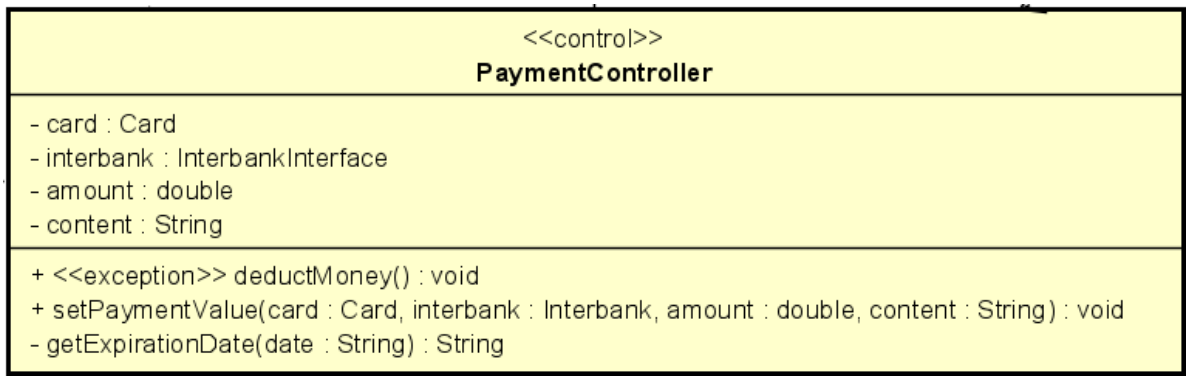
CLASS DESIGN

1. Class “PaymentController”



Attribute

#	Name	Data type	Default value	Description
1	card	Card	NULL	Represent the card used for payment
2	amount	double	NULL	Represent total amount of the transaction
3	content	String	NULL	Represent content of transaction
4	interbank	InterbankInterface	NULL	Represent the interbank

Operation

#	Name	Return type	Description (purpose)
1	deductMoney	void	Send request to interbank API to deduct money from card to pay for renting bike
2	setPaymentValue	void	Set value for all attributes of *PaymentController* class

Parameter:

- card – the credit card used for payment
- interbank – interbank of transaction
- amount – total amount of transaction
- content – content of transaction

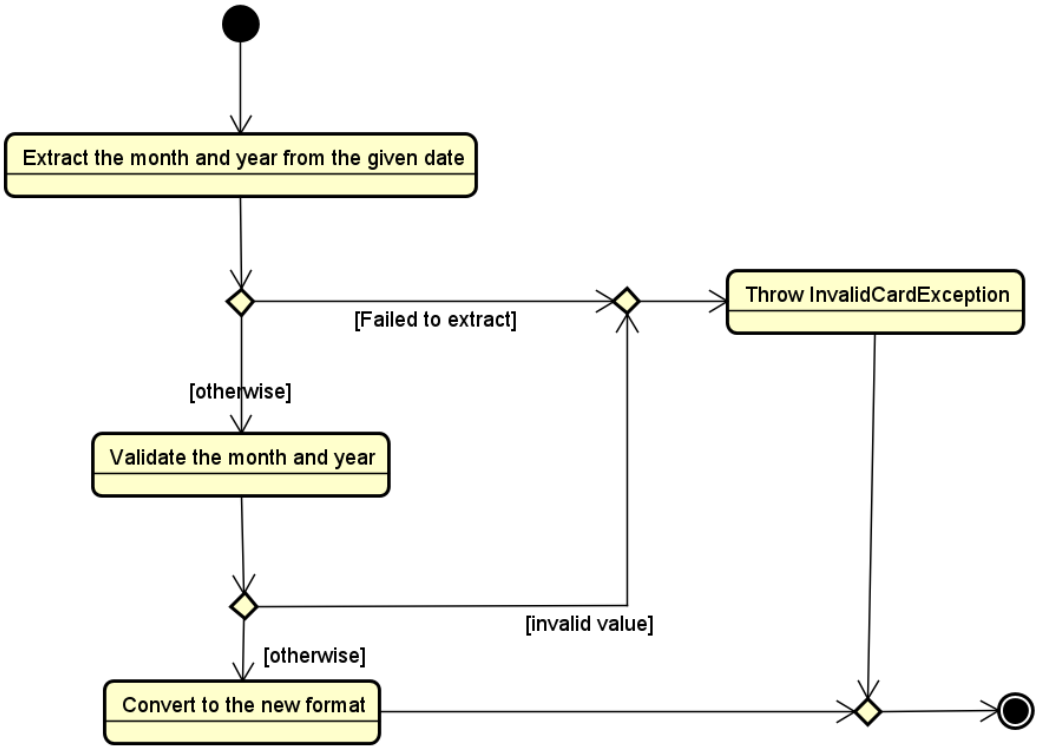
Exception:

- PaymentException: if responded with error that transaction is failed

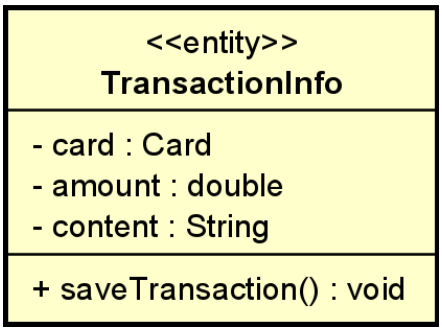
State None

Method

- getExpirationDate: given the String “date” representing the expiration date in the format “”mm/yy”, this method convert it into the required format “mmyy”. The algorithm is illustrated as follows.



