Assignment 3

Generated by Doxygen 1.8.15

Chapter 1

Class Index

1.1 Class List

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

Chapter 2

File Index

2.1 File List

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

Chapter 3

Class Documentation

3.1 BoardT Class Reference

Class representing the GameBoard.

`#include <GameBoard.h>`

Public Member Functions

- BoardT (vector< CardT > in)

  Constructs a new BoardT.
- bool is_valid_tab_mv (CategoryT a, unsigned int x, unsigned int y)

  Checks if moving the element from one Tableau to another component is valid or not.
- bool is_valid_waste_mv (CategoryT a, unsigned int x)

  Checks if moving the element from Waste to another component is valid or not.
- bool is_valid_deck_mv ()

  Checks if moving the element from deck to another component is valid or not.
- void tab_mv (CategoryT a, unsigned int x, unsigned int y)

  moves the element from deck to another component
- void waste_mv (CategoryT a, unsigned int x)

  Moves a card from the waste.
- void deck_mv ()

  Moves a card from the deck.
- CardStackT get_tab (unsigned int a)

  Returns a cardstack for cards at the tableau at the specific index.
- CardStackT get_foundation (unsigned int a)

  Returns a cardstack for cards at the foundation at the specific index.
- CardStackT get_deck ()

  Returns the deck.
- CardStackT get_waste ()

  Returns the waste.
- bool valid_mv_exists ()

  checks if a valid move exists
- bool is_win_state ()

  checks if the BoardT is currently in win state

3.1.1 Detailed Description

Class representing the GameBoard.

Includes four main components Tableau, Foundation, Deck, and Waste

3.1.2 Constructor & Destructor Documentation

3.1.2.1 BoardT()

```
BoardT::BoardT (
            vector< CardT > in )
```

Constructs a new BoardT.

Parameters

in	The vector of CardT (aka the whole two card deck)

3.1.3 Member Function Documentation

3.1.3.1 get_deck()

```
CardStackT BoardT::get_deck ( )
```

Returns the deck.

Returns

the deck

3.1.3.2 get_foundation()

```
CardStackT BoardT::get_foundation (
            unsigned int a )
```

Returns a cardstack for cards at the foundation at the specific index.

Parameters

a	An integer corresponding to the stack for the foundation

Returns

cardstack for cards at the foundation at the specific index

3.1.3.3 get_tab()

```
CardStackT BoardT::get_tab (
            unsigned int a )
```

Returns a cardstack for cards at the tableau at the specific index.

Parameters

a	An integer corresponding to the stack for the tableau

Returns

cardstack for cards at the tableau at the specific index

3.1.3.4 get_waste()

```
CardStackT BoardT::get_waste ( )
```

Returns the waste.

Returns

the waste

3.1.3.5 is_valid_deck_mv()

```
bool BoardT::is_valid_deck_mv ( )
```

Checks if moving the element from deck to another component is valid or not.

Returns

if valid -> true else -> false

3.1.3.6 is_valid_tab_mv()

```
bool BoardT::is_valid_tab_mv (
            CategoryT a,
            unsigned int x,
            unsigned int y )
```

Checks if moving the element from one Tableau to another component is valid or not.

Parameters

a	The (Component) where the top of tableau want to go
x	Index of Tableau (from)
y	Index of another (Component) (to)

Returns

if valid -> true else -> false

3.1.3.7 is_valid_waste_mv()

```
bool BoardT::is_valid_waste_mv (
            CategoryT a,
            unsigned int x )
```

Checks if moving the element from Waste to another component is valid or not.

Parameters

a	The (Component) where the top of waste want to go
x	Index of another (Component) (to)

Returns

if valid -> true else -> false

3.1.3.8 is_win_state()

```
bool BoardT::is_win_state ( )
```

checks if the BoardT is currently in win state

Returns

true if the game has been won, false otherwise

3.1.3.9 tab_mv()

```
void BoardT::tab_mv (
            CategoryT a,
            unsigned int x,
            unsigned int y )
```

moves the element from deck to another component

Parameters

a	The (Component) where the top of tableau want to go
x	Index of which Tableau (Component) (from)
y	Index of another (Component) (to)

3.1.3.10 valid_mv_exists()

```
bool BoardT::valid_mv_exists ( )
```

checks if a valid move exists

Returns

true if a valid move exists, false otherwise

3.1.3.11 waste_mv()

```
void BoardT::waste_mv (
            CategoryT a,
            unsigned int x )
```

Moves a card from the waste.

