponiveis()	19
4.4.4.2 calcularValorLocacao()	19
4.4.4.3 getCodigo()	20
4.4.4.4 getTipo()	20
4.4.4.5 getTitulo()	20
4.4.4.6 getUnidadesDisponiveis()	21
4.4.4.7 removerUnidadeDisponivel()	21
4.4.5 Friends And Related Symbol Documentation	21
4.4.5.1 operator<<	21
4.5 Fita Class Reference	22
4.5.1 Detailed Description	22
4.5.2 Constructor & Destructor Documentation	23
4.5.2.1 Fita()	23
4.5.3 Member Function Documentation	23
4.5.3.1 calcularValorLocacao()	23
4.5.3.2 estaRebobinada()	24
4.5.3.3 rebobinar()	24
4.6 Locacao Class Reference	24
4.6.1 Detailed Description	25
4.6.2 Member Function Documentation	25
4.6.2.1 alugarFilmes()	25
4.6.2.2 cadastrarCliente()	26
4.6.2.3 cadastrarFilme()	26
4.6.2.4 devolverFilmes()	26
4.6.2.5 emiteReciboAluguel()	27
4.6.2.6 emiteReciboDevolucao()	27
4.6.2.7 getCliente()	27
4.6.2.8 getFilme()	28
4.6.2.9 listarClientesCodigo()	28
4.6.2.10 listarClientesNome()	29
4.6.2.11 listarFilmesCodigo()	29
4.6.2.12 listarFilmesTitulo()	29
4.6.2.13 removerCliente()	29
4.6.2.14 removerFilme()	29

4.6.3 Member Data Documentation	30
4.6.3.1 _catalogo_filmes	30
4.6.3.2 _clientes_cadastrados	30
4.7 Sistema Class Reference	30
4.7.1 Detailed Description	31
4.7.2 Constructor & Destructor Documentation	31
4.7.2.1 Sistema()	31
4.7.3 Member Function Documentation	31
4.7.3.1 alugarFilmes()	31
4.7.3.2 cadastrarCliente()	32
4.7.3.3 cadastrarFilme()	32
4.7.3.4 devolverFilmes()	32
4.7.3.5 displayHelp()	32
4.7.3.6 lerArquivo()	33
4.7.3.7 listarClientes()	33
4.7.3.8 listarFilmes()	33
4.7.3.9 removerCliente()	33
4.7.3.10 removerFilme()	34
<b>5 File Documentation</b>	<b>35</b>
5.1 include/bluray.hpp File Reference	35
5.1.1 Detailed Description	35
5.2 bluray.hpp	35
5.3 include/cliente.hpp File Reference	36
5.3.1 Detailed Description	36
5.4 cliente.hpp	36
5.5 include/dvd.hpp File Reference	37
5.5.1 Detailed Description	37
5.5.2 Enumeration Type Documentation	37
5.5.2.1 Categoria	37
5.6 dvd.hpp	37
5.7 include/filme.hpp File Reference	38
5.7.1 Detailed Description	38
5.7.2 Function Documentation	38
5.7.2.1 operator<<()	38
5.8 filme.hpp	39
5.9 include/fita.hpp File Reference	39
5.9.1 Detailed Description	40
5.10 fita.hpp	40
5.11 include/formatacao.hpp File Reference	40
5.11.1 Detailed Description	41
5.11.2 Function Documentation	41

5.11.2.1 firstUpper()	41
5.11.2.2 toLowerCase()	41
5.11.2.3 toUpperCase()	42
5.12 formatacao.hpp	42
5.13 include/locacao.hpp File Reference	43
5.13.1 Detailed Description	43
5.14 locacao.hpp	43
5.15 include/sistema.hpp File Reference	44
5.15.1 Detailed Description	44
5.16 sistema.hpp	44
5.17 src/Cliente/cliente.cpp File Reference	44
5.17.1 Detailed Description	45
5.18 src/Filme/blueray.cpp File Reference	45
5.19 src/Filme/dvd.cpp File Reference	45
5.19.1 Detailed Description	45
5.20 src/Filme/filme.cpp File Reference	45
5.20.1 Detailed Description	45
5.21 src/Filme/fita.cpp File Reference	45
5.21.1 Detailed Description	45
5.22 src/formatacao.cpp File Reference	46
5.22.1 Detailed Description	46
5.22.2 Function Documentation	46
5.22.2.1 firstUpper()	46
5.22.2.2 toLowerCase()	46
5.22.2.3 toUpperCase()	47
5.23 src/locacao.cpp File Reference	47
5.23.1 Detailed Description	47
5.24 src/main.cpp File Reference	48
5.24.1 Detailed Description	48
5.24.2 Function Documentation	48
5.24.2.1 main()	48
5.25 src/sistema.cpp File Reference	48
5.25.1 Detailed Description	49

# Chapter 1

## Hierarchical Index

### 1.1 Class Hierarchy

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

Cliente . . . . .	11
Filme . . . . .	17
Bluray . . . . .	7
DVD . . . . .	14
Fita . . . . .	22
Locacao . . . . .	24
Sistema . . . . .	30





## Chapter 2

# Class Index

### 2.1 Class List

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

<a href="#">Bluray</a>	.....	7
<a href="#">Cliente</a>	.....	11
<a href="#">DVD</a>	.....	14
<a href="#">Filme</a>	.....	17
<a href="#">Fita</a>	.....	22
<a href="#">Locacao</a>	.....	
	Classe responsável por gerenciar as operações de locação de filmes	24
<a href="#">Sistema</a>	.....	
	Classe que representa o sistema de locação de filmes	30



## Chapter 3

# File Index

### 3.1 File List

Here is a list of all files with brief descriptions:

include/bluray.hpp	Declaração da classe <a href="#">Bluray</a> , seus métodos e atributos . . . . .	35
include/cliente.hpp	Declaração da classe <a href="#">Cliente</a> , seus métodos e atributos . . . . .	36
include/dvd.hpp	Declaração da classe <a href="#">DVD</a> , seus métodos e atributos . . . . .	37
include/filme.hpp	Declaração da classe <a href="#">Filme</a> , seus métodos e atributos . . . . .	38
include/fita.hpp	Declaração da classe <a href="#">Fita</a> , seus métodos e atributos . . . . .	39
include/formatacao.hpp	Funções para formatação de strings . . . . .	40
include/locacao.hpp	Declaração da classe <a href="#">Locacao</a> , seus métodos e atributos . . . . .	43
include/sistema.hpp	Declaração da classe <a href="#">Sistema</a> e seus métodos . . . . .	44
src/formatacao.cpp	Implementação das funções de formatação de strings . . . . .	46
src/locacao.cpp	Implementação dos métodos da classe <a href="#">Locacao</a> . . . . .	47
src/main.cpp	Arquivo principal para o programa Locadora PDS . . . . .	48
src/sistema.cpp	Implementação dos métodos da classe <a href="#">Sistema</a> . . . . .	48
src/Cliente/cliente.cpp	Implementação dos métodos da classe <a href="#">Cliente</a> . . . . .	44
src/Filme/bluray.cpp	. . . . .	45
src/Filme/dvd.cpp	Implementação dos métodos da classe <a href="#">DVD</a> , uma classe filha da classe <a href="#">Filme</a> . . . . .	45
src/Filme/filme.cpp	Implementação dos métodos da classe <a href="#">Filme</a> , uma classe mãe para os diferentes tipos de filmes . . . . .	45
src/Filme/fita.cpp	Implementação dos métodos da classe <a href="#">Fita</a> , uma classe filha da classe <a href="#">Filme</a> , representando uma fita de vídeo . . . . .	45



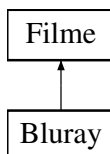
## Chapter 4

# Class Documentation

### 4.1 Bluray Class Reference

```
#include <bluray.hpp>
```

Inheritance diagram for Bluray:



#### Public Member Functions

- [Bluray](#) (const std::string &titulo, std::string codigo, int unidadesDisponiveis)  
*Construtor da classe [Bluray](#).*
- void [alugarAparelho](#) ()  
*Realiza o aluguel de um aparelho Blu-ray junto com o filme.*
- void [devolverAparelho](#) ()  
*Devolve o aparelho Blu-ray alugado.*
- double [calcularValorLocacao](#) (int dias) const override  
*Calcula o valor total da locação do Blu-ray.*

#### Public Member Functions inherited from [Filme](#)

- [Filme](#) (const std::string &titulo, std::string codigo, [Tipo](#) tipo, int unidadesDisponiveis)  
*Construtor da classe [Filme](#).*
- virtual [~Filme](#) ()  
*Destrutor da classe [Filme](#) (virtual para permitir polimorfismo).*
- std::string [getCodigo](#) () const  
*Getter para o código do filme.*
- const std::string & [getTitulo](#) () const  
*Getter para o título do filme.*
- int [getUnidadesDisponiveis](#) () const

Getter para o número de unidades disponíveis.