2. Class "TransactionInfo"



Attribute

#	Name	Data type	Default value	Description
1	card	Card	NULL	Represent the card used for payment
2	amount	double	NULL	Represent total amount of the transaction
3	content	String	NULL	Represent content of transaction

Operation

#	Name	Return type	Description (purpose)
1	saveTransaction	void	Save the transaction between software and interbank

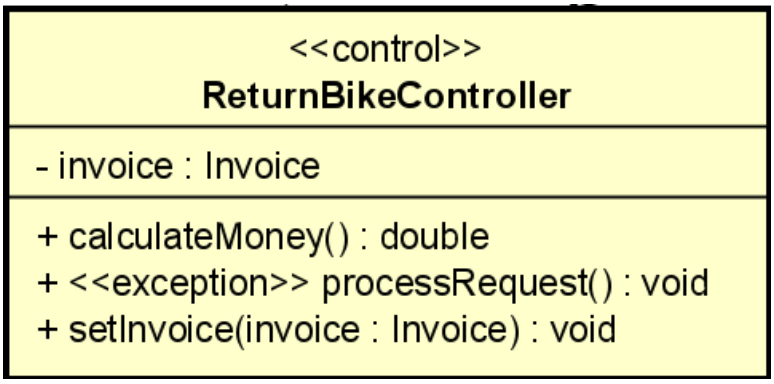
Parameter: None

Exception: None

Method None

State None

3. Class "ReturnBikeController"



Attribute

#	Name	Data type	Default value	Description
1	invoice	Invoice	NULL	Represent invoice of returning bike

Operation

#	Name	Return type	Description (purpose)
1	calculateMoney	double	Calculate total amount that customer has to pay when return bike
2	processRequest	void	Process returning bike request, call *ReturnBikeHandler* and *InvoiceHandler* class
3	setInvoice	void	Set value for invoice after calculate money

Parameter:

- Invoice – invoice of returning bike

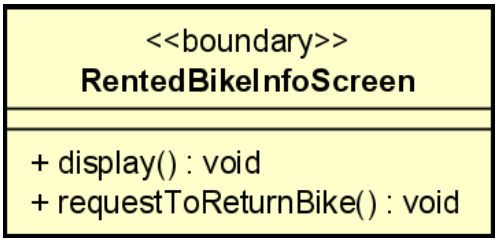
Exception:

- ReturnBikeException: if responded with error that return bike request is failed

State None

Method None

4. Class "RentedBikeInfoScreen"



Attribute

Operation

#	Name	Return type	Description (purpose)
1	display	void	Display detail of renting bike include amount up to now that customer has to pay
2	requestToReturnBike	void	Request to return bike

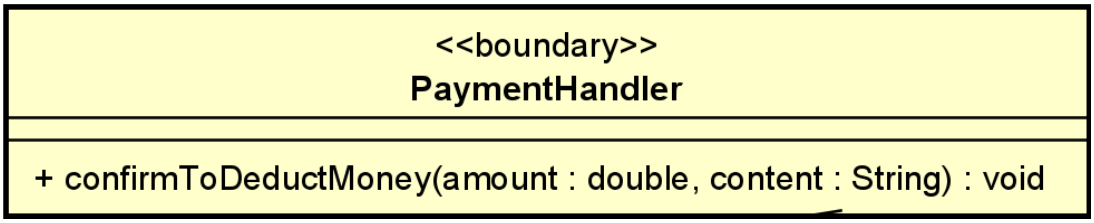
Parameter: None

Exception: None

Method None

State None

5. Class "PaymentHandler"



Attribute

Operation

#	Name	Return type	Description (purpose)
1	confirmToDeductMoney	void	Confirm to deduct money from card to pay for renting bike

Parameter:

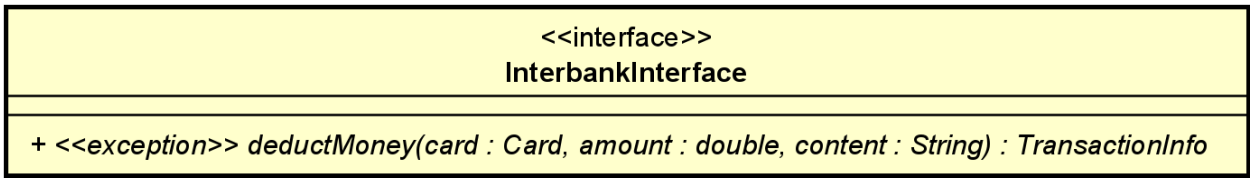
- amount: total amount that interbank will deduct from customer's card
- content: content of transaction

Exception: None

Method None

State None

6. Class "InterbankInterface"



Attribute

Operation

#	Name	Return type	Description (purpose)
1	deductMoney	void	Confirm to deduct money from card to pay for renting bike

Parameter:

- amount: total amount that interbank will deduct from customer's card
- content: content of transaction
- card : the credit card used for payment

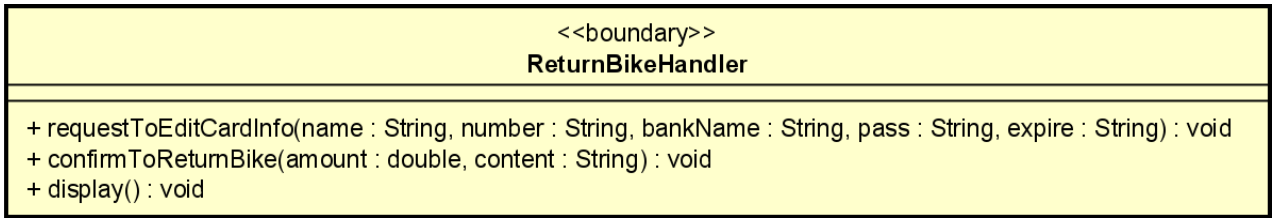
Exception:

- InterbankPaymentException: if responded with a pre-defined error code
- UnrecognizeException: if responded with an unknown error code or something goes wrong

Method None

State None

7. Class "ReturnBikeHandler"



Attribute

Operation

#	Name	Return type	Description (purpose)
1	requestToEditCardInfo	void	Edit card information
2	confirmToReturnBike	void	Confirm to return bike
3	display	void	Display detailed information of return bike and card information of customer

Parameter:

- amount: total amount that customer has to pay for returning bike

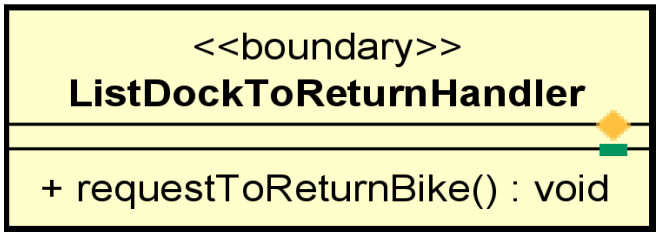
- content: content of transaction
- name: card's holder name
- number: card number
- bankName: name of interbank
- pass: security code of card
- expire: expiration date of card

Exception: None

Method None

State None

8. Class "ListDockToReturnHandler"



Attribute

Operation

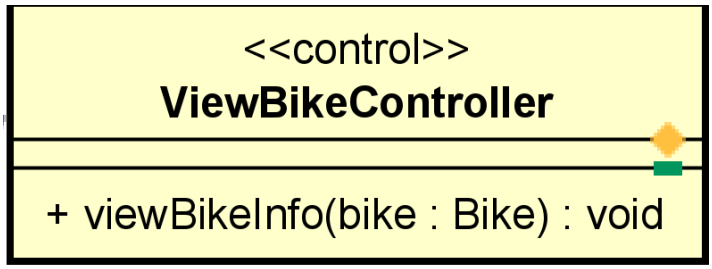
#	Name	Return type	Description (purpose)
1	requestToReturnBike	void	Request to return bike, call selectDockMakerController

Parameter none

Method none

State none

9. Class "ViewBikeController"



Attribute none

Operation

#	Name	Return type	Description (purpose)
1	viewBikeInfo	void	Process request view specific bike info

Parameter

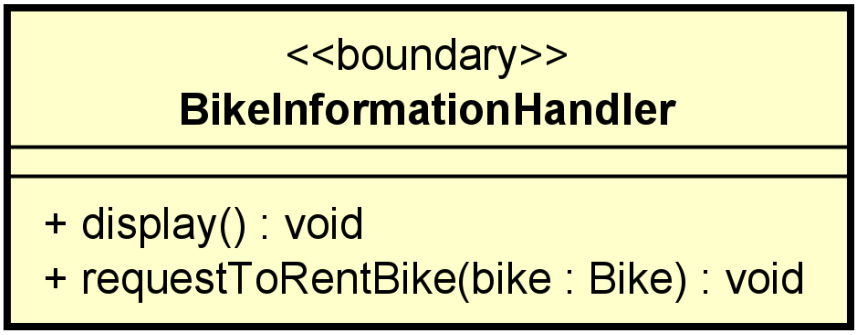
- bike: bike that want to view information

Exception : ViewBikeException

Method none

State none

10. Class "BikeInformationHandler"



Attribute none

Operation

#	Name	Return type	Description (purpose)
1	display	void	Display bike information
2	requestToRentBike	Void	When user submit to rent bike, sent request to rentBikeController

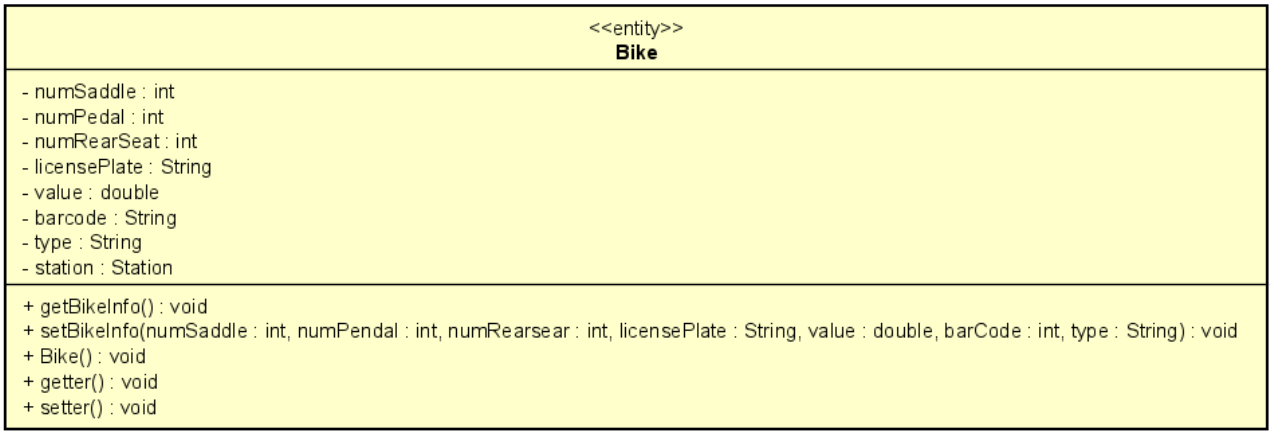
Parameter

- bike: bike that user want to rent

Method none

State none

11. Class "Bike"



Attribute

#	Name	Data type	Default value	Description
1	numSaddle	int	NULL	Number saddle of the bike
2	numPedal	int	NULL	Number pedal of the bike
3	numRearSeat	Int	NULL	Number rear seat of the bike
4	licensePlate	String	NULL	Represent license plate of the bike