Parameters

a	The category for the cards
x	The card that will have a card placed on it

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

- include/GameBoard.h

3.2 CardT Struct Reference

Describes the number of the card.

`#include <CardTypes.h>`

Public Attributes

- SuitT s
  first component of playing card (Suit or color)
- RankT r
  second component of playing card (Number 1-13)

3.2.1 Detailed Description

Describes the number of the card.

The documentation for this struct was generated from the following file:

- include/CardTypes.h

3.3 Stack< T > Class Template Reference

Class representing Stack (i.e. vector) of class T.

`#include <Stack.h>`

Public Member Functions

- Stack (vector< T > in)
  Constructor for new Stack.
- Stack push (T a)
  inserts element at the back of the stack
- Stack pop ()
  removes element from the stack
- T top ()
  gets the top element in the stack
- unsigned int size ()
  get the size of the Stack
- vector< T > toSeq ()
  returns the Stack in form of vector

3.3.1 Detailed Description

template<class T>
class Stack< T >

Class representing Stack (i.e. vector) of class T.

Data structure (stack) and its components

3.3.2 Constructor & Destructor Documentation

3.3.2.1 Stack()

```
template<class T>
Stack< T >::Stack (
            vector< T > in )
```

Constructor for new Stack.

Parameters

in	The input stack for the new Stack

3.3.3 Member Function Documentation

3.3.3.1 pop()

```
template<class T>
Stack Stack< T >::pop ( )
```

removes element from the stack

Returns

the Stack where element came from

3.3.3.2 push()

```
template<class T>
Stack Stack< T >::push (
            T a )
```

inserts element at the back of the stack

Parameters

a	The card that will be added

Returns

the Stack where element went

3.3.3.3 size()

```
template<class T>
unsigned int Stack< T >::size ( )
```

get the size of the Stack

Returns

natural number from (0 to n)

3.3.3.4 top()

```
template<class T>
T Stack< T >::top ( )
```

gets the top element in the stack

Returns

The top element in the Stack

3.3.3.5 toSeq()

```
template<class T>
vector<T> Stack< T >::toSeq ( )
```

returns the Stack in form of vector

Returns

vector form of the Stack

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

- include/Stack.h

Chapter 4

File Documentation

4.1 include/CardStack.h File Reference

defining CardStack

```
#include "CardTypes.h"
#include "Stack.h"
```

Typedefs

- typedef Stack< CardT > CardStackT
  Stack of type "CardT" alias is CardStackT.

4.1.1 Detailed Description

defining CardStack

Author

Shivam Taneja

4.2 include/CardTypes.h File Reference

All the types of cards and constants required.

```
#include <utility>
```

Classes

- struct CardT
  Describes the number of the card.

Macros

- #define ACE 1

  RankT for an Ace.
- #define JACK 11

  RankT for an Jack.
- #define QUEEN 12

  RankT for a Queen.
- #define KING 13

  RankT for a King.

Typedefs

- typedef unsigned short int RankT

  Describes the rank of a card.

Enumerations

- enum CategoryT { **Tableau**, **Foundation**, **Deck**, **Waste** }

  Describes the Category of where the card is.
- enum SuitT { **Heart** = 0, **Diamond**, **Club**, **Spade** }

  Describes the suit of the card.

Variables

- const int TOTAL_CARDS = 104

  total number of cards

4.2.1 Detailed Description

All the types of cards and constants required.

Author

Shivam Taneja

4.3 include/GameBoard.h File Reference

API and state variables of Board game ie the actual UI for user.

```
#include <iostream>
#include <functional>
#include <vector>
#include "CardTypes.h"
#include "CardStack.h"
```

Classes

- class BoardT

  Class representing the GameBoard.

4.3.1 Detailed Description

API and state variables of Board game ie the actual UI for user.

Author

Shivam Taneja

4.4 include/Stack.h File Reference

API for stack with prototypes of functions and state variables.

```
#include <vector>
#include <iostream>
```

Classes

- class Stack< T >

  Class representing Stack (i.e. vector) of class T.

4.4.1 Detailed Description

API for stack with prototypes of functions and state variables.

Author

Shivam Taneja

Index