- Tipo `getTipo ()` const

Getter para o tipo do filme.

- void `adicionarUnidadesDisponiveis (int quantidade)`

Adiciona unidades disponíveis para locação.

- void `removerUnidadeDisponivel ()`

Remove uma unidade disponível.

## Static Public Member Functions

- static int `getQuantidadeAparelhosDisponiveis ()`

Obtém a quantidade de aparelhos Blu-ray disponíveis para locação.

- static void `setQuantidadeAparelhos (int quantidade)`

Define a quantidade de aparelhos Blu-ray disponíveis para locação.

## Additional Inherited Members

## Public Types inherited from Filme

- enum class Tipo { DVD , BluRay , Fita }

Enumeração que representa os tipos de filme.

### 4.1.1 Detailed Description

A classe `Bluray` contém atributos como título, código, quantidade de unidades disponíveis, quantidade de aparelhos Blu-ray e informações sobre o aluguel de aparelhos.

#### Parameters

<code>quantidadeAparelhos</code>	Quantidade de aparelhos Blu-ray disponíveis para aluguel.
<code>playerAlugado</code>	Indica se um player Blu-ray foi alugado junto com o filme.

### 4.1.2 Constructor & Destructor Documentation

#### 4.1.2.1 Bluray()

```
Bluray::Bluray (
    const std::string & _titulo,
    std::string _codigo,
    int _unidadesDisponiveis )
```

Construtor da classe `Bluray`.

#### Parameters

<code>titulo</code>	Título do filme.
<code>codigo</code>	Código único do filme.

## Parameters

<i>unidadesDisponiveis</i>	Número de unidades disponíveis para locação.
<i>_titulo</i>	Título do Blu-ray.
<i>_codigo</i>	Código único do Blu-ray.
<i>_unidadesDisponiveis</i>	Quantidade de unidades disponíveis do Blu-ray.

### 4.1.3 Member Function Documentation

#### 4.1.3.1 alugarAparelho()

```
void Bluray::alugarAparelho ( )
```

Realiza o aluguel de um aparelho Blu-ray junto com o filme.

Realiza o aluguel de um aparelho de Blu-ray, se disponível.

## Note

A quantidade de aparelhos disponíveis será reduzida.

## Exceptions

<i>std::invalid_argument</i>	Se não houver aparelhos disponíveis.
<i>std::invalid_argument</i>	Se todos os aparelhos de Blu-ray estiverem alugados.

#### 4.1.3.2 calcularValorLocacao()

```
double Bluray::calcularValorLocacao (
    int dias ) const [override], [virtual]
```

Calcula o valor total da locação do Blu-ray.

## Parameters

<i>dias</i>	Número de dias pelos quais o filme será alugado.
-------------	--

## Returns

O valor total da locação.

## Note

Adiciona um custo fixo se um aparelho Blu-ray foi alugado.

**Parameters**

<i>dias</i>	Número de dias pelos quais o filme será alugado.
-------------	--

**Returns**

O valor total da locação.

Implements [Filme](#).

**4.1.3.3 devolverAparelho()**

```
void Bluray::devolverAparelho ( )
```

Devolve o aparelho Blu-ray alugado.

Devolve um aparelho de Blu-ray.

**Note**

A quantidade de aparelhos disponíveis será incrementada.

**4.1.3.4 getQuantidadeAparelhosDisponiveis()**

```
int Bluray::getQuantidadeAparelhosDisponiveis ( ) [static]
```

Obtém a quantidade de aparelhos Blu-ray disponíveis para locação.

Obtém a quantidade de aparelhos de Blu-ray disponíveis.

**Returns**

A quantidade de aparelhos disponíveis.

A quantidade de aparelhos de Blu-ray disponíveis.

**4.1.3.5 setQuantidadeAparelhos()**

```
void Bluray::setQuantidadeAparelhos (
    int quantidade ) [static]
```

Define a quantidade de aparelhos Blu-ray disponíveis para locação.

Define a quantidade total de aparelhos de Blu-ray.

**Parameters**

<i>quantidade</i>	Nova quantidade de aparelhos disponíveis.
<i>quantidade</i>	A nova quantidade total de aparelhos de Blu-ray.



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

- include/bluray.hpp
- src/Filme/bluray.cpp

## 4.2 Cliente Class Reference

```
#include <cliente.hpp>
```

### Public Member Functions

- [Cliente](#) (string nome, string cpf)  
*Construtor da classe [Cliente](#).*
- [~Cliente](#) ()  
*Destrutor da classe [Cliente](#).*
- void [mostrarInfo](#) ()  
*Exibe informações do cliente.*
- void [adicionarPontos](#) (int pontos)  
*Adiciona pontos de fidelidade ao cliente.*
- void [zerarPontos](#) ()  
*Zera os pontos de fidelidade do cliente.*
- string [getNome](#) ()  
*Obtém o nome do cliente.*
- string [getCpf](#) ()  
*Obtém o CPF do cliente.*
- vector< [Filme](#) \* > & [getFilmesAlugados](#) ()  
*Obtém a lista de filmes alugados pelo cliente.*
- int [getPontos](#) ()  
*Obtém os pontos de fidelidade do cliente.*
- void [adicionarFilmeAlugado](#) ([Filme](#) \*filme)  
*Adiciona um filme à lista de filmes alugados pelo cliente.*
- void [devolverFilmesAlugados](#) ()  
*Devolve todos os filmes alugados pelo cliente.*

### 4.2.1 Detailed Description

A classe [Cliente](#) possui atributos como nome, CPF, pontos de fidelidade e uma lista de filmes alugados. Além disso, contém métodos para exibir informações do cliente, adicionar pontos de fidelidade, zerar pontos, obter nome, CPF, lista de filmes alugados e pontos de fidelidade, adicionar filme à lista de filmes alugados e devolver todos os filmes alugados.

#### Parameters

<code>_nome</code>	Nome do cliente.
<code>_cpf</code>	CPF do cliente.
<code>_pontos_fidelidade</code>	Pontos de fidelidade acumulados pelo cliente.
<code>_filmes_alugados</code>	Lista de filmes alugados pelo cliente.

## 4.2.2 Constructor & Destructor Documentation

### 4.2.2.1 Cliente()

```
Cliente::Cliente (
    string nome,
    string cpf )
```

Construtor da classe [Cliente](#).

#### Parameters

<i>nome</i>	Nome do cliente.
<i>cpf</i>	CPF do cliente.

### 4.2.2.2 ~Cliente()

```
Cliente::~~Cliente ( )
```

Destrutor da classe [Cliente](#).

## 4.2.3 Member Function Documentation

### 4.2.3.1 adicionarFilmeAlugado()

```
void Cliente::adicionarFilmeAlugado (
    Filme * filme )
```

Adiciona um filme à lista de filmes alugados pelo cliente.

#### Parameters

<i>filme</i>	<a href="#">Filme</a> a ser adicionado.
<i>filme</i>	Ponteiro para o filme a ser adicionado.

### 4.2.3.2 adicionarPontos()

```
void Cliente::adicionarPontos (
    int pontos )
```

Adiciona pontos de fidelidade ao cliente.

#### Parameters

<i>pontos</i>	Pontos a serem adicionados.
<i>pontos</i>	Quantidade de pontos a serem adicionados.

#### 4.2.3.3 devolverFilmesAlugados()

```
void Cliente::devolverFilmesAlugados ( )
```

Devolve todos os filmes alugados pelo cliente.

#### 4.2.3.4 getCpf()

```
string Cliente::getCpf ( )
```

Obtém o CPF do cliente.

##### Returns

CPF do cliente.

#### 4.2.3.5 getFilmesAlugados()

```
vector< Filme * > & Cliente::getFilmesAlugados ( )
```

Obtém a lista de filmes alugados pelo cliente.

##### Returns

Lista de filmes alugados.

Vetor contendo os filmes alugados.

#### 4.2.3.6 getNome()

```
string Cliente::getNome ( )
```

Obtém o nome do cliente.

##### Returns

Nome do cliente.

#### 4.2.3.7 getPontos()

```
int Cliente::getPontos ( )
```

Obtém os pontos de fidelidade do cliente.

Obtém a quantidade de pontos de fidelidade do cliente.

##### Returns

Pontos de fidelidade.

Quantidade de pontos de fidelidade.

#### 4.2.3.8 mostrarInfo()

```
void Cliente::mostrarInfo ( )
```

Exibe informações do cliente.

Exibe as informações do cliente, incluindo nome, CPF e pontos de fidelidade.

#### 4.2.3.9 zerarPontos()

```
void Cliente::zerarPontos ( )
```

Zera os pontos de fidelidade do cliente.

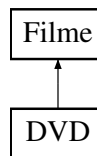
The documentation for this class was generated from the following files:

- [include/cliente.hpp](#)
- [src/Cliente/cliente.cpp](#)

## 4.3 DVD Class Reference

```
#include <dvd.hpp>
```

Inheritance diagram for DVD:



### Public Member Functions

- [DVD](#) (const std::string &titulo, std::string codigo, int unidadesDisponiveis, [Categoria](#) \_categoria)  
*Construtor da classe [DVD](#).*
- double [calcularValorLocacao](#) (int dias) const override  
*Calcula o valor total da locação do [DVD](#).*

### Public Member Functions inherited from [Filme](#)

- [Filme](#) (const std::string &titulo, std::string codigo, [Tipo](#) tipo, int unidadesDisponiveis)  
*Construtor da classe [Filme](#).*
- virtual [~Filme](#) ()  
*Destrutor da classe [Filme](#) (virtual para permitir polimorfismo).*
- std::string [getCodigo](#) () const  
*Getter para o código do filme.*
- const std::string & [getTitulo](#) () const  
*Getter para o título do filme.*
- int [getUnidadesDisponiveis](#) () const  
*Getter para o número de unidades disponíveis.*
- [Tipo](#) [getTipo](#) () const  
*Getter para o tipo do filme.*
- void [adicionarUnidadesDisponiveis](#) (int quantidade)  
*Adiciona unidades disponíveis para locação.*
- void [removerUnidadeDisponivel](#) ()  
*Remove uma unidade disponível.*

### Protected Attributes

- [Categoria categoria](#)

### Additional Inherited Members

### Public Types inherited from [Filme](#)

- enum class [Tipo](#) { [DVD](#) , [BluRay](#) , [Fita](#) }  
*Enumeração que representa os tipos de filme.*

### 4.3.1 Detailed Description

A classe [DVD](#) contém atributos como título, código, quantidade de unidades disponíveis e informações sobre a categoria do filme.

#### Parameters

<i>categoria</i>	Categoria do <a href="#">DVD</a> (Lançamento, Estoque, Promoção).
------------------	---

### 4.3.2 Constructor & Destructor Documentation

#### 4.3.2.1 DVD()

```
DVD::DVD (
    const std::string & _titulo,
    std::string _codigo,
    int _unidadesDisponiveis,
    Categoria _categoria )
```

Construtor da classe [DVD](#).

#### Parameters

<i>titulo</i>	Título do filme.
<i>codigo</i>	Código único do filme.
<i>unidadesDisponiveis</i>	Número de unidades disponíveis para locação.
<i>_categoria</i>	Categoria do <a href="#">DVD</a> (Lançamento, Estoque, Promoção).
<i>_titulo</i>	Título do <a href="#">DVD</a> .
<i>_codigo</i>	Código único do <a href="#">DVD</a> .
<i>_unidadesDisponiveis</i>	Quantidade de unidades disponíveis do <a href="#">DVD</a> .
<i>_categoria</i>	Categoria do <a href="#">DVD</a> (Lançamento, Estoque ou Promoção).

### 4.3.3 Member Function Documentation

#### 4.3.3.1 calcularValorLocacao()

```
double DVD::calcularValorLocacao (
    int dias ) const [override], [virtual]
```

Calcula o valor total da locação do [DVD](#).

Calcula o valor de locação do [DVD](#), considerando a categoria e a quantidade de dias.

##### Parameters

<i>dias</i>	Número de dias pelos quais o filme será alugado.
-------------	--

##### Returns

O valor total da locação.

##### Note

Adiciona um custo fixo baseado na categoria do [DVD](#).

##### Parameters

<i>dias</i>	O número de dias para locação.
-------------	--------------------------------

##### Returns

O valor total da locação do [DVD](#).

Implements [Filme](#).

### 4.3.4 Member Data Documentation

#### 4.3.4.1 categoria

```
Categoria DVD::categoria [protected]
```

Categoria do [DVD](#) (Lançamento, Estoque, Promoção).

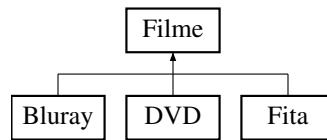
The documentation for this class was generated from the following files:

- include/[dvd.hpp](#)
- src/Filme/[dvd.cpp](#)

## 4.4 Filme Class Reference

```
#include <filme.hpp>
```

Inheritance diagram for Filme:



### Public Types

- enum class [Tipo](#) { [DVD](#) , [BluRay](#) , [Fita](#) }  
*Enumeração que representa os tipos de filme.*

### Public Member Functions

- [Filme](#) (const std::string &titulo, std::string codigo, [Tipo](#) tipo, int unidadesDisponiveis)  
*Construtor da classe [Filme](#).*
- virtual [~Filme](#) ()  
*Destrutor da classe [Filme](#) (virtual para permitir polimorfismo).*
- std::string [getCodigo](#) () const  
*Getter para o código do filme.*
- const std::string & [getTitulo](#) () const  
*Getter para o título do filme.*
- int [getUnidadesDisponiveis](#) () const  
*Getter para o número de unidades disponíveis.*
- [Tipo](#) [getTipo](#) () const  
*Getter para o tipo do filme.*
- void [adicionarUnidadesDisponiveis](#) (int quantidade)  
*Adiciona unidades disponíveis para locação.*
- void [removerUnidadeDisponivel](#) ()  
*Remove uma unidade disponível.*
- virtual double [calcularValorLocacao](#) (int dias) const =0  
*Calcula o valor total da locação do filme.*

### Friends

- std::ostream & [operator<<](#) (std::ostream &os, const [Filme::Tipo](#) &tipo)  
*Sobrecarga do operador de saída para imprimir o tipo do filme.*

### 4.4.1 Detailed Description

A classe [Filme](#) contém atributos como título, código, quantidade de unidades disponíveis, tipo do filme ([DVD](#), [BluRay](#), [Fita](#)) e métodos para manipulação desses atributos.

## Parameters

<i>titulo</i>	Título do filme.
<i>codigo</i>	Código único do filme.
<i>unidadesDisponiveis</i>	Número de unidades disponíveis para locação.
<i>tipo</i>	Tipo do filme ( <a href="#">DVD</a> , BluRay, <a href="#">Fita</a> ).

## 4.4.2 Member Enumeration Documentation

### 4.4.2.1 Tipo

```
enum class Filme::Tipo [strong]
```

Enumeração que representa os tipos de filme.

## Enumerator

DVD	Tipo de filme: <a href="#">DVD</a> .
BluRay	Tipo de filme: BluRay.
Fita	Tipo de filme: <a href="#">Fita</a> .

## 4.4.3 Constructor & Destructor Documentation

### 4.4.3.1 Filme()

```
Filme::Filme (
    const std::string & titulo,
    std::string codigo,
    Filme::Tipo tipo,
    int unidadesDisponiveis )
```

Construtor da classe [Filme](#).