5	value	double	NULL	Represent value of the bike
6	barcode	String	NULL	Represent barcode of the bike
7	type	String	NULL	Represent type of the bike
8	station	Station	NULL	Represent bike in which station

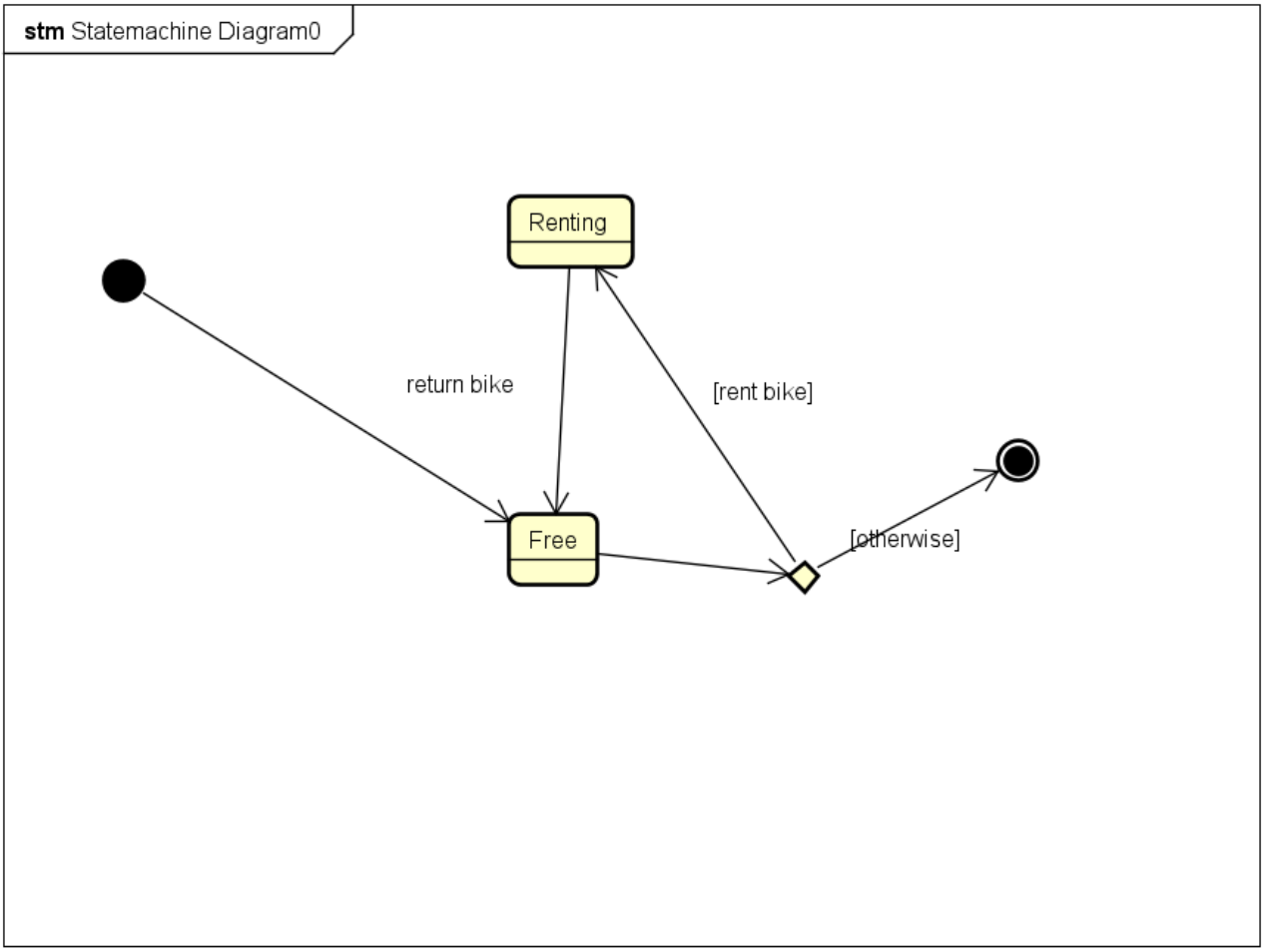
Operation

#	Name	Return type	Description (purpose)
1	getBikeInfo	void	Get bike information for display
2	setBikeInfo	void	Set bike information
3	Bike	void	Constructor
4	getter	void	Get all attribute in acronym
5	setter	void	Set value for each attribute in acronym

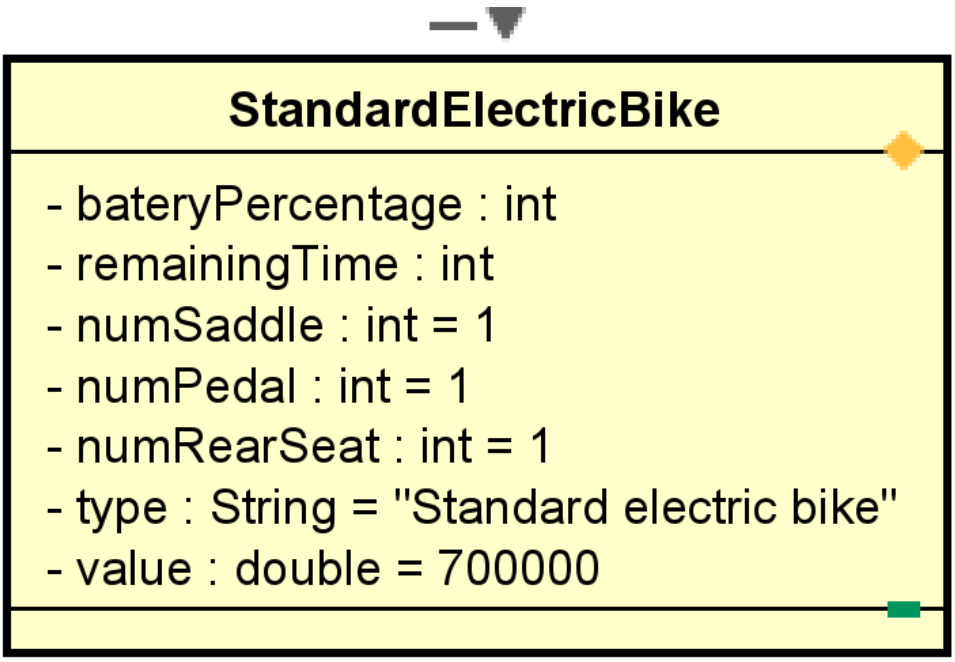
Parameter : same like attribute

Method none

State



12. Class “StandardElectricBike”

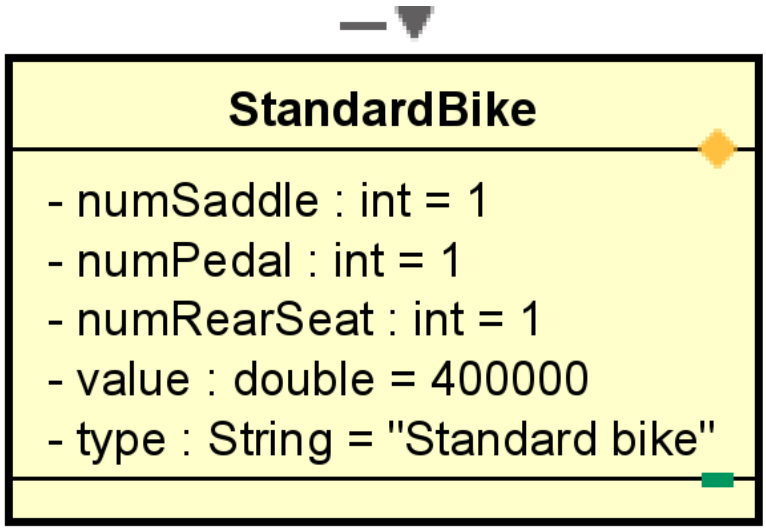


Attribute

#	Name	Data type	Default value	Description
1	numSaddle	int	1	Unchanged value
2	numPedal	int	1	Constant value
3	numRearSeat	int	1	Constant value
4	value	double	700000	Constant value
5	type	String	“Standard electric bike”	Constant type

Operation inherit
Method none
State none

13. Class “Standard bike”

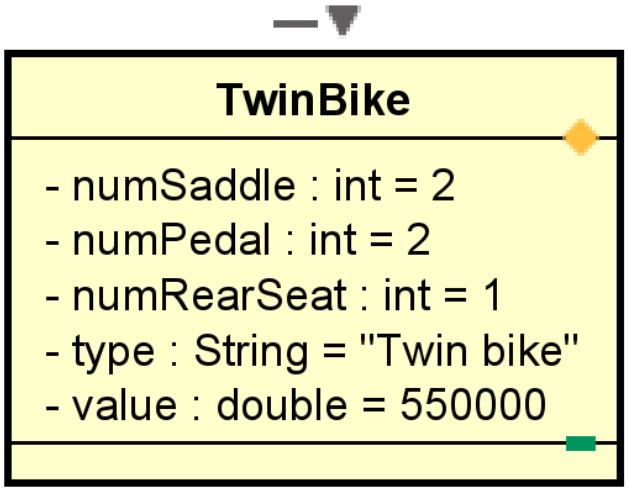


Attribute

#	Name	Data type	Default value	Description
1	numSaddle	int	1	Unchanged value
2	numPedal	int	1	Constant value
3	numRearSeat	int	1	Constant value
4	value	double	400000	Constant value
5	type	String	“Standard bike”	Constant type

Operation inherit
Method none
State none

14. Class “TwinBike”

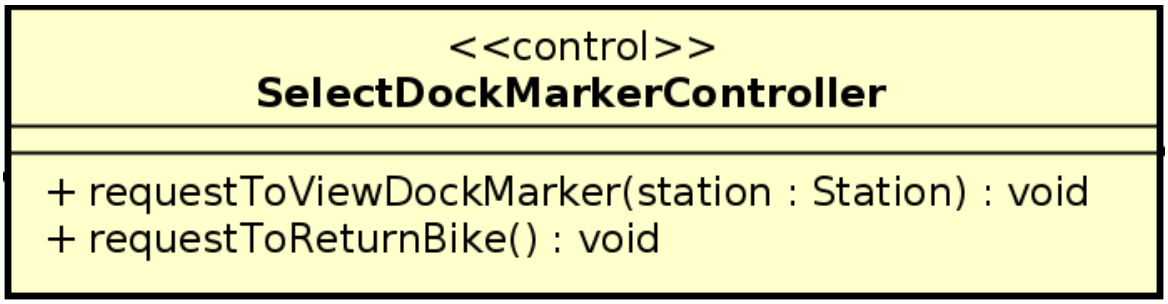


Attribute

#	Name	Data type	Default value	Description
1	numSaddle	int	2	Unchanged value
2	numPedal	int	2	Constant value
3	numRearSeat	int	1	Constant value
4	value	double	550000	Constant value
5	type	String	“Twin bike”	Constant type

