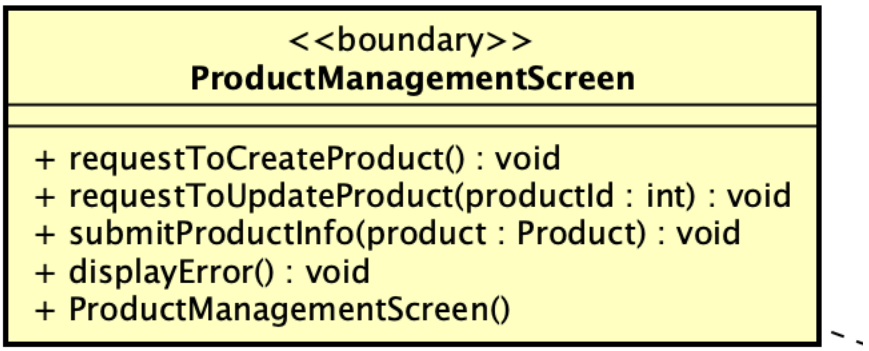
Design for class "ProductManagementScreen"



#	Name	Data type	Default value	Description
1				
2				

Table 1. Example of attribute design

#	Name	Return type	Description (purpose)
1	requestToCreateProduct	void	Initiates the process to create a new product
2	requestToUpdateProduct	void	Initiates the process to update an existing product
3	submitProductInfo	void	Submits the product information to the system
4	displayError	void	Display error to the Product Manager
5	ProductManagementScreen		Constructor for the class

Table 2. Example of operation design

Parameter

#	Name	Default value	Description

1	productId	None	The unique identifier for a product, used in requestToUpdateProduct()
2	product	None	The Product object containing all product information, used in submitProductInfo()

Exception

#	**Name**	**Description**
1	InvalidProductException	Thrown when product information is invalid
2	ProductNotFoundException	Thrown when a product with the specified ID cannot be found

Method

How to use parameters / attributes

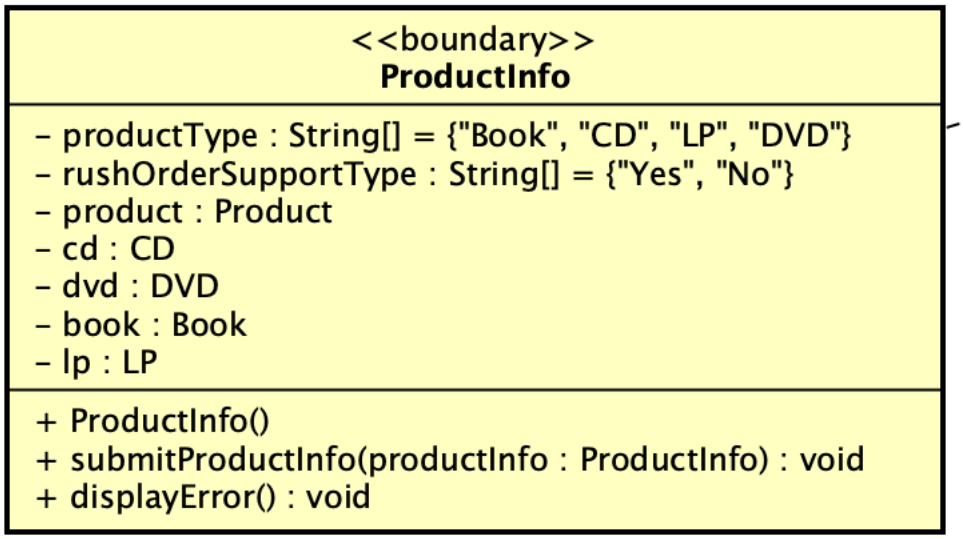
- The **productId** parameter is used to identify which product to update when calling the requestToUpdateProduct method.
- The **product** parameter is a Product object, containing all the information of a product, including its relevant details that need to be submitted to the system.

