My Project

Generated by Doxygen 1.9.1

Chapter 1

Namespace Index

1.1 Namespace List

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

Chapter 2

Hierarchical Index

2.1 Class Hierarchy

This inheritance list is sorted roughly, but not completely, alphabetically:

Chapter 3

Class Index

3.1 Class List

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

Chapter 4

Namespace Documentation

4.1 Ships_JosefLukasek Namespace Reference

Classes

- class AIPlayer

  This class is responsible for AI player.
- class ShipsForm

  This class is responsible for the GUI of the game.
- class Ship

  Represents a game plan grid.
- class Square

  Represents a square on the game plan grid.
- class GamePlan

  Represents a game plan with ships and squares.
- class **NetworkHandler**

  This class is responsible for network communication.
- class **Program**

Enumerations

- enum class SquareState { **Free** , **Occupied** , **Hit** }

  Enumeration to represent the state of a square on the grid.
- enum class PlanState { **Locked** , **Placing** , **Standby** , **Hidden** }

  Enumeration to represent the state of the game plan.
- enum class **GameState** {
  MainMenu , **MultiMenu** , **SetHost** , **SetClient** ,
  Connecting , **Placing** , **GameClient** , **GameHost** ,
  MultiGameOver , **SinglePlacing** , **SingleGame** , **SingleGameOver** }

4.1.1 Enumeration Type Documentation

4.1.1.1 PlanState

enum Ships_JosefLukasek.PlanState [strong]

Enumeration to represent the state of the game plan.

4.1.1.2 SquareState

enum Ships_JosefLukasek.SquareState [strong]

Enumeration to represent the state of a square on the grid.

Chapter 5

Class Documentation

5.1 Ships_JosefLukasek.AIPlayer Class Reference

This class is responsible for AI player.

Public Member Functions

- AIPlayer ()

  Initializes a new instance of the AIPlayer class and generates all possible shots.
- string GetPlan ()

  Upon call generates a plan for ship placement

Public Attributes

- int

  Upon call generates a random coordinate for shot that has not been shot yet

5.1.1 Detailed Description

This class is responsible for AI player.

5.1.2 Constructor & Destructor Documentation

5.1.2.1 AIPlayer()

```
Ships_JosefLukasek.AIPlayer.AIPlayer ( )  [inline]
```

Initializes a new instance of the AIPlayer class and generates all possible shots.

5.1.3 Member Function Documentation

5.1.3.1 GetPlan()

```
string Ships_JosefLukasek.AIPlayer.GetPlan ( )  [inline]
```

Upon call generates a plan for ship placement

Returns

The plan.

5.1.4 Member Data Documentation

5.1.4.1 int

```
Ships_JosefLukasek.AIPlayer.int
```

Upon call generates a random coordinate for shot that has not been shot yet

Returns

The shot.

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

- Ships-JosefLukasek/Ships-JosefLukasek/AIPlayer.cs

5.2 Ships_JosefLukasek.GamePlan Class Reference

Represents a game plan with ships and squares.

Public Member Functions

- GamePlan (ShipsForm form, int defaultLeft, int defaultTop, Action< bool,(int i, int j)> shootCallBack)

  Initializes a new game plan with the specified form, default left and top positions, and shoot callback action.
- void OnPlanClick (object sender, EventArgs e)

  Handles the click event on a square in the game plan grid.
- void OnPlanHover (object? sender, EventArgs? e)

  Handles the hover event over a square in the game plan grid.
- bool MarkSquareAsHit ((int i, int j) pos)

  Marks the square as hit.
- bool TryReadyLock ()

  Tries to lock the game plan with the condition that all ships are placed.
- void Lock ()

  Locks the game plan and refreshes the graphics.
- void Unlock ()

  Unlocks the game plan so that player can click on squares and refreshes the graphics.
- void PickShip (int length)

  Picks a ship of the specified length to be placed on the game plan grid.
- void RotateCurrShip ()

  Rotates the current ship 90 degrees clockwise.
- override string ToString ()

  Converts the game plan grid to a string representation.
- bool LoadPlanFromString (string plan)

  Loads a game plan grid from a string representation.
- void Resize (bool local)

  Resizes the game plan grid based on the size of the form's client area.
- void Dispose ()

  Disposes of the game plan grid.

Public Attributes

- int i

  Converts a button to grid coordinates.

Properties

- PlanState state `[get]`

  The current state of the game plan.
- bool IsReady = false `[get]`

  Indicates whether the game plan is ready to be used because all ships have been placed.
- int hitCounter = 0 `[get]`

  Indicates how many ship squares have been hit.

5.2.1 Detailed Description

Represents a game plan with ships and squares.

5.2.2 Constructor & Destructor Documentation

5.2.2.1 GamePlan()

```
Ships_JosefLukasek.GamePlan.GamePlan (
            ShipsForm form,
            int defaultLeft,
            int defaultTop,
            Action< bool,(int i, int j)> shootCallBack )  [inline]
```

Initializes a new game plan with the specified form, default left and top positions, and shoot callback action.

Parameters

form	The form associated with the game plan.
defaultLeft	The default left position of the grid.
defaultTop	The default top position of the grid.
shootCallBack	The callback action for shooting on a square.

5.2.3 Member Function Documentation

5.2.3.1 Dispose()

```
void Ships_JosefLukasek.GamePlan.Dispose ( )  [inline]
```

Disposes of the game plan grid.

5.2.3.2 LoadPlanFromString()

```
bool Ships_JosefLukasek.GamePlan.LoadPlanFromString (
            string plan )  [inline]
```

Loads a game plan grid from a string representation.

Parameters

plan	The string representation of the game plan grid.

Returns

True if the string was valid, otherwise false.

5.2.3.3 Lock()

```
void Ships_JosefLukasek.GamePlan.Lock ( )  [inline]
```

Locks the game plan and refreshes the graphics.

5.2.3.4 MarkSquareAsHit()

```
bool Ships_JosefLukasek.GamePlan.MarkSquareAsHit (
            (int i, int j) pos )  [inline]
```

Marks the square as hit.

Parameters

pos	The position of the square.

Returns

True if the square was hit, false otherwise.

5.2.3.5 OnPlanClick()

```
void Ships_JosefLukasek.GamePlan.OnPlanClick (
            object sender,
            EventArgs e )  [inline]
```

Handles the click event on a square in the game plan grid.

Parameters

sender	The sender object.
e	The event arguments.

5.2.3.6 OnPlanHover()

```
void Ships_JosefLukasek.GamePlan.OnPlanHover (
            object?  sender,
            EventArgs?  e )  [inline]
```

Handles the hover event over a square in the game plan grid.

Parameters

sender	The sender object.
e	The event arguments.

5.2.3.7 PickShip()

```
void Ships_JosefLukasek.GamePlan.PickShip (
            int length )  [inline]
```

Picks a ship of the specified length to be placed on the game plan grid.

Parameters

length	The length of the ship to pick.

5.2.3.8 Resize()

```
void Ships_JosefLukasek.GamePlan.Resize (
            bool local )  [inline]
```

Resizes the game plan grid based on the size of the form's client area.

5.2.3.9 RotateCurrShip()

```
void Ships_JosefLukasek.GamePlan.RotateCurrShip ( )  [inline]
```

Rotates the current ship 90 degrees clockwise.

5.2.3.10 ToString()

```
override string Ships_JosefLukasek.GamePlan.ToString ( )  [inline]
```

Converts the game plan grid to a string representation.

Returns

A string representing the game plan grid.

5.2.3.11 TryReadyLock()

```
bool Ships_JosefLukasek.GamePlan.TryReadyLock ( )  [inline]
```

Tries to lock the game plan with the condition that all ships are placed.

Returns

True if all ships are placed, false otherwise.

5.2.3.12 Unlock()

```
void Ships_JosefLukasek.GamePlan.Unlock ( )  [inline]
```

Unlocks the game plan so that player can click on squares and refreshes the graphics.

5.2.4 Member Data Documentation

5.2.4.1 i

```
int Ships_JosefