Boards Class Reference

Public Member Functions

	Boards ()
	~Boards ()
bool	**isValid** (int, int)
void	**displayOffensiveBoard** () const
void	**displayDefensiveBoard** () const Displays the offensive board of hits and misses for the. More...
void	**displayBoth** () const
bool	**isHit** (int, int)
void	**markShips** (int, int)
bool	**gameWon** ()
char	**getLocation** (int, int)
void	**boardShot** (int, int)
void	**ownBoardHit** (int, int)
void	**ownBoardMiss** (int, int)
void	**otherBoardHit** (int, int)
void	**otherBoardMiss** (int, int)
void	**clearBoards** ()

Private Attributes

char	**offensiveBoard** [8][8] Creating a 2D array with size 8*8.
char	**defensiveBoard** [8][8] Creating a 2D array with size 8*8 for player to see their own board.
int	**rows** Get the coordinate's row and col number.
int	**cols**

Constructor & Destructor Documentation

◆ Boards()

Boards::Boards ( )

Precondition

Player wants to make their boards.

Postcondition

Boards are filled with water and ready to use.

◆ ~Boards()

Boards::~Boards ( )

Precondition

Users are done with the game.

Postcondition

Memory is freed from allocation.

Member Function Documentation

◆ boardShot()

```
void Boards::boardShot ( int  column,
                         int  row
                       )
```

Precondition

Takes two integers, one for column and one for row

Postcondition

none

Note

Marks the board with either an 'H' if a ship 'S' is found or an 'M' if water '~' is found

◆ clearBoards()

void Boards::clearBoards ( )

Precondition
Game is over and users want a rematch.

Postcondition
Boards are cleaned off.

◆ displayBoth()

void Boards::displayBoth ( ) const

Precondition
none

Postcondition
none

Note
Displays both the offensive and defensive boards for the current Player

◆ displayDefensiveBoard()

void Boards::displayDefensiveBoard ( ) const

Displays the offensive board of hits and misses for the.

Precondition
none

Postcondition
none

Note
Displays the defensive board for the current Player

◆ displayOffensiveBoard()

void Boards::displayOffensiveBoard ( ) const

Precondition
none

Postcondition
none

Note
Displays the offensive board for the current Player

◆ gameWon()

bool Boards::gameWon ( )

Precondition
none

Postcondition
Returns a boolean value

Note
Returns true if no more 'S' exist in the defensive array

◆ getLocation()

```
char Boards::getLocation ( int  column,
                           int  row
                         )
```

Precondition
Takes two integers, one for column and one for row

Postcondition
Returns a character

Note
Returns the character located at the specified location

◆ isHit()

```
bool Boards::isHit ( int  column,
                     int  row
                   )
```

Precondition
Takes two integers, one for column and one for row

Postcondition
returns a boolean value

Note
returns true if the position indicated is 'S'

◆ isValid()

```
bool Boards::isValid ( int  column,
                       int  row
                     )
```

Precondition
Takes two integers, one for column and one for row

Postcondition
returns a boolean value

Note
returns true if the position indicated is '~'

◆ markShips()

```
void Boards::markShips ( int  column,
                         int  row
                       )
```

Precondition

Takes two integers, one for column and one for row

Postcondition

none

Note

Marks the array with an 'S' at the specified location

◆ otherBoardHit()

```
void Boards::otherBoardHit ( int  column,
                             int  row
                           )
```

Precondition

Takes two integers, one for column and one for row

Postcondition

none

Note

When the player is being shoot at marks the hits, on their defensive board if the shot is a hit, marks the specified location with an 'M'

◆ otherBoardMiss()

```
void Boards::otherBoardMiss ( int  column,
                              int  row
                            )
```

Precondition

Takes two integers, one for column and one for row

Postcondition

none

Note

When the player is being shoot at marks the misses, on their defensive board if the shot is a miss, marks the specified location with an 'M'

◆ ownBoardHit()

```
void Boards::ownBoardHit ( int  column,
                           int  row
                         )
```

Precondition

Takes two integers, one for column and one for row

Postcondition

none

Note

When the player is shooting marks the hits, on their offensive board if the shot is a hit, marks the specified location with an 'M'

◆ ownBoardMiss()

```
void Boards::ownBoardMiss ( int  column,
                            int  row
                          )
```

Precondition

Takes two integers, one for column and one for row

Postcondition

none

Note

When the player is shooting marks the misses, on their offensive board if the shot is a hit, marks the specified location with an 'M'

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

- C:/Users/Qing Dong/Desktop/EECS_448_Seg_Faults-master/**Boards.h**
- C:/Users/Qing Dong/Desktop/EECS_448_Seg_Faults-master/Boards.cpp

---