Operation inherit
Method none
State none

15. Class “SelectDockMarkerController”



Attribute none

Operation

#	Name	Return type	Description (purpose)
1	requestToViewDockMarker	void	Process request to view the dock marker
2	RequestToReturnBike	void	Process to request to return the bike

Parameter

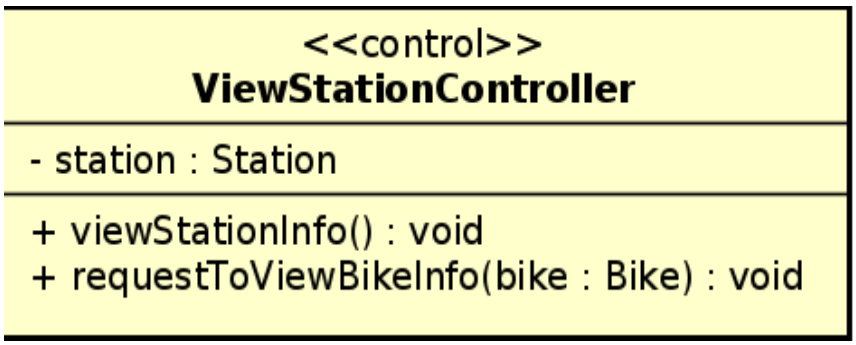
- station: station that want to view information

Exception : ViewStationException

Method none

State none

16. Class “ViewStationController”



Attribute station: station that user want to view information

Operation

#	Name	Return type	Description (purpose)
1	viewStationInfo	void	View station information
2	requestToViewBikeInfo	Void	Process to request to view bike information

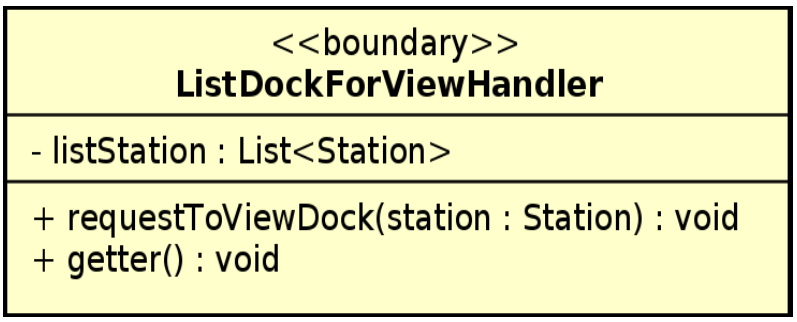
Parameter

- bike: bike that user want to rent

Method none

State none

17. Class "ListDockForViewHandler"



Attribute listStation: a list of stations

Operation

#	Name	Return type	Description (purpose)
1	requestToViewDock	void	Process to request to view dock
2	getter	Void	get method

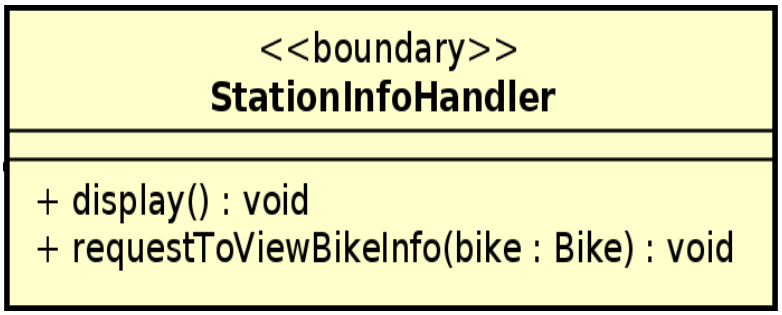
Parameter

- station: station that user want to view information

Method none

State none

18. Class "StationInfoHandler"



Attribute none

Operation

#	Name	Return type	Description (purpose)
1	display	void	Display station information
2	requestToViewBikeInfo	Void	Process to request to view bike information