## Parameters

<i>titulo</i>	Título do filme.
<i>codigo</i>	Código único do filme.
<i>tipo</i>	Tipo do filme ( <a href="#">DVD</a> , BluRay, <a href="#">Fita</a> ).
<i>unidadesDisponiveis</i>	Número de unidades disponíveis para locação.
<i>titulo</i>	Título do filme.
<i>codigo</i>	Código único do filme.
<i>tipo</i>	Tipo do filme ( <a href="#">DVD</a> , BluRay, etc.).
<i>unidadesDisponiveis</i>	Quantidade de unidades disponíveis do filme.



#### 4.4.3.2 ~Filme()

```
Filme::~~Filme ( ) [virtual]
```

Destruitor da classe [Filme](#) (virtual para permitir polimorfismo).

Destruitor da classe [Filme](#).

### 4.4.4 Member Function Documentation

#### 4.4.4.1 adicionarUnidadesDisponiveis()

```
void Filme::adicionarUnidadesDisponiveis (
    int quantidade )
```

Adiciona unidades disponíveis para locação.

Adiciona unidades disponíveis ao filme.

##### Parameters

<i>quantidade</i>	A quantidade de unidades a serem adicionadas.
-------------------	---

#### 4.4.4.2 calcularValorLocacao()

```
double Filme::calcularValorLocacao (
    int dias ) const [pure virtual]
```

Calcula o valor total da locação do filme.

Calcula o valor de locação do filme, considerando a quantidade de dias.

##### Parameters

<i>dias</i>	Número de dias pelos quais o filme será alugado.
-------------	--

##### Returns

O valor total da locação.

##### Note

Este método é virtual puro para permitir polimorfismo nas classes derivadas.

##### Parameters

<i>dias</i>	Número de dias para locação.
-------------	------------------------------

**Returns**

Valor total da locação do filme.

Implemented in [Bluray](#), [DVD](#), and [Fita](#).

**4.4.4.3 getCodigo()**

```
std::string Filme::getCodigo ( ) const
```

Getter para o código do filme.

Obtém o código único do filme.

**Returns**

O código único do filme.

**4.4.4.4 getTipo()**

```
Filme::Tipo Filme::getTipo ( ) const
```

Getter para o tipo do filme.

Obtém o tipo do filme.

**Returns**

O tipo do filme ([DVD](#), [BluRay](#), [Fita](#)).

O tipo do filme ([DVD](#), [BluRay](#), etc.).

**4.4.4.5 getTitulo()**

```
const std::string & Filme::getTitulo ( ) const
```

Getter para o título do filme.

Obtém o título do filme.

**Returns**

O título do filme.

#### 4.4.4.6 getUnidadesDisponiveis()

```
int Filme::getUnidadesDisponiveis ( ) const
```

Getter para o número de unidades disponíveis.

Obtém a quantidade de unidades disponíveis do filme.

##### Returns

O número de unidades disponíveis para locação.

A quantidade de unidades disponíveis do filme.

#### 4.4.4.7 removerUnidadeDisponivel()

```
void Filme::removerUnidadeDisponivel ( )
```

Remove uma unidade disponível.

Remove uma unidade disponível do filme.

### 4.4.5 Friends And Related Symbol Documentation

#### 4.4.5.1 operator<<

```
std::ostream & operator<< (
    std::ostream & os,
    const Filme::Tipo & tipo ) [friend]
```

Sobrecarga do operador de saída para imprimir o tipo do filme.

##### Parameters

<i>os</i>	Objeto de fluxo de saída.
<i>tipo</i>	Tipo do filme ( <a href="#">DVD</a> , <a href="#">BluRay</a> , <a href="#">Fita</a> ).

##### Returns

O objeto de fluxo de saída.

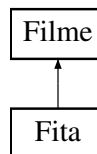
The documentation for this class was generated from the following files:

- [include/filme.hpp](#)
- [src/Filme/filme.cpp](#)

## 4.5 Fita Class Reference

```
#include <fita.hpp>
```

Inheritance diagram for Fita:



### Public Member Functions

- [Fita](#) (const std::string &titulo, std::string codigo, int quantidade, bool \_rebobinada)  
*Construtor da classe [Fita](#).*
- bool [estaRebobinada](#) () const  
*Verifica se a fita está rebobinada.*
- void [rebobinar](#) ()  
*Rebobina a fita.*
- double [calcularValorLocacao](#) (int dias) const override  
*Calcula o valor total da locação da fita.*

### Public Member Functions inherited from [Filme](#)

- [Filme](#) (const std::string &titulo, std::string codigo, [Tipo](#) tipo, int unidadesDisponiveis)  
*Construtor da classe [Filme](#).*
- virtual [~Filme](#) ()  
*Destrutor da classe [Filme](#) (virtual para permitir polimorfismo).*
- std::string [getCodigo](#) () const  
*Getter para o código do filme.*
- const std::string & [getTitulo](#) () const  
*Getter para o título do filme.*
- int [getUnidadesDisponiveis](#) () const  
*Getter para o número de unidades disponíveis.*
- [Tipo](#) [getTipo](#) () const  
*Getter para o tipo do filme.*
- void [adicionarUnidadesDisponiveis](#) (int quantidade)  
*Adiciona unidades disponíveis para locação.*
- void [removerUnidadeDisponivel](#) ()  
*Remove uma unidade disponível.*

### Additional Inherited Members

### Public Types inherited from [Filme](#)

- enum class [Tipo](#) { [DVD](#) , [BluRay](#) , [Fita](#) }  
*Enumeração que representa os tipos de filme.*

#### 4.5.1 Detailed Description

A classe [Fita](#) contém atributos como título, código, quantidade de unidades disponíveis, informações sobre se a fita está rebobinada e métodos para manipulação desses atributos.

## Parameters

<i>rebobinada</i>	Indica se a fita está rebobinada (true) ou não (false).
-------------------	---

## 4.5.2 Constructor & Destructor Documentation

### 4.5.2.1 Fita()

```
Fita::Fita (
    const std::string & _titulo,
    std::string _codigo,
    int _quantidade,
    bool _rebobinada )
```

Construtor da classe [Fita](#).

## Parameters

<i>titulo</i>	Título do filme.
<i>codigo</i>	Código único do filme.
<i>quantidade</i>	Número de unidades disponíveis para locação.
<i>_rebobinada</i>	Indica se a fita está rebobinada (true) ou não (false).
<i>_titulo</i>	Título da fita.
<i>_codigo</i>	Código único da fita.
<i>_quantidade</i>	Quantidade de unidades disponíveis da fita.
<i>_rebobinada</i>	Estado da rebobinação da fita.

## 4.5.3 Member Function Documentation

### 4.5.3.1 calcularValorLocacao()

```
double Fita::calcularValorLocacao (
    int dias ) const [override], [virtual]
```

Calcula o valor total da locação da fita.

Calcula o valor de locação da fita com base na quantidade de dias e no estado de rebobinação.

## Parameters

<i>dias</i>	Número de dias pelos quais o filme será alugado.
-------------	--

## Returns

O valor total da locação.

**Parameters**

<i>dias</i>	Quantidade de dias para locação.
-------------	----------------------------------

**Returns**

O valor total da locação da fita.

Implements [Filme](#).

**4.5.3.2 estaRebobinada()**

```
bool Fita::estaRebobinada ( ) const
```

Verifica se a fita está rebobinada.

**Returns**

True se a fita estiver rebobinada, false caso contrário.

Verdadeiro se a fita estiver rebobinada, falso caso contrário.

**4.5.3.3 rebobinar()**

```
void Fita::rebobinar ( )
```

Rebobina a fita.

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

- include/[fita.hpp](#)
- src/Filme/[fita.cpp](#)

## 4.6 Locacao Class Reference

Classe responsável por gerenciar as operações de locação de filmes.

```
#include <locacao.hpp>
```

## Public Member Functions

- void `cadastrarFilme` (`Filme` \*filme)  
*Cadastra um novo filme no catálogo.*
- void `removerFilme` (std::string codigo)  
*Remove um filme do catálogo pelo código.*
- void `alugarFilmes` (const std::string &cpf, std::vector< `Filme` \* > &filmes)  
*Realiza o aluguel de filmes para um cliente.*
- void `emiteReciboAluguel` (`Cliente` \*cliente, std::vector< `Filme` \* > &filmes)  
*Emite o recibo de aluguel para um cliente.*
- void `devolverFilmes` (`Cliente` \*cliente, int dias)  
*Realiza a devolução de filmes por parte do cliente.*
- void `emiteReciboDevolucao` (`Cliente` \*cliente, int dias)  
*Emite o recibo de devolução para um cliente.*
- void `listarFilmesCodigo` ()  
*Lista os filmes cadastrados no sistema ordenados por código.*
- void `listarFilmesTitulo` ()  
*Lista os filmes cadastrados no sistema ordenados por título.*
- void `cadastrarCliente` (const std::string &nome, const std::string &cpf)  
*Cadastra um novo cliente no sistema.*
- void `removerCliente` (const std::string &cpf)  
*Remove um cliente do sistema pelo CPF.*
- void `listarClientesCodigo` ()  
*Lista os clientes cadastrados no sistema ordenados por código.*
- void `listarClientesNome` ()  
*Lista os clientes cadastrados no sistema ordenados por nome.*
- `Cliente` \* `getCliente` (const std::string &cpf)  
*Obtém um cliente pelo CPF.*
- `Filme` \* `getFilme` (std::string codigo\_filme)  
*Obtém um filme pelo código.*

## Protected Attributes

- std::vector< `Filme` \* > `_catalogo_filmes`
- std::vector< `Cliente` \* > `_clientes_cadastrados`

### 4.6.1 Detailed Description

Classe responsável por gerenciar as operações de locação de filmes.

A classe `Locacao` realiza o cadastro, remoção, aluguel e devolução de filmes, além de manter registros dos filmes disponíveis no catálogo e dos clientes cadastrados.

### 4.6.2 Member Function Documentation

#### 4.6.2.1 `alugarFilmes()`

```
void Locacao::alugarFilmes (
    const std::string & cpf,
    std::vector< Filme * > & filmes_a_alugar )
```

Realiza o aluguel de filmes para um cliente.

Realiza a locação de filmes para um cliente.

## Parameters

<i>cpf</i>	CPF do cliente.
<i>filmes</i>	Vetor de ponteiros para os filmes a serem alugados.
<i>cpf</i>	CPF do cliente que deseja alugar os filmes.
<i>filmes_a_alugar</i>	Vetor de ponteiros para os filmes que o cliente deseja alugar.

**4.6.2.2 cadastrarCliente()**

```
void Locacao::cadastrarCliente (
    const std::string & nome,
    const std::string & cpf )
```

Cadastra um novo cliente no sistema.

## Parameters

<i>nome</i>	Nome do cliente.
<i>cpf</i>	CPF do cliente.
<i>nome</i>	Nome do cliente a ser cadastrado.
<i>cpf</i>	CPF do cliente a ser cadastrado.

**4.6.2.3 cadastrarFilme()**

```
void Locacao::cadastrarFilme (
    Filme * filme )
```

Cadastra um novo filme no catálogo.

## Parameters

<i>filme</i>	Ponteiro para o filme a ser cadastrado.
--------------	---

**4.6.2.4 devolverFilmes()**

```
void Locacao::devolverFilmes (
    Cliente * cliente,
    int dias )
```

Realiza a devolução de filmes por parte do cliente.

Realiza a devolução de filmes por um cliente.

## Parameters

<i>cliente</i>	Ponteiro para o cliente.
<i>dias</i>	Número de dias pelos quais os filmes foram alugados.
<i>cliente</i>	Ponteiro para o cliente que está devolvendo os filmes.
<i>dias</i>	Número de dias pelos quais os filmes foram alugados.



#### 4.6.2.5 emiteReciboAluguel()

```
void Locacao::emiteReciboAluguel (
    Cliente * cliente,
    std::vector< Filme * > & filmes )
```

Emita o recibo de aluguel para um cliente.

Emita um recibo de aluguel para o cliente.

##### Parameters

<i>cliente</i>	Ponteiro para o cliente.
<i>filmes</i>	Vetor de ponteiros para os filmes alugados.
<i>cliente</i>	Ponteiro para o cliente que alugou os filmes.
<i>filmes</i>	Vetor de ponteiros para os filmes alugados.

#### 4.6.2.6 emiteReciboDevolucao()

```
void Locacao::emiteReciboDevolucao (
    Cliente * cliente,
    int dias )
```

Emita o recibo de devolução para um cliente.

Emita um recibo de devolução para o cliente.

##### Parameters

<i>cliente</i>	Ponteiro para o cliente.
<i>dias</i>	Número de dias pelos quais os filmes foram alugados.
<i>cliente</i>	Ponteiro para o cliente que devolveu os filmes.
<i>dias</i>	Número de dias pelos quais os filmes foram alugados.

#### 4.6.2.7 getCliente()

```
Cliente * Locacao::getCliente (
    const std::string & cpf )
```

Obtém um cliente pelo CPF.

Retorna um ponteiro para um [Cliente](#) com base no CPF fornecido.

##### Parameters

<i>cpf</i>	CPF do cliente.
------------	-----------------

**Returns**

Ponteiro para o cliente encontrado, ou nullptr se não encontrado.

**Parameters**

<i>cpf</i>	CPF do cliente a ser encontrado.
------------	----------------------------------

**Returns**

Ponteiro para o [Cliente](#) encontrado ou nullptr se não for encontrado.

**4.6.2.8 getFilme()**

```
Filme * Locacao::getFilme (
    std::string codigo_filme )
```

Obtém um filme pelo código.

Retorna um ponteiro para um [Filme](#) com base no código fornecido.

**Parameters**

<i>codigo_filme</i>	Código do filme.
---------------------	------------------

**Returns**

Ponteiro para o filme encontrado, ou nullptr se não encontrado.

**Parameters**

<i>codigo_filme</i>	Código do filme a ser encontrado.
---------------------	-----------------------------------

**Returns**

Ponteiro para o [Filme](#) encontrado ou nullptr se não for encontrado.

**4.6.2.9 listarClientesCodigo()**

```
void Locacao::listarClientesCodigo ( )
```

Lista os clientes cadastrados no sistema ordenados por código.

Lista os clientes ordenados por código.

#### 4.6.2.10 listarClientesNome()

```
void Locacao::listarClientesNome ( )
```

Lista os clientes cadastrados no sistema ordenados por nome.

Lista os clientes ordenados por nome.

#### 4.6.2.11 listarFilmesCodigo()

```
void Locacao::listarFilmesCodigo ( )
```

Lista os filmes cadastrados no sistema ordenados por código.

Lista os filmes ordenados por código.

#### 4.6.2.12 listarFilmesTitulo()

```
void Locacao::listarFilmesTitulo ( )
```

Lista os filmes cadastrados no sistema ordenados por título.

Lista os filmes ordenados por título.

#### 4.6.2.13 removerCliente()

```
void Locacao::removerCliente (
    const std::string & cpf )
```

Remove um cliente do sistema pelo CPF.

Remove um cliente do sistema com base no CPF fornecido.

##### Parameters

<i>cpf</i>	CPF do cliente a ser removido.
<i>cpf</i>	CPF do cliente a ser removido.

##### Exceptions

<i>std::invalid_argument</i>	se o cliente não for encontrado.
------------------------------	----------------------------------

#### 4.6.2.14 removerFilme()

```
void Locacao::removerFilme (
    std::string codigo )
```

Remove um filme do catálogo pelo código.

Remove um filme do catálogo com base no código fornecido.

#### Parameters

<i>codigo</i>	Código do filme a ser removido.
<i>codigo</i>	Código do filme a ser removido.

#### Exceptions

<i>std::invalid_argument</i>	se o filme não for encontrado.
------------------------------	--------------------------------

### 4.6.3 Member Data Documentation

#### 4.6.3.1 \_catalogo\_filmes

```
std::vector<Filme*> Locacao::_catalogo_filmes [protected]
```

Vetor que armazena os filmes disponíveis para locação.

#### 4.6.3.2 \_clientes\_cadastrados

```
std::vector<Cliente*> Locacao::_clientes_cadastrados [protected]
```

Vetor que armazena os clientes cadastrados no sistema.

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

- [include/locacao.hpp](#)
- [src/locacao.cpp](#)

## 4.7 Sistema Class Reference

Classe que representa o sistema de locação de filmes.

```
#include <sistema.hpp>
```

## Public Member Functions

- [Sistema](#) ()  
*Construtor da classe [Sistema](#).*
- void [displayHelp](#) ()  
*Exibe informações de ajuda sobre os comandos disponíveis no sistema.*
- void [cadastrarFilme](#) ()  
*Realiza a operação de cadastrar um novo filme no sistema.*
- void [cadastrarCliente](#) ()  
*Realiza a operação de cadastrar um novo cliente no sistema.*
- void [removerCliente](#) ()  
*Realiza a operação de remover um cliente do sistema.*
- void [removerFilme](#) ()  
*Realiza a operação de remover um filme do sistema.*
- void [listarFilmes](#) ()  
*Lista os filmes cadastrados no sistema.*
- void [listarClientes](#) ()  
*Lista os clientes cadastrados no sistema.*
- void [devolverFilmes](#) ()  
*Realiza a operação de devolução de filmes por parte do cliente.*
- void [alugarFilmes](#) ()  
*Realiza a operação de aluguel de filmes para um cliente.*
- void [lerArquivo](#) ()  
*Realiza a leitura de informações de arquivo para inicializar o sistema.*

### 4.7.1 Detailed Description

Classe que representa o sistema de locação de filmes.

A classe [Sistema](#) gerencia as operações do sistema de locação, interagindo com a classe [Locacao](#).

### 4.7.2 Constructor & Destructor Documentation

#### 4.7.2.1 Sistema()

```
Sistema::Sistema ( )
```

Construtor da classe [Sistema](#).

Construtor da classe [Sistema](#). Inicializa o objeto de [Locacao](#) no construtor do [Sistema](#).

Inicializa uma instância da classe [Sistema](#).

### 4.7.3 Member Function Documentation

#### 4.7.3.1 alugarFilmes()

```
void Sistema::alugarFilmes ( )
```

Realiza a operação de aluguel de filmes para um cliente.

Solicita informações e aluga filmes para um cliente.

**Exceptions**

<i>std::invalid_argument</i>	se o cliente não for encontrado ou o código do filme for inexistente.
------------------------------	---

**4.7.3.2 cadastrarCliente()**

```
void Sistema::cadastrarCliente ( )
```

Realiza a operação de cadastrar um novo cliente no sistema.

Solicita informações e cadastra um novo cliente na locadora.

**Exceptions**

<i>std::invalid_argument</i>	se o CPF fornecido for inválido.
------------------------------	----------------------------------

**4.7.3.3 cadastrarFilme()**

```
void Sistema::cadastrarFilme ( )
```

Realiza a operação de cadastrar um novo filme no sistema.

Solicita informações e cadastra um novo filme na locadora.

**Exceptions**

<i>std::invalid_argument</i>	se o tipo de filme for inválido.
------------------------------	----------------------------------

**4.7.3.4 devolverFilmes()**

```
void Sistema::devolverFilmes ( )
```

Realiza a operação de devolução de filmes por parte do cliente.

Permite que um cliente devolva filmes alugados.

**Exceptions**

<i>std::out_of_range</i>	se nenhum filme estiver alugado para o cliente.
<i>std::invalid_argument</i>	se o cliente não for encontrado.

**4.7.3.5 displayHelp()**

```
void Sistema::displayHelp ( )
```

Exibe informações de ajuda sobre os comandos disponíveis no sistema.

Exibe o menu de ajuda com os comandos disponíveis.

#### 4.7.3.6 lerArquivo()

```
void Sistema::lerArquivo ( )
```

Realiza a leitura de informações de arquivo para inicializar o sistema.

Lê informações de um arquivo e cadastra os filmes na locadora.

##### Exceptions

<code>std::invalid_argument</code>	se o arquivo não existir.
------------------------------------	---------------------------

#### 4.7.3.7 listarClientes()

```
void Sistema::listarClientes ( )
```

Lista os clientes cadastrados no sistema.

Lista os clientes cadastrados na locadora por CPF ou nome.

##### Exceptions

<code>std::invalid_argument</code>	se a opção de listagem for inválida.
------------------------------------	--------------------------------------

#### 4.7.3.8 listarFilmes()

```
void Sistema::listarFilmes ( )
```

Lista os filmes cadastrados no sistema.

Lista os filmes disponíveis na locadora por código ou título.

##### Exceptions

<code>std::invalid_argument</code>	se a opção de listagem for inválida.
------------------------------------	--------------------------------------

#### 4.7.3.9 removerCliente()

```
void Sistema::removerCliente ( )
```

Realiza a operação de remover um cliente do sistema.

Remove um cliente da locadora com base no CPF.

### Exceptions

<code>std::out_of_range</code>	se nenhum filme estiver alugado para o cliente.
--------------------------------	---

#### 4.7.3.10 removerFilme()

```
void Sistema::removerFilme ( )
```

Realiza a operação de remover um filme do sistema.

Remove um filme da locadora com base no código.

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

- [include/sistema.hpp](#)
- [src/sistema.cpp](#)



# Chapter 5

## File Documentation

### 5.1 include/bluray.hpp File Reference

Declaração da classe [Bluray](#), seus métodos e atributos.

```
#include <iostream>
#include <string>
#include "filme.hpp"
```

#### Classes

- class [Bluray](#)

#### 5.1.1 Detailed Description

Declaração da classe [Bluray](#), seus métodos e atributos.

Este arquivo define a classe [Bluray](#), que representa um filme no formato Blu-ray. A classe herda da classe [Filme](#) e adiciona funcionalidades específicas ao Blu-ray, como a gestão de aparelhos Blu-ray e cálculo de valor de locação.

### 5.2 bluray.hpp

[Go to the documentation of this file.](#)

```
00001 #ifndef BLURAY_HPP
00002 #define BLURAY_HPP
00003
00004 #include <iostream>
00005 #include <string>
00006
00007 #include "filme.hpp"
00008
00022 class Bluray : public Filme {
00023 private:
00024     static int quantidadeAparelhos;
00025     bool playerAlugado;
00027 public:
00034     Bluray(const std::string& titulo, std::string codigo, int unidadesDisponiveis);
00035
00040     static int getQuantidadeAparelhosDisponiveis();
00041
00046     static void setQuantidadeAparelhos(int quantidade);
00047
00054     void alugarAparelho();
00055
00060     void devolverAparelho();
00061
00068     double calcularValorLocacao(int dias) const override;
00069 };
00070
00071 #endif // BLURAY_HPP
```

## 5.3 include/cliente.hpp File Reference

Declaração da classe [Cliente](#), seus métodos e atributos.

```
#include <iostream>
#include <vector>
#include "filme.hpp"
```

### Classes

- class [Cliente](#)

### 5.3.1 Detailed Description

Declaração da classe [Cliente](#), seus métodos e atributos.

Este arquivo define a classe [Cliente](#), que representa um cliente do sistema de locação de filmes.

## 5.4 cliente.hpp

[Go to the documentation of this file.](#)

```
00001 #ifndef CLIENTE_H
00002 #define CLIENTE_H
00003
00004 #include <iostream>
00005 #include <vector>
00006
00007 #include "filme.hpp"
00008
00009 using namespace std;
00010
00026 class Cliente
00027 {
00028 private:
00029     string _nome;
00030     string _cpf;
00031     int _pontos_fidelidade;
00032     vector <Filme*> _filmes_alugados;
00034 public:
00040     Cliente(string nome, string cpf);
00041
00045     ~Cliente();
00046
00050     void mostrarInfo();
00051
00056     void adicionarPontos(int pontos);
00057
00061     void zerarPontos();
00062
00063     // Getters
00064
00069     string getNome();
00070
00075     string getCpf();
00076
00081     vector<Filme*>& getFilmesAlugados();
00082
00087     int getPontos();
00088
00093     void adicionarFilmeAlugado(Filme* filme);
00094
00098     void devolverFilmesAlugados();
00099 };
00100
00101 #endif // CLIENTE_H
```

## 5.5 include/dvd.hpp File Reference

Declaração da classe [DVD](#), seus métodos e atributos.

```
#include "filme.hpp"
```

### Classes

- class [DVD](#)

### Enumerations

- enum class [Categoria](#) { [Lancamento](#) , [Estoque](#) , [Promocao](#) }

### 5.5.1 Detailed Description

Declaração da classe [DVD](#), seus métodos e atributos.

Este arquivo define a classe [DVD](#), que representa um filme no formato [DVD](#). A classe herda da classe [Filme](#) e adiciona funcionalidades específicas ao [DVD](#), como a definição da categoria e cálculo de valor de locação.

### 5.5.2 Enumeration Type Documentation

#### 5.5.2.1 Categoria

```
enum class Categoria [strong]
```

#### Enumerator

Lancamento	Categoria de lançamento do <a href="#">DVD</a> .
Estoque	Categoria de estoque do <a href="#">DVD</a> .
Promocao	Categoria de promoção do <a href="#">DVD</a> .

## 5.6 dvd.hpp

[Go to the documentation of this file.](#)

```
00001 #ifndef DVD_HPP
00002 #define DVD_HPP
00003
00004 #include "filme.hpp"
00005
00018 enum class Categoria {
00019     Lancamento,
00020     Estoque,
00021     Promocao
00022 };
00023
00024 class DVD : public Filme {
```

```

00025 protected:
00026     Categoria categoria;
00028 public:
00036     DVD(const std::string& titulo, std::string codigo, int unidadesDisponiveis, Categoria _categoria);
00037
00044     double calcularValorLocacao(int dias) const override;
00045 };
00046
00047 #endif // DVD_HPP

```

## 5.7 include/filme.hpp File Reference

Declaração da classe [Filme](#), seus métodos e atributos.

```

#include <iostream>
#include <string>

```

### Classes

- class [Filme](#)

### Functions

- `std::ostream & operator<< (std::ostream &os, const Filme::Tipo &tipo)`  
*Sobrecarga do operador de saída para imprimir o tipo do filme.*

### 5.7.1 Detailed Description

Declaração da classe [Filme](#), seus métodos e atributos.

Este arquivo define a classe base [Filme](#), que representa um filme genérico. As classes específicas ([DVD](#), [BluRay](#), [Fita](#)) herdam desta classe base.

### 5.7.2 Function Documentation

#### 5.7.2.1 operator<<()

```

std::ostream & operator<< (
    std::ostream & os,
    const Filme::Tipo & tipo ) [inline]

```

Sobrecarga do operador de saída para imprimir o tipo do filme.

#### Parameters

<i>os</i>	Objeto de fluxo de saída.
<i>tipo</i>	Tipo do filme ( <a href="#">DVD</a> , <a href="#">BluRay</a> , <a href="#">Fita</a> ).

## Returns

O objeto de fluxo de saída.

## 5.8 filme.hpp

[Go to the documentation of this file.](#)

```

00001 #ifndef FILME_HPP
00002 #define FILME_HPP
00003
00004 #include <iostream>
00005 #include <string>
00006
00021 class Filme {
00022 public:
00026     enum class Tipo {
00027         DVD,
00028         BluRay,
00029         Fita
00030     };
00031
00032 private:
00033     std::string titulo;
00034     std::string codigo;
00035     int unidadesDisponiveis;
00036     Tipo tipo;
00038 public:
00046     Filme(const std::string& titulo, std::string codigo, Tipo tipo, int unidadesDisponiveis);
00047
00051     virtual ~Filme();
00052
00057     std::string getCodigo() const;
00058
00063     const std::string& getTitulo() const;
00064
00069     int getUnidadesDisponiveis() const;
00070
00075     Tipo getTipo() const;
00076
00081     void adicionarUnidadesDisponiveis(int quantidade);
00082
00086     void removerUnidadeDisponivel();
00087
00094     virtual double calcularValorLocacao(int dias) const = 0;
00095
00102     friend std::ostream& operator<<(std::ostream& os, const Filme::Tipo& tipo);
00103 };
00104
00111 inline std::ostream& operator<<(std::ostream& os, const Filme::Tipo& tipo) {
00112     switch (tipo) {
00113         case Filme::Tipo::DVD:
00114             os << "DVD";
00115             break;
00116         case Filme::Tipo::BluRay:
00117             os << "BluRay";
00118             break;
00119         case Filme::Tipo::Fita:
00120             os << "Fita";
00121             break;
00122     }
00123     return os;
00124 }
00125
00126 #endif // FILME_HPP

```

## 5.9 include/fita.hpp File Reference

Declaração da classe [Fita](#), seus métodos e atributos.

```
#include "filme.hpp"
```

## Classes

- class [Fita](#)

### 5.9.1 Detailed Description

Declaração da classe [Fita](#), seus métodos e atributos.

Este arquivo define a classe [Fita](#), que representa um filme no formato de fita. A classe herda da classe [Filme](#) e adiciona funcionalidades específicas às fitas, como rebobinamento e cálculo de valor de locação.

## 5.10 fita.hpp

[Go to the documentation of this file.](#)

```
00001 #ifndef FITA_HPP
00002 #define FITA_HPP
00003
00004 #include "filme.hpp"
00005
00018 class Fita : public Filme {
00019 private:
00020     bool rebobinada;
00022 public:
00030     Fita(const std::string& titulo, std::string codigo, int quantidade, bool _rebobinada);
00031
00036     bool estaRebobinada() const;
00037
00041     void rebobinar();
00042
00048     double calcularValorLocacao(int dias) const override;
00049 };
00050
00051 #endif // FITA_HPP
```

## 5.11 include/formatacao.hpp File Reference

Funções para formatação de strings.

```
#include <iostream>
#include <string>
#include <algorithm>
#include <cctype>
#include <iomanip>
```

## Functions

- `std::string toLowerCase (const std::string &input)`  
*Converte uma string para letras minúsculas.*
- `std::string toUpperCase (const std::string &input)`  
*Converte uma string para letras maiúsculas.*
- `std::string firstUpper (const std::string &input)`  
*Converte a primeira letra de uma string para maiúscula.*

### 5.11.1 Detailed Description

Funções para formatação de strings.

### 5.11.2 Function Documentation

#### 5.11.2.1 firstUpper()

```
std::string firstUpper (
    const std::string & input )
```

Converte a primeira letra de uma string para maiúscula.

##### Parameters

<i>input</i>	A string a ser convertida.
--------------	----------------------------

##### Returns

A string com a primeira letra em maiúscula.

Converte a primeira letra de uma string para maiúscula.

##### Parameters

<i>input</i>	A string a ser convertida.
--------------	----------------------------

##### Returns

A string com a primeira letra de cada palavra em maiúscula.

#### 5.11.2.2 toLowerCase()

```
std::string toLowerCase (
    const std::string & input )
```

Converte uma string para letras minúsculas.

##### Parameters

<i>input</i>	A string a ser convertida.
--------------	----------------------------

##### Returns

A string convertida para letras minúsculas.

Converte uma string para letras minúsculas.

**Parameters**

<i>input</i>	A string a ser convertida.
--------------	----------------------------

**Returns**

A string convertida para minúsculas.

**5.11.2.3 toUpperCase()**

```
std::string toUpperCase (  
    const std::string & input )
```

Converte uma string para letras maiúsculas.

**Parameters**

<i>input</i>	A string a ser convertida.
--------------	----------------------------

**Returns**

A string convertida para letras maiúsculas.

Converte uma string para letras maiúsculas.

**Parameters**

<i>input</i>	A string a ser convertida.
--------------	----------------------------

**Returns**

A string convertida para maiúsculas.

**5.12 formatacao.hpp**

[Go to the documentation of this file.](#)

```
00001 #ifndef FORMATACAO_H  
00002 #define FORMATACAO_H  
00003  
00004 #include <iostream>  
00005 #include <string>  
00006 #include <algorithm>  
00007 #include <cctype>  
00008 #include <iomanip>  
00009  
00021 std::string toLowerCase(const std::string& input);  
00022  
00029 std::string toUpperCase(const std::string& input);  
00030  
00037 std::string firstUpper(const std::string& input);  
00038  
00039 #endif // FORMATACAO_H
```



## 5.13 include/locacao.hpp File Reference

Declaração da classe [Locacao](#), seus métodos e atributos.

```
#include <iostream>
#include <vector>
#include "filme.hpp"
#include "cliente.hpp"
```

### Classes

- class [Locacao](#)

*Classe responsável por gerenciar as operações de locação de filmes.*

### 5.13.1 Detailed Description

Declaração da classe [Locacao](#), seus métodos e atributos.

## 5.14 locacao.hpp

[Go to the documentation of this file.](#)

```
00001 #ifndef LOCACAO_HPP
00002 #define LOCACAO_HPP
00003
00004 #include <iostream>
00005 #include <vector>
00006
00007 #include "filme.hpp"
00008 #include "cliente.hpp"
00009
00022 class Locacao {
00023 protected:
00024     std::vector<Filme*> _catalogo_filmes;
00025     std::vector<Cliente*> _clientes_cadastrados;
00027 public:
00033     void cadastrarFilme(Filme* filme);
00034
00040     void removerFilme(std::string codigo);
00041
00048     void alugarFilmes(const std::string& cpf, std::vector<Filme*>& filmes);
00049
00056     void emitirReciboAluguel(Cliente* cliente, std::vector<Filme*>& filmes);
00057
00064     void devolverFilmes(Cliente* cliente, int dias);
00065
00072     void emitirReciboDevolucao(Cliente* cliente, int dias);
00073
00077     void listarFilmesCodigo();
00078
00082     void listarFilmesTitulo();
00083
00090     void cadastrarCliente(const std::string& nome, const std::string& cpf);
00091
00097     void removerCliente(const std::string& cpf);
00098
00102     void listarClientesCodigo();
00103
00107     void listarClientesNome();
00108
00115     Cliente* getCliente(const std::string& cpf);
00116
00123     Filme* getFilme(std::string codigo_filme);
00124 };
00125
00126 #endif // LOCACAO_HPP
```

## 5.15 include/sistema.hpp File Reference

Declaração da classe [Sistema](#) e seus métodos.

```
#include "locacao.hpp"
#include "filme.hpp"
```

### Classes

- class [Sistema](#)

*Classe que representa o sistema de locação de filmes.*

### 5.15.1 Detailed Description

Declaração da classe [Sistema](#) e seus métodos.

## 5.16 sistema.hpp

[Go to the documentation of this file.](#)

```
00001 #ifndef SISTEMA_H
00002 #define SISTEMA_H
00003
00004 #include "locacao.hpp"
00005 #include "filme.hpp"
00006
00017 class Sistema {
00018 private:
00019     Locacao _locacao;
00020     vector<Filme*> filmes;
00022 public:
00028     Sistema();
00029
00033     void displayHelp();
00034
00038     void cadastrarFilme();
00039
00043     void cadastrarCliente();
00044
00048     void removerCliente();
00049
00053     void removerFilme();
00054
00058     void listarFilmes();
00059
00063     void listarClientes();
00064
00068     void devolverFilmes();
00069
00073     void alugarFilmes();
00074
00078     void lerArquivo();
00079 };
00080
00081 #endif // SISTEMA_H
```

## 5.17 src/Cliente/cliente.cpp File Reference

Implementação dos métodos da classe [Cliente](#).

```
#include "cliente.hpp"
#include <algorithm>
```

### 5.17.1 Detailed Description

Implementação dos métodos da classe [Cliente](#).

## 5.18 src/Filme/blueray.cpp File Reference

```
#include "bluray.hpp"
#include <stdexcept>
```

## 5.19 src/Filme/dvd.cpp File Reference

Implementação dos métodos da classe [DVD](#), uma classe filha da classe [Filme](#).

```
#include "dvd.hpp"
```

### 5.19.1 Detailed Description

Implementação dos métodos da classe [DVD](#), uma classe filha da classe [Filme](#).

## 5.20 src/Filme/filme.cpp File Reference

Implementação dos métodos da classe [Filme](#), uma classe mãe para os diferentes tipos de filmes.

```
#include "filme.hpp"
```

### 5.20.1 Detailed Description

Implementação dos métodos da classe [Filme](#), uma classe mãe para os diferentes tipos de filmes.

## 5.21 src/Filme/fita.cpp File Reference

Implementação dos métodos da classe [Fita](#), uma classe filha da classe [Filme](#), representando uma fita de vídeo.

```
#include "fita.hpp"
```

### 5.21.1 Detailed Description

Implementação dos métodos da classe [Fita](#), uma classe filha da classe [Filme](#), representando uma fita de vídeo.

## 5.22 src/formatacao.cpp File Reference

Implementação das funções de formatação de strings.

```
#include "formatacao.hpp"
#include <algorithm>
#include <cctype>
```

### Functions

- `std::string toLowerCase` (const `std::string` &input)  
*Converte uma string para minúsculas.*
- `std::string toUpperCase` (const `std::string` &input)  
*Converte uma string para maiúsculas.*
- `std::string firstUpper` (const `std::string` &input)  
*Converte a primeira letra de cada palavra para maiúscula.*

### 5.22.1 Detailed Description

Implementação das funções de formatação de strings.

### 5.22.2 Function Documentation

#### 5.22.2.1 firstUpper()

```
std::string firstUpper (
    const std::string & input )
```

Converte a primeira letra de cada palavra para maiúscula.

Converte a primeira letra de uma string para maiúscula.

#### Parameters

<i>input</i>	A string a ser convertida.
--------------	----------------------------

#### Returns

A string com a primeira letra de cada palavra em maiúscula.

#### 5.22.2.2 toLowerCase()

```
std::string toLowerCase (
    const std::string & input )
```

Converte uma string para minúsculas.

Converte uma string para letras minúsculas.

**Parameters**

<i>input</i>	A string a ser convertida.
--------------	----------------------------

**Returns**

A string convertida para minúsculas.

**5.22.2.3 toUpperCase()**

```
std::string toUpperCase (
    const std::string & input )
```

Converte uma string para maiúsculas.

Converte uma string para letras maiúsculas.

**Parameters**

<i>input</i>	A string a ser convertida.
--------------	----------------------------

**Returns**

A string convertida para maiúsculas.

**5.23 src/locacao.cpp File Reference**

Implementação dos métodos da classe [Locacao](#).

```
#include "locacao.hpp"
#include "dvd.hpp"
#include "bluray.hpp"
#include "fita.hpp"
#include <algorithm>
#include <iostream>
```

**5.23.1 Detailed Description**

Implementação dos métodos da classe [Locacao](#).

## 5.24 src/main.cpp File Reference

Arquivo principal para o programa Locadora PDS.

```
#include <iostream>
#include <string>
#include <fstream>
#include <algorithm>
#include <cctype>
#include <iomanip>
#include "locacao.hpp"
#include "dvd.hpp"
#include "bluray.hpp"
#include "fita.hpp"
#include "formatacao.hpp"
#include "sistema.hpp"
```

### Functions

- int [main](#) ()

*Função principal para o programa Locadora PDS.*

### 5.24.1 Detailed Description

Arquivo principal para o programa Locadora PDS.

### 5.24.2 Function Documentation

#### 5.24.2.1 main()

```
int main ( )
```

Função principal para o programa Locadora PDS.

#### Returns

0 em execução bem-sucedida.

## 5.25 src/sistema.cpp File Reference

Implementação dos métodos da classe [Sistema](#).

```
#include <iostream>
#include <string>
#include <fstream>
#include <algorithm>
#include <cctype>
#include <iomanip>
#include "locacao.hpp"
#include "dvd.hpp"
#include "bluray.hpp"
#include "fita.hpp"
#include "formatacao.hpp"
#include "sistema.hpp"
```

### 5.25.1 Detailed Description

Implementação dos métodos da classe [Sistema](#).