Flowchart / activity diagram / sequence diagram if the method has a complex / special algorithm

State

State diagram if any

Design for class "ProductInfo"



#	Name	Data type	Default value	Description
1	productType	String[]	{"Book", "CD", "LP", "DVD"}	Array of supported product types
2	rushOrderSupportType	String[]	{"Yes", "No"}	Options for rush order support
3	product	Product	None	Reference to the general Product object
4	cd	CD	None	Reference to CD type product
5	dvd	DVD	None	Reference to DVD type product
6	lp	LP	None	Reference to LP type product
7	book	Book	None	Reference to Book type product

Table 1. Example of attribute design

#	Name	Return type	Description (purpose)
1	ProductInfo		Constructor for the class
2	submitProductInfo	void	Submits the product information to the system
3	displayError	void	Display error to the Product Manager

Table 2. Example of operation design

Parameter

#	Name	Default value	Description
1	productInfo	None	The ProductInfo object containing all product information (used in submitProductInfo)

Exception

#	Name	Description
1	InvalidProductInfoException	Thrown when product information is invalid or incomplete
2	UnsupportedProductTypeException	Thrown when an unsupported product type is specified

Method

How to use parameters / attributes

- The **productType** attribute is used to determine which type of product is being created or modified, which affects what fields are displayed and required.

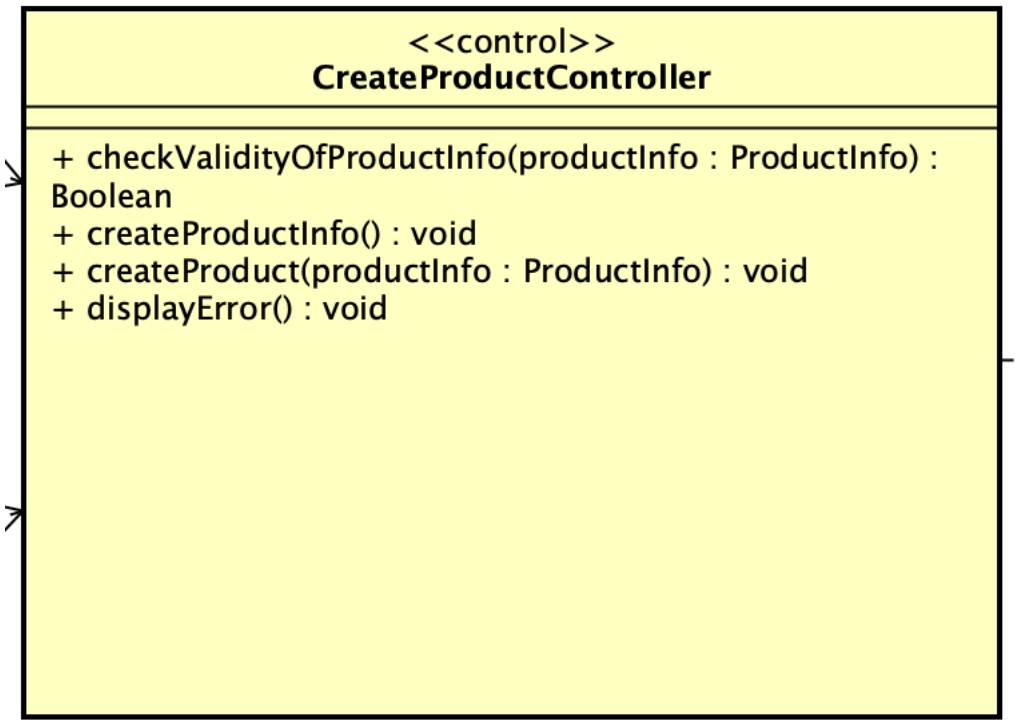
- The **rushOrderSupportType** attribute indicates whether rush order is available for the product.
- The **product, cd, dvd, book,** and **lp** attributes store references to the appropriate product objects based on the selected product type. Only one of these specific product type references (cd, dvd, book, or lp) will be populated based on the productType selection.
- The **productInfo** parameter in submitProductInfo() contains all the information that needs to be validated and submitted to the system.

Flowchart / activity diagram / sequence diagram if the method has a complex / special algorithm

State

State diagram if any

Design for class "CreateProductController"



#	Name	Data type	Default value	Description
1				
2				

Table 1. Example of attribute design

#	Name	Return type	Description (purpose)
1	checkValidityOfProductInfo	Boolean	Validates the product information provided
2	createProductInfo	void	Initializes the process to create new product information
3	createProduct	void	Creates a new product in the system
4	displayError	void	Notify error messages to the user

Table 2. Example of operation design

Parameter

#	Name	Default value	Description
1	productInfo	None	The ProductInfo object containing all product details (used in checkValidityOfProductInfo and createProduct)

Exception

#	Name	Description
1	InvalidProductException	Thrown when product information fails validation
2	DatabaseConnectionException	Thrown when unable to connect to the database
3	DuplicateProductException	Thrown when attempting to create a product that already exists

Method

How to use parameters / attributes

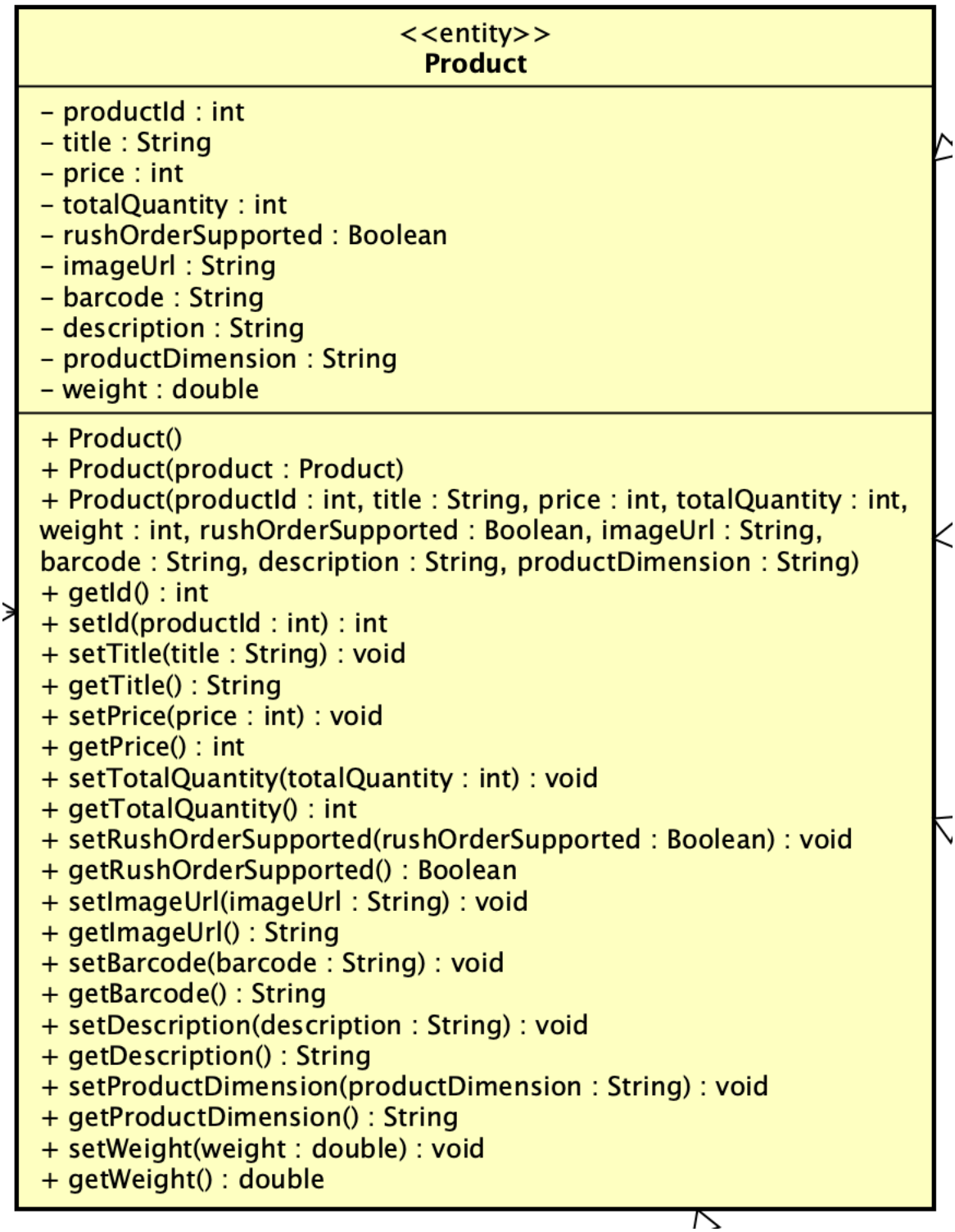
The **productInfo** parameter contains all the necessary information to create a product, including its type (Book, CD, LP, or DVD) and all relevant attributes for that type.
This parameter is used in both the **checkValidityOfProductInfo** method and the **createProduct** method.

Flowchart / activity diagram / sequence diagram if the method has a complex / special algorithm

State

State diagram if any

Design for class "Product"



#	Name	Data type	Default value	Description
1	productId	int	None	Unique identifier for the product
2	title	String	None	Title or name of the product
3	price	int	None	Price of the product
4	totalQuantity	int	None	Total available quantity of the product
5	rushOrderSupported	Boolean	None	Indicates if rush order is supported for this product
6	imageUrl	String	None	URL to the product image
7	barcode	String	None	Barcode number of the product
8	description	String	None	Detailed description of the product
9	productDimension	String	None	Physical dimensions of the product
10	weight	double	None	Weight of the product

Table 1. Example of attribute design

#	Name	Return Type	Description
1	Product()		Default constructor
2	Product(product : Product)		Copy constructor that creates a new Product from an existing one

3	Product(productId, title, price, totalQuantity, weight, rushOrderSupported, imageUrl, barcode, description, productDimension)		Parameterized constructor
4	getId	int	Returns the product ID
5	setId	void	Sets the product ID
6	getTitle	String	Returns the product title
7	setTitle	void	Sets the product title
8	getPrice	int	Returns the product price
9	setPrice	void	Sets the product price
10	getTotalQuantity	int	Returns the total available quantity
11	setTotalQuantity	void	Sets the total available quantity
12	getRushOrderSupported	Boolean	Returns whether rush order is supported
13	setRushOrderSupported	void	Sets whether rush order is supported
14	getImageUrl	String	Returns the product image URL
15	setImageUrl	void	Sets the product image URL
16	getBarcode	String	Returns the product barcode
17	setBarcode	void	Sets the product barcode
18	getDescription	String	Returns the product description
19	setDescription	void	Sets the product description
20	getProductDimension	String	Returns the product dimensions
21	setProductDimension	void	Sets the product dimensions
22	getWeight	double	Returns the product weight
23	setWeight	void	Sets the product weight

Table 2. Example of operation design

Parameter

#	Name	Default value	Description
1	productId	None	Unique identifier for the product
2	title	None	Title or name of the product
3	price	None	Price of the product
4	totalQuantity	None	Total available quantity of the product
5	weight	None	Weight of the product
6	rushOrderSupported	None	Indicates if rush order is supported
7	imageUrl	None	URL to the product image
8	barcode	None	Barcode number of the product
9	description	None	Detailed description of the product
10	productDimension	None	Physical dimensions of the product
11	product	None	Existing Product object to copy from

Exception

#	Name	Description
1	IllegalArgumentException	Thrown when invalid parameter values are provided
2	NullPointerException	Thrown when required values are NULL

Method

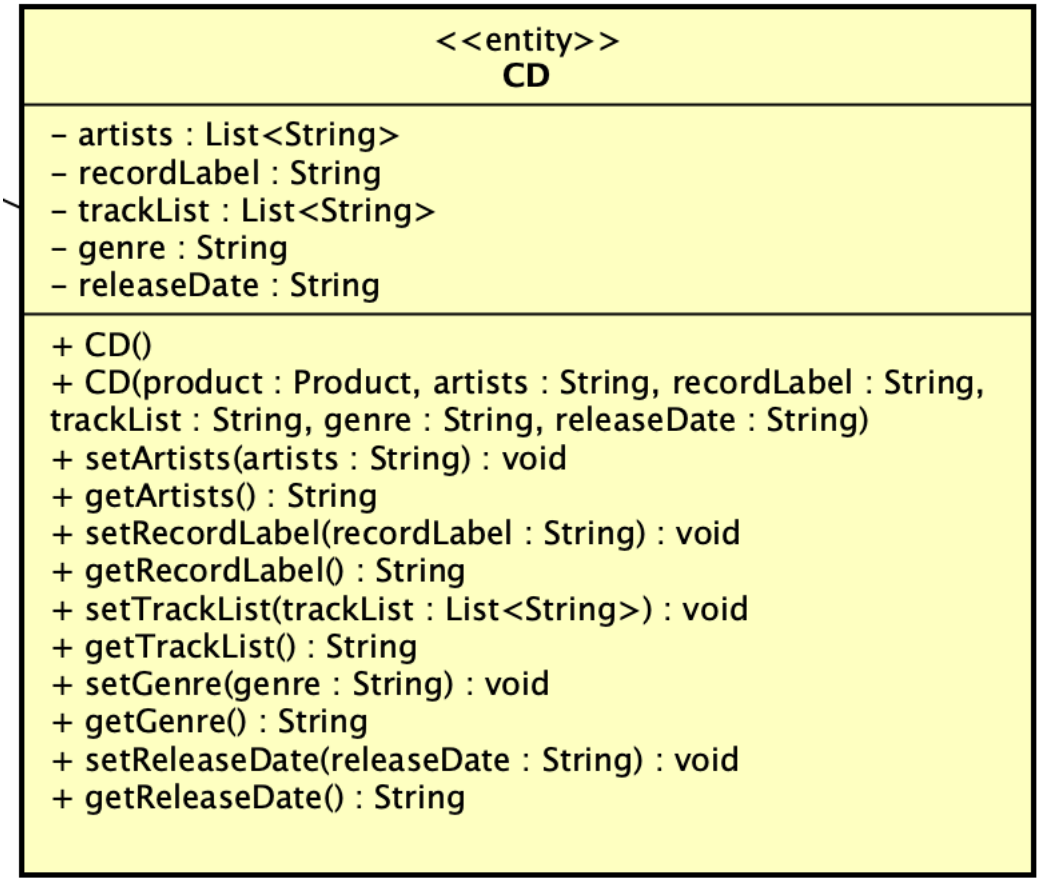
How to use parameters / attributes

Flowchart / activity diagram / sequence diagram if the method has a complex / special algorithm

State

State diagram if any

Design for class “CD”



#	Name	Data type	Default value	Description
1	artists	List<String>	None	List of artists for the CD
2	recordLabel	String	None	Record label that published the CD
3	trackList	List<String>	None	List of tracks on the CD
4	genre	String	None	Genre or musical style of the CD
5	releaseDate	String	None	Release date of the CD

Table 1. Example of attribute design

#	Name	Return type	Description (purpose)
1	CD()		Default constructor
2	CD(product, artists, recordLabel, trackList, genre, releaseDate)		Parameterized constructor
3	setArtists	void	Sets the list of artists
4	getArtists	List<String>	Returns the artists
5	setRecordLabel	void	Sets the record label
6	getRecordLabel	String	Returns the record label
7	setTrackList	void	Sets the track list
8	getTrackList	List<String>	Returns the track list
9	setGenre	void	Sets the genre
10	getGenre	String	Returns the genre
11	setReleaseDate	void	Sets the release date
12	getReleaseDate	String	Returns the release date

Table 2. Example of operation design

Parameter

#	Name	Default value	Description
1	product	None	Product object containing basic product information
2	artists	None	List or string of artist names
3	recordLabel	None	Name of the record label
4	trackList	None	List or string of track names
5	genre	None	Musical genre of the CD

6	releaseDate	None	Date of release

Exception

#	**Name**	**Description**
1	IllegalArgumentException	Thrown when invalid parameter values are provided
2	NullPointerException	Thrown when required values are null

Method

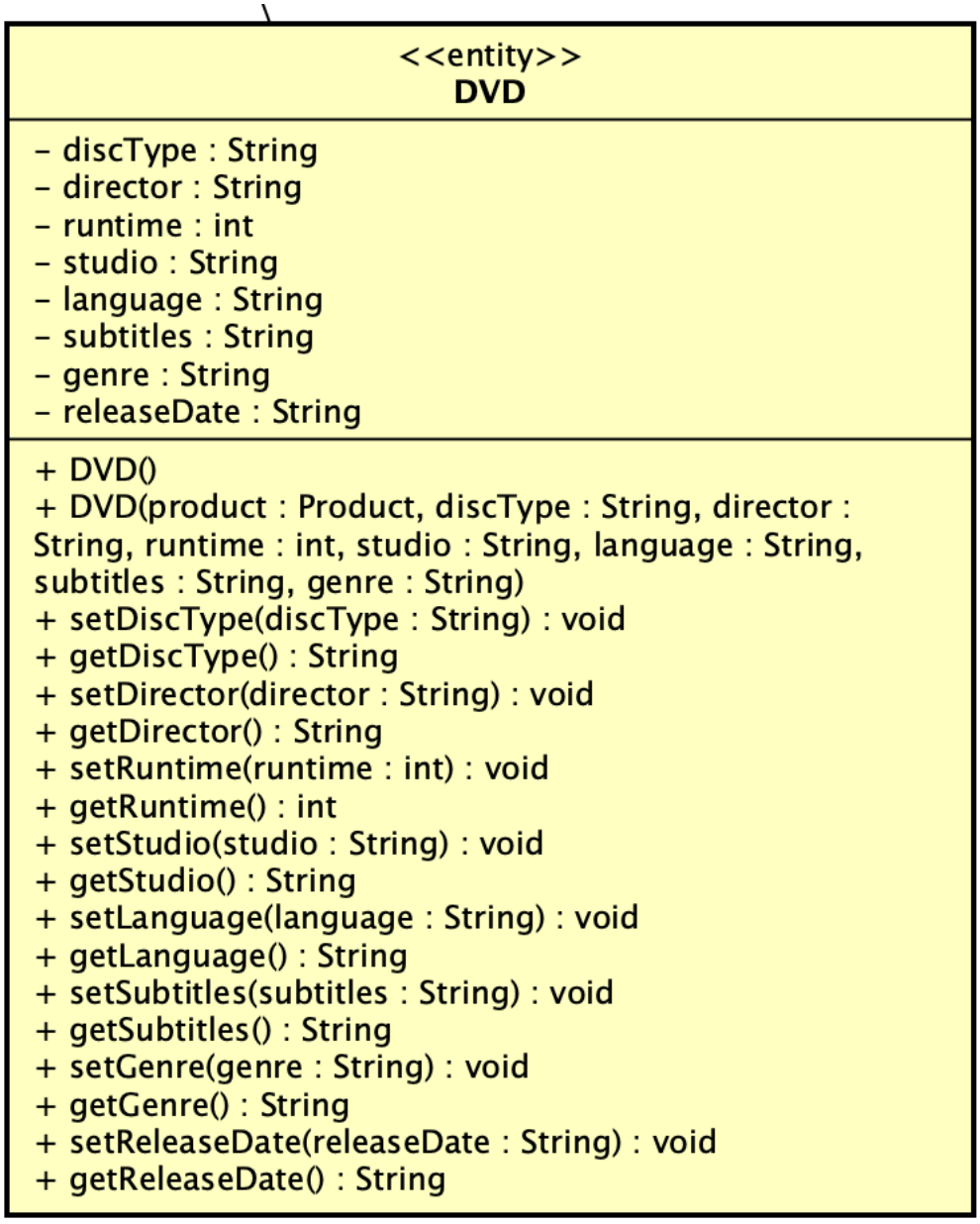
How to use parameters / attributes

Flowchart / activity diagram / sequence diagram if the method has a complex / special algorithm

State

State diagram if any

Design for class "DVD"



#	Name	Data Type	Default value	Description
1	discType	String	None	Type of DVD disc (e.g., Single-layer, Dual-layer)
2	director	String	None	Name of the film director

3	runtime	int	None	Length of the DVD content in minutes
4	studio	String	None	Production studio
5	language	String	None	Primary language of the content
6	subtitles	String	None	Available subtitle languages
7	genre	String	None	Genre or category of the DVD content

Table 1. Example of attribute design

#	Name	Return Type	Description
1	DVD()		Default constructor
2	DVD(product, discType, director, runtime, studio, language, subtitles, genre)		Parameterized constructor
3	setDiscType	void	Sets the disc type
4	getDiscType	String	Returns the disc type
5	setDirector	void	Sets the director
6	getDirector	String	Returns the director
7	setRuntime	void	Sets the runtime
8	getRuntime	int	Returns the runtime
9	setStudio	void	Sets the studio
10	getStudio	String	Returns the studio
11	setLanguage	void	Sets the language
12	getLanguage	String	Returns the language
13	setSubtitles	void	Sets the subtitles
14	getSubtitles	String	Returns the subtitles

Table 2. Example of operation design

Parameter

#	Name	Default Value	Description
1	product	None	Product object containing basic product information
2	discType	None	Type of DVD disc
3	director	None	Name of the director
4	runtime	None	Length of content in minutes
5	studio	None	Name of production studio
6	language	None	Primary language of content
7	subtitles	None	Available subtitle languages
8	genre	None	Genre of the content
9	releaseDate	None	Date of DVD release

Exception

#	Name	Description
1	IllegalArgumentException	Thrown when invalid parameter values are provided
2	NullPointerException	Thrown when required values are NULL

Method

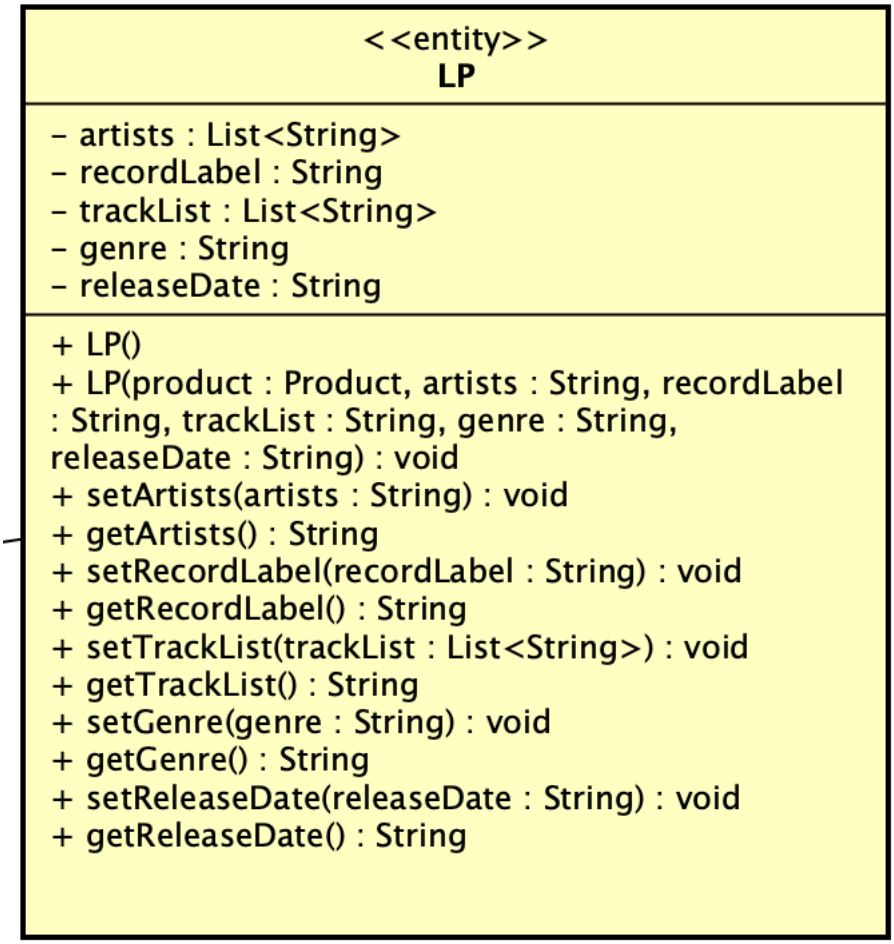
How to use parameters / attributes

Flowchart / activity diagram / sequence diagram if the method has a complex / special algorithm

State

State diagram if any

Design for class “LP”



#	Name	Data type	Default value	Description
1	artists	List<String>	None	List of artists for the LP
2	recordLabel	String	None	Record label that published the LP
3	trackList	List<String>	None	List of tracks on the LP
4	genre	String	None	Genre or musical style of the LP
5	releaseDate	String	None	Release date of the LP

Table 1. Example of attribute design

#	Name	Return type	Description (purpose)
1	CD()		Default constructor
2	CD(product, artists, recordLabel, trackList, genre, releaseDate)		Parameterized constructor
3	setArtists	void	Sets the list of artists
4	getArtists	List<String>	Returns the artists
5	setRecordLabel	void	Sets the record label
6	getRecordLabel	String	Returns the record label
7	setTrackList	void	Sets the track list
8	getTrackList	List<String>	Returns the track list
9	setGenre	void	Sets the genre
10	getGenre	String	Returns the genre
11	setReleaseDate	void	Sets the release date
12	getReleaseDate	String	Returns the release date

Table 2. Example of operation design

Parameter

#	Name	Default value	Description
1	product	None	Product object containing basic product information
2	artists	None	List or string of artist names
3	recordLabel	None	Name of the record label
4	trackList	None	List or string of track names
5	genre	None	Musical genre of the LP

6	releaseDate	None	Date of release

Exception

#	**Name**	**Description**
1	IllegalArgumentException	Thrown when invalid parameter values are provided
2	NullPointerException	Thrown when required values are null

Method

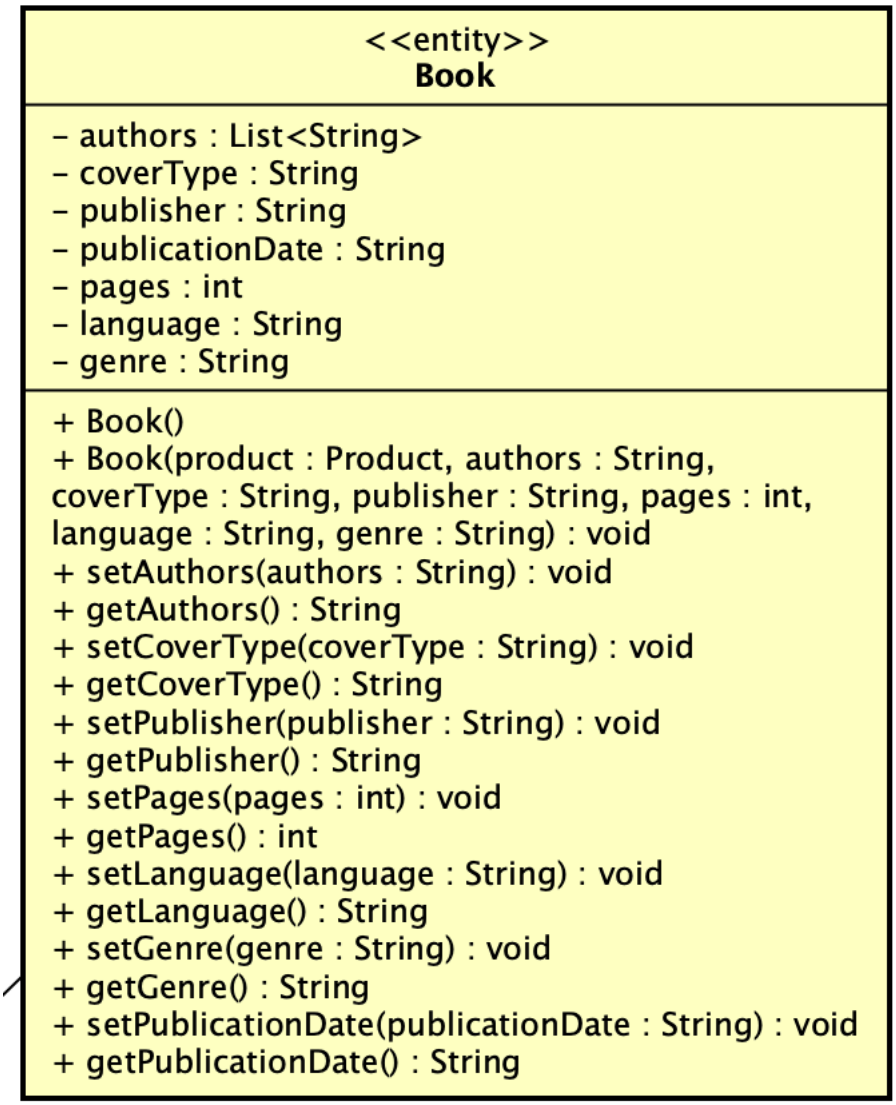
How to use parameters / attributes

Flowchart / activity diagram / sequence diagram if the method has a complex / special algorithm

State

State diagram if any

Design for class "Book"



#	Name	Data Type	Default value	Description
1	authors	List<String>	None	List of author names for the book
2	coverType	String	None	Type of book cover (e.g., hardcover, paperback)
3	publisher	String	None	Name of the book publisher

4	publicationDate	String	None	Publication date of the book
5	pages	int	None	Number of pages in the book
6	language	String	None	Language the book is written in
7	genre	String	None	Genre or category of the book

Table 1. Example of attribute design

#	Name	Return Type	Description
1	Book()		Default constructor
2	Book(product, authors, coverType, publisher, pages, language, genre)		Parameterized constructor
3	setAuthors	void	Sets the list of authors
4	getAuthors	String	Returns the authors
5	setCoverType	void	Sets the cover type
6	getCoverType	String	Returns the cover type
7	setPublisher	void	Sets the publisher
8	getPublisher	String	Returns the publisher
9	setPages	void	Sets the number of pages
10	getPages	int	Returns the number of pages
11	setLanguage	void	Sets the language
12	getLanguage	String	Returns the language
13	setGenre	void	Sets the genre
14	getGenre	String	Returns the genre

15	setPublicationDate	void	Sets the publication date
16	getPublicationDate	String	Returns the publication date

Table 2. Example of operation design

Parameter

#	Name	Default Value	Description
1	product	None	Product object containing basic product information
2	authors	None	List or string of author names
3	coverType	None	Type of book cover
4	publisher	None	Name of publisher
5	pages	None	Number of pages
6	language	None	Language of the book
7	genre	None	Genre/category of the book
8	publicationDate	None	Date of publication

Exception

#	Name	Description
1	IllegalArgumentException	Thrown when invalid parameter values are provided
2	NullPointerException	Thrown when required values are null

Method

How to use parameters / attributes

Flowchart / activity diagram / sequence diagram if the method has a complex / special algorithm

State

State diagram if any