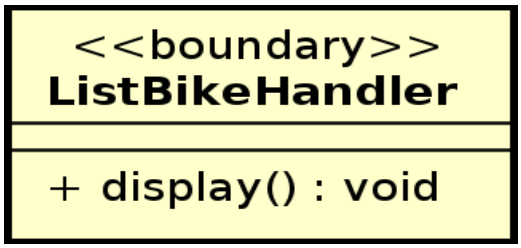
Parameter

- bike: bike that user want to view information

Method none

State none

19. Class “ListBikeHandler”



Attribute none

Operation

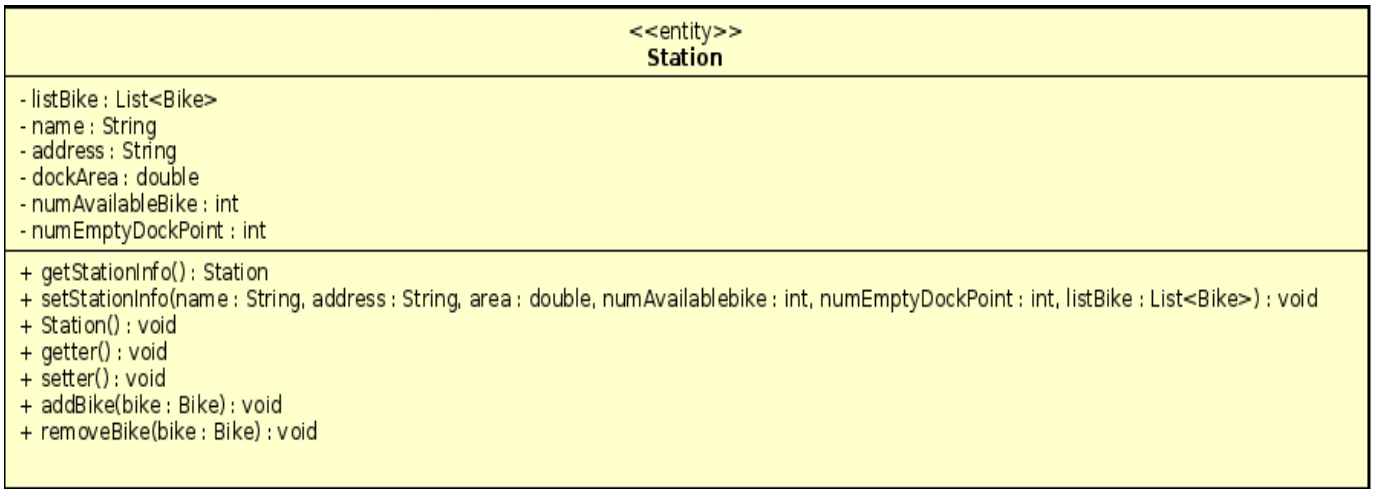
#	Name	Return type	Description (purpose)
1	display	void	Display list of bikes

Parameter none

Method none

State none

20. Class “Station”



Attribute

#	Name	Data type	Default value	Description
1	listBike	List<Bike>	NULL	List the bikes in the station
2	name	String	NULL	Name of the station
3	address	String	NULL	Address of the station
4	dockArea	double	NULL	Area of the dock
5	numAvailableBike	int	NULL	Number of available bikes
6	numEmptyDockPoint	int	NULL	Number of empty dock points

Operation

#	Name	Return type	Description (purpose)

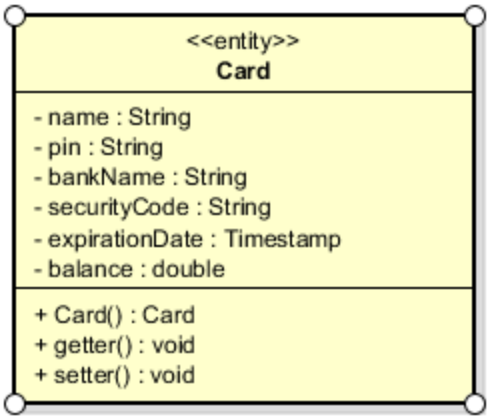
1	getStationInfo	Station	Get station information for display
2	setStationInfo	void	Set station information
3	Station	void	Constructor
4	getter	void	Get all attribute in acronym
5	setter	void	Set value for each attribute in acronym
6	addBike	void	Add bike to the station if user return bike
7	removeBike	void	Remove bike in the station if user rent bike

Parameter : same like attribute

Method none

State none

21. Class "Card"



Attribute

#	Name	Data type	Default value	Description
1	name	String	NULL	Name of the owner of the credit card
2	pin	String	NULL	Pin code of the credit card
3	bankName	String	NULL	The name of the bank that provides the credit card
4	securityCode	String	NULL	Security code of the credit card
5	expirationDate	Timestamp	NULL	The expiration date of the card, in form MM/YYYY
6	Balance	double	0.00	The balance of the card

Operation

#	Name	Return type	Description (purpose)
1	Card	Card	Constructor
2	getter	void	Get all attribute in acronym
3	setter	void	Set value for each attribute in acronym

Parameter: same as attributes

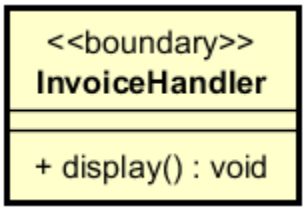
State

None

Method

None

22. Class "InvoiceHandler"



Attribute

None

Operation

#	Name	Return type	Description (purpose)
1	display	void	Display the invoice screen

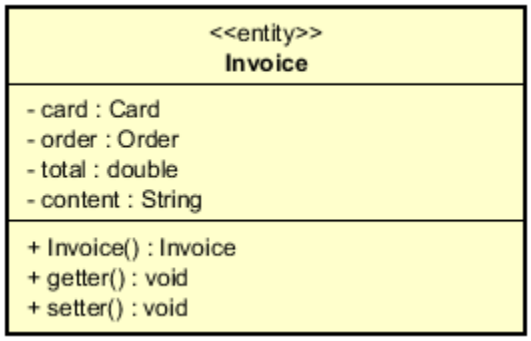
State

None

Method

None

23. Class "Invoice"



Attribute

#	Name	Data type	Default value	Description
1	card	Card	NULL	The credit card that was used for this transaction/invoice
2	order	Order	NULL	The order that was used for this transaction/invoice
3	total	double	0.00	The amount of money that was transferred in the transaction/The amount of money that was deducted from the credit card in the transaction
4	content	String	NULL	The details of the transaction/invoice

Operation

#	Name	Return type	Description (purpose)
1	Invoice	Invoice	Constructor
2	getter	void	Get all attribute in acronym
3	setter	void	Set value for each attribute in acronym

Parameter: same as attributes

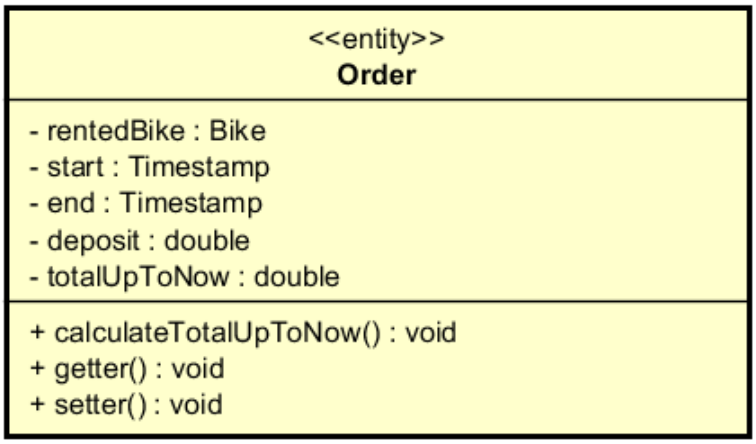
State

None

Method

None

24. Class "Order"



Attribute

#	Name	Data type	Default value	Description
1	rentedBike	Bike	NULL	The bike that was rented by user
2	start	Timestamp	Time when the bike was rented	The timestamp which user rented the bike
3	end	Timestamp	Current Time	The timestamp which user returns bike (in case this order is for returning bike), or current time (in case this order is for renting bike)
4	deposit	double	0.00	The deposit when renting the bike
5	totalUpToNow	double	0.00	Total renting money up to now (not include deposit)

Operation

#	Name	Return type	Description (purpose)
1	calculateTotalUpToNow	void	Calculate the renting amount up to now (not include deposit)
2	getter	void	Get all attribute in acronym
3	setter	void	Set value for each attribute in acronym

Parameter: same as attributes

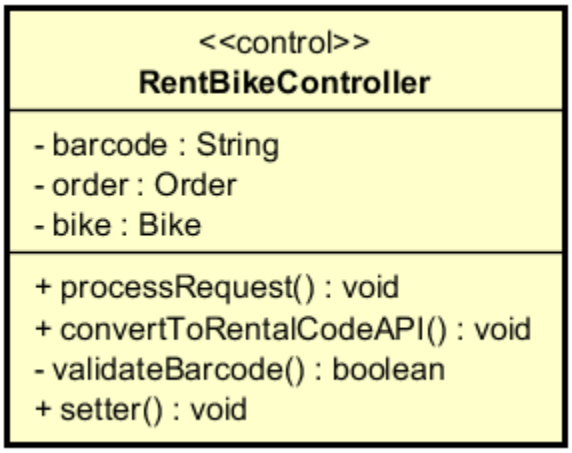
State

None

Method

None

25. Class “RentBikeController”



Attribute

#	Name	Data type	Default value	Description
1	barcode	String	NULL	The barcode of the rented bike
2	order	Order	NULL	The order is made when renting bike
3	bike	Bike	NULL	The rented bike

Operation

#	Name	Return type	Description (purpose)
1	processRequest	void	Process the request of renting bike, to see that if the request comes from user entering the barcode or user choosing the bike
2	validateBarcode	boolean	In case the user entering the barcode (the barcode attribute is not empty), then we should check if the barcode is valid. If it is, call the setter of the bike attribute. If it is not, an error is displayed in the barcode screen.
3	convertToRentalCodeAPI	void	Call the API to convert the barcode into rental code
4	setter	void	Set value for each attribute in acronym

Parameter:

None

Exception:

- RentBikeException: if responded with error that rent bike request is failed

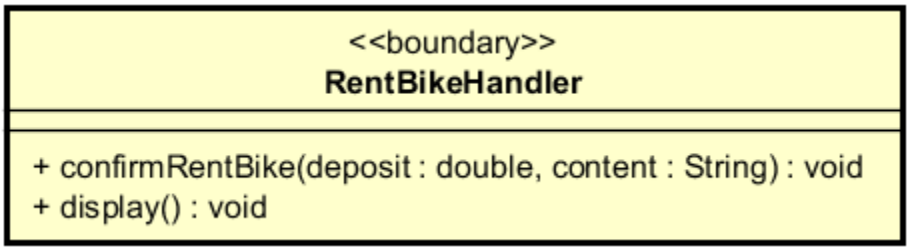
State

None

Method

None

26. Class "RentBikeHandler"



Attribute

None

Operation

#	Name	Return type	Description (purpose)
1	display	void	Display the rent bike screen
2	confirmRentBike	void	Display the payment screen, send the deposit amount and the transaction content to the payment controller

Parameter:

- deposit - the deposit amount that user has to pay when renting bike
- content - the details of the transaction when sending to payment

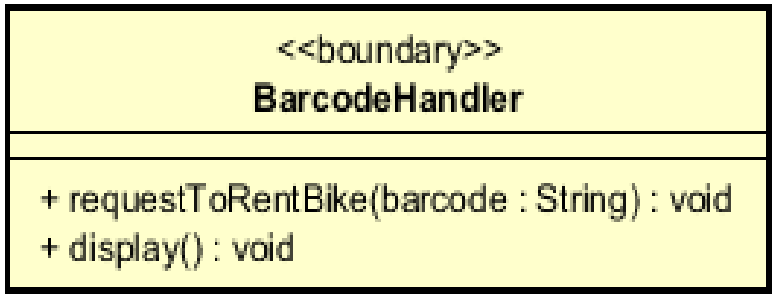
State

None

Method

None

27. Class "BarcodeHandler"



Attribute

None

Operation

#	Name	Return type	Description (purpose)
1	display	void	Display the barcode screen
2	requestToRentBike	void	After the user inputs the barcode, this function sends the barcode to controller to process the rent-bike request

Parameter:

- barcode: the code that user inputs when he/she wants to rent bike

State

None

Method

None