MemoSet class reference [1]

Sebastian 'spax' Pape

December 5th, 2002

[1]thanks to Till Schümmer and Peter Tandler for their support

Contents

1 MemoSetApplicationModel

inherits from: `CoastApplicationModel` and provides some methods for accessing the MemoSetGame.

slots:

game (type: singleValue) - just a reference to the actual MemoSetGame instance

1.1 methods

1.1.1 class

instance creation

newInCluster: aCluster cardType: aMemoSetCardSubclass calls self newInCluster: aCluster which is inherited from the CoastApplicationModel and requires aMemoSetCardSubclass which is used to determine the game type. This method actually creates the game instance.

1.1.2 instance

accessing

game a simple access method

game: aMemoSetGame a simple access method

views

localApplicationModelClass returns the local application model class

2 MemoSetCard

inherits from: `CoastModel` and is only an abstract class. It implements some general methods which are used for selecting and deselecting cards.

slots

container (type: singleValue) - the card should know who owns it. Container is a MemoSetUser if the card is taken or a MemoSetGame if it is still in play.

selected (type: set) - the card should know by which users it is selected.

2.1 methods

2.1.1 class

constants

numberOfCardsNeededForSet This method should return an Integer value how many cards are needed for a complete set. Due to the reason `MemoSetCard` is an abstract class, it is the subclasses' responsibility to implement this method.

isGameCardClass should return `true` if the class is a "real" card class

instance creation

createSetWithAllCards This is not a normal implementation of instance creation, because you do not need only one card and you don't know which cards really exist outside. So here the method returns all possible cards in a set.Due to the reason `MemoSetCard` is an abstract class, it is the subclasses' responsibility to implement this method.

2.1.2 instance

accessing

clickedByUser select or deselect the card and test if the user has marked a set. Do some simple verification.

container a simple acces method to the container slot

container: aGameOrUser a simple acces method to the container slot

delesectByAllUsers deselects the card by all users

numberOfCardsNeededForSet just an interface to ask the class

scoreForSet: aSet should return the score for a given set. Due to the reason `MemoSetCard` is an abstract class, it is the subclasses' responsibility to implement this method.

selectedBy returns a Collection with the users which currently selected this card

default

defaultViewClass return the view class for this kind of card. Due to the reason `MemoSetCard` is an abstract class, it is the subclasses' responsibility to implement this method.

testing

isASet: aList The method gets a list with `self class numberOfCardsNeededForSet` (see 2.1.1) elements and decide if it is a valid set. Due to the reason `MemoSetCard` is an abstract class, it is the subclasses' responsibility to implement this method.

isSelected returns true if `self selectedBy` is empty (see 2.1.2).

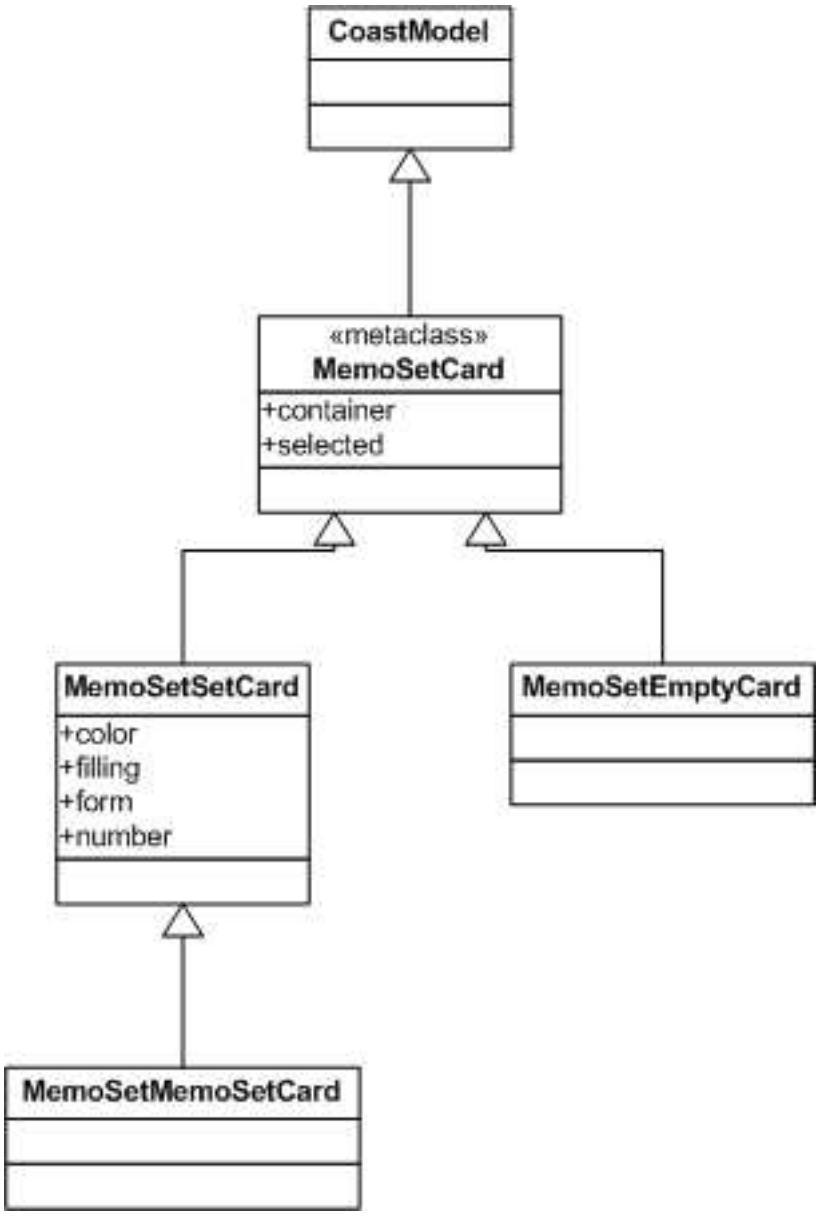
private accessing

deselectByUser: aUser deselects the card for a given `MemoSetUser`.

selectByUser: aUser selects the card for a given `MemoSetUser`.

2.2 subclasses

2.2.1 overview



2.2.2 MemoSetEmptyCard

inherits from: `MemoSetCard` and is a template for empty card slots. Therefore it isn't a GameCardClass, too.

slots This class owns only inherited slots.

class methods

numberOfCardsNeededForSet(constants) An empty card does not need any other cards for a set: Therefore return always: `0`.

createSetWithAllCards(instance creation) Return a set with exactly one empty card.

instance methods

clickedByUser: aUser (accessing) just do nothing. An empty card can not be selected.

scoreForSet: aSet (accessing) an empty card scores `0`.

defaultViewClass(default) returns the `MemoSetEmptyCardView class`.

isASet: aList (testing) An empty card is never a set. Therefore return always: `false`.

2.2.3 MemoSetSetCard

inherits from: `MemoSetCard` and implements the original Set cards.

slots: This class owns additional slots which characterize the cards.

color (type: singleValue) - the color of the card's symbols (red, green or blue)

filling (type: singleValue) - the filling of the card's symbols (empty, half or full)

form (type: singleValue) - the form of the card's symbols (ellipse, rectangle ot triangle)

number (type: singleValue) - the number of the card's symbol (one, two or three)

class methods

numberOfCardsNeededForSet (constants) A card of this set does need two other cards for a set: Therefore return always: `3`.

isGameCardClass (constants) we are a real GameCardClass therefore return `true`.

createSetWithAllCards (instance creation) Create a set of all possible cards. Let's see: We have four properties and each has three possible values. That's a total of 3^4=72 cards.

colors (private) returns a Set with all possible ColorValues of the elements on the cards.

fillings (private) returns a Set with all possible fillings (asString) of the elements on the cards.

forms (private) returns a Set with all possible forms (asString) of the elements on the cards.

numbers (private) returns a Set with all possible numbers of the elements on the cards.

color: aColor filling: aFilling form: aForm number: aNumber (private instance creation) creates a card with the given properties.

instance methods

color is a simple access method to the color slot.

filling is a simple access method to the filling slot.

form is a simple access method to the form slot.

number is a simple access method to the number slot.

scoreForSet: aSet returns the score for a Set. Each difference of a property counts as one point. You can find more information at the games help file.

defaultViewClass (default) returns the `MemoSetSetCardView class`.

isASet: aList (testing) We need to test for each property if it is the same or different at all three cards. Only if this is true for all four properties the cards are a set and we return `true`.

propertiesAreEqualOrAllDifferentTo: first and: second and: third (private testing) is getting three properties and returns if they are alle the same or all different.

2.2.4 MemoSetMemoSetCard

inherits from: `MemoSetSetCard` and implements the "Memory Set" cards.

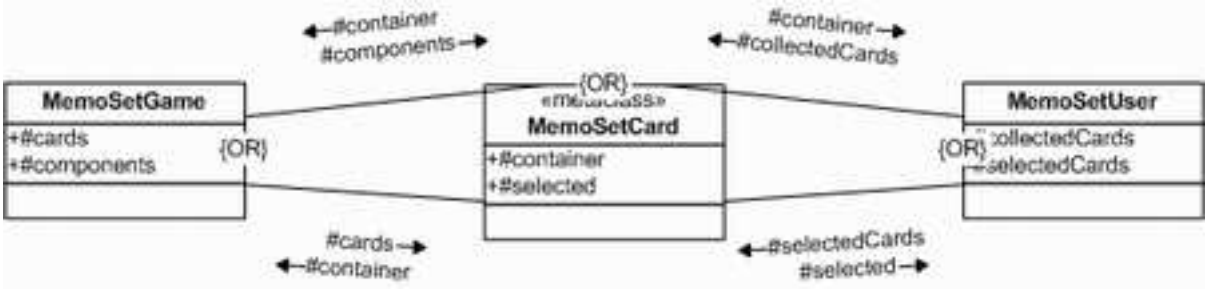
slots This class owns only inherited slots.

instance methods

defaultViewClass (default) returns the `MemoSetMemoSetCardView class`.

2.3 relations

2.3.1 to MemoSetGame and MemoSetUser



3 MemoSetCardController

inherits from: CoastController

3.1 methods

3.1.1 instance

events

redButonPressedEvent: inform the model (a subclass of MemoSetCard) of the controller that it has been clicked on.

4 MemoSetCardView

inherits from: `CoastView`

4.1 methods

4.1.1 class

constants

borderSize returns the size of the border used for awareness to show the user which other users are curently selecting the card.

cardBackground returns the color of the cardBackground used if the card is shown but not selected by the user.

cardSize returns the size of the card.

size returns the size of the card including the border at all four sides.

resources

unselectedCardImage an image of the back of the card.

4.1.2 instance

accessing

composeBounds returns the proper bounds including the border.

defaultControllerClass returns the `MemoSetCardController class`.

displaying

displayCardBackgroundOn: aGraphicsContext displays the background of a card.

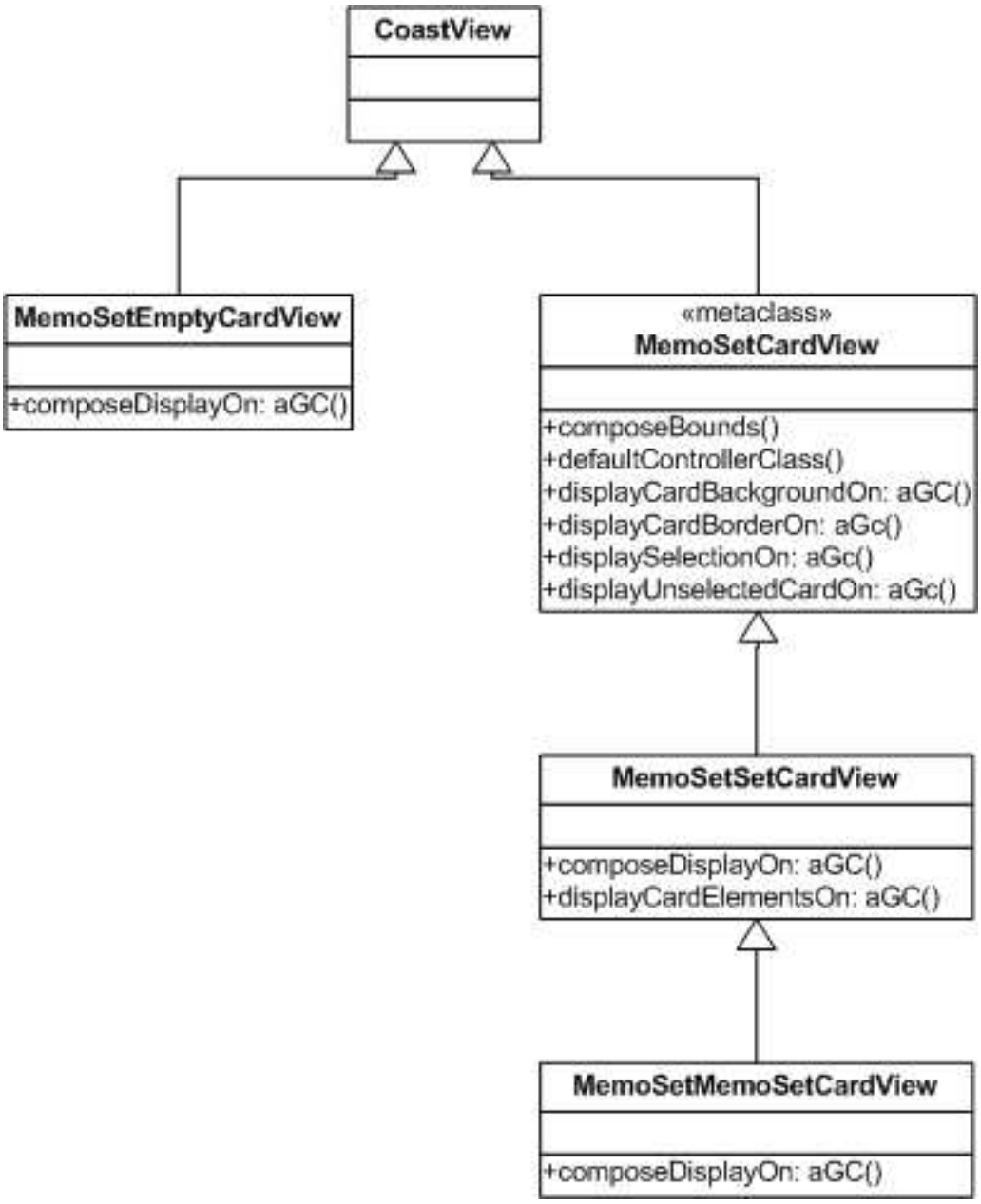
displayCardBorderOn: aGraphicsContext displays the border of a card.

displaySelectionOn: aGraphicsContext displays the selection of a card into the card's border.

displayUnselectedCardOn: aGraphicsContext displays the back of a card.

4.2 subclasses

4.2.1 overview



4.2.2 MemoSetEmptyCardView

inherits from: `CoastView`

instance methods

displayOn: aGraphicsContext do not display anything.

4.2.3 MemoSetSetCardView

inherits from: `MemoSetCardView`

class methods

objectSize (constants) returns the proper object size of the elements (rectangle, ellipse, triangle).

padding (constants) returns the proper disctance between the elements and the border of the card.

instance methods

composeDisplayOn: aGraphicsContext (displaying) displays the card.

displayCardElementsOn: aGraphicsContext (displaying) is the sub-method which actually draws the elements by using the methods mentioned at the private protocoll.

ellipseAt: aPoint(private) is used for drawing ellipsoids.

emptyPattern(private) returns an empty pattern.

fullPattern(private) returns a full pattern.

halfPattern(private) returns a half filled pattern.

p1(private) returns the point at which one object is displayed.

p2(private) returns two points where two objects are displayed.

p3(private) returns three points where three objects are displayed.

rectangleAt: aPoint(private) is used for drawing rectangles.

triangleAt: aPoint(private) is used for drawing triangles.

4.2.4 MemoSetMemoSetCardView

inherits from: `MemoSetSetCardView`

instance methods

composeDisplayOn: aGraphicsContext (displaying) displays the card by calling the super method if the card is selected or otherwise drawing the back of a card.

5 MemoSetClient

inherits from: `CoastApplicationClientWithUI`

5.1 methods

5.1.1 instance

user management

defaultUserClass sets the defaultUserClass to MemoSetUser

6 MemoSetGame

inherits from: `CoastModel` and represents the game. Here cards will be taken from the pile and placed on the table.

slots:

cards (type: set) - the cards "on the pile".

components (type: dictionary) - the cards "on the table".

6.1 methods

6.1.1 class

instance creation

newGame: aMemoSetCardSubclass actually creates the game, gets all cards from the MemoSetCardSubclass and places some of them "on the table".

6.1.2 instance

accessing

cardsOnPile returns the number of cards on the pile.

components returns all cards which are currently on the table.

gameName returns the gameName.

action

collectCardsForUser: aUser collects a set from the table an gives it to the user.

getNewCard takes a card from the pile. If the pile is empty it creates a emptyCard.

7 MemoSetGameView

inherits from: `CoastCompositeView`

7.1 methods

7.1.1 instance

composing

composeComponents takes all cards on the table and translates their position from the dictionary to coordinates.

8 MemoSetLauncher

inherits from: `UI.ApplicationModel`

instance variables

server - a valueholder for the server inputfield

username - a valueholder for the username inputfield

usercolor - the actual color of the user

8.1 methods

8.1.1 class

accessing

localServerIsRunning returns a Boolean if the Coast Mediator is started.

interface specs

windowSpecs specifies the userinterface for the launcher.

interface opening

open opens a new launcher instance.

resources

mediatorLabel returns the string for the 'Start/Stop server' Button depending on localServerIsRunning.

memoSetLogo contains the background logo for the launcher

menu calls MemoSetMenu

windowIcon contains the windowIcon

windowIconMask contains the corresponding mask for the windowIcon

8.1.2 instance

accessing

appClient starts COAST, provides the MemoSetClient with the correct host and returns it

host returns an proved value of servername

inverseColor: aColorValue returns a the inverse color for aColorValue

randomColor returns a randomly selected ColorValue

actions all methods are called if the corresponding button or menuitem is activated.

help opens the help file

joinGame opens a SelectionDialog with the existing volumes and calls startGame: with the selected volume.

mediator starts or stops the CoastMediator depending on the state of it.

newGame opens a Dialog, asks for a new game name and then calls startGame:

playerColor opens a ColorSelectionDialog and sets the color at the special volume '#users', changes the color of the colorbutton, too.

stopGame exits the current volume

visitAragon opens a browser with the aragon website by calling the method at `MemoSetMenu class`

visitCoast opens a browser with the Coast website by calling the method at `MemoSetMenu class`

visitMemoSet opens a browser with the MemoSet website by calling the method at `MemoSetMenu class`

aspects

server a simple access to the valueholder for server

username a simple access to the valueholder for username

dialogs

dialogAboutMemoSet shows a short dialog about MemoSet.

dialogChooseGameType ask which type of cards (subclasses of `MemoSetCard`) shoud be used

dialogChooseNewVolume choose a new volume

dialogChooseVolumeFrom: volumes choose a volume from the given list

dialogNewPassForUser: aUsername ask for new password

dialogPassNoMatch show a warning that the passwords don't match.

dialogRevalidatePassForUser: aUsername let the user reenter his new password

dialogValidatePassForUser: aUsername ask for the user's password

dialogWrongPass tell the user his password was wrong

events

noticeOfWindowClose: t1 stops the MemoSetClient and the CoastMediator if the MemoSetLauncher window is closed.

serverChanged whenever the server is changed, the username InputField should be cleared.

usernameChanged if the username changes we have to update the usercolor. We read it at the special volume '#users'.

validateUsername: anInputBox we have to validate the user here. There are two possibilities:

the user already exists - then we have to check his password and accept or decline the change

the user does not exist - then we first have to ask him for a password, check if he remembers and did no typo and then create a new user.

utility

setColorButtonColor: aColorValue changes the color of the colorbutton to aColorValue

startCoast starts the MemoSetClient if he is not already running

startGame: aName enters the specified game

startGame: aName ofType: aMemoSetCardSubclass creates the game with the specified type of cards (subclasses of `MemoSetCard`)

startMediator starts the COAST mediator

stopMediator stops the COAST mediator

9 MemoSetLocalApplicationModel

inherits from: `CoastLocalApplicationModel`

slots:

gameView - is computed by computeGameView

players - is computed by computePlayers

userView - is computed by computeUserView

9.1 methods

9.1.1 class

accessing

sharedApplicationModelClass returns the shared application model class MemoSetApplicationModel

interface specs

windowSpecs specifies the userinterface for the launcher.

resources

menu calls MemoSetMenu

windowIcon contains the windowIcon

9.1.2 instance

actions

collectCards starts the "let's see if a set is selected and collect it"-action.

help opens the help file

visitAragon opens a browser with the aragon website by calling the method at `MemoSetMenu class`

visitCoast opens a browser with the Coast website by calling the method at `MemoSetMenu class`

visitMemoSet opens a browser with the MemoSet website by calling the method at `MemoSetMenu class`

aspects

cardsOnPile a simple access to the valueholder for the number of cards left on the pile

gameName a simple access to the valueholder for the gameName

playersAspect a simple access to the SelectionInList with the players

scoreAspect a simple access to the valueholder for the player's score

setNumberAspect a simple access to the valueholder for the user's number of sets

composing

computeGameView is the computation method for the corresponding slot.

computePlayers is the computation method for the corresponding slot.

computeUserView is the computation method for the corresponding slot.

events

noticeOfWindowClose: t1 if the window is closed shutdown the MemoSetClient

view

gameView is a simple access to the gameView slot

userView is a simple access to the userView slot

10 MemoSetMenu

inherits from: `Object` implements a short menu which is used from the MemoSetLauncher and the MemoSetLocalApplicationModel.

10.1 methods

10.1.1 class

resources

menu is the menu itself.

utility

help opens the help file for MemoSet by using a Win32Process.

openBrowserAt: urlStringInput opens the default Application for urls with the given url by using a Win32Process.

11 MemoSetUser

inherits from: `CoastUser`

slots:

collectedCards (type: orderedCollection) - the cards the user has collected.

color (type: singleValue) - the user's color.

password (type: singleValue) - the user's password.

score (type: singleValue) - the user's score.

selectedCards (type: singleValue) - the cards the user has selected.

sets (type: singleValue) - the number of user's sets.

11.1 methods

11.1.1 instance

accessing

collectCards: aCollection take the cards and change the values for sets and score.

collectedCards is an access method to the cards the user has already collected.

color is an access method to the user's color.

color: aColorValue is an access method to the user's color.

lastSet assuming the user has laid his cards on his pile the method returns the last set.

password is an access method to the user's password.

password: aString is an access method to the user's password.

score is an access method to the user's score.

selectedCards returns the cards the user has selected.

sets is an access method to the user's number of sets.

comparing

¡= makes it possible to sort user.

printing

printString overwriting this method makes it much nicer. Now there is displayed the user's name, his score and the number of sets he already has.

12 MemoSetUserView

inherits from: `CoastUserView`

12.1 methods

12.1.1 instance

composing

composeComponents takes the last three cards the user collected and calculates their positions.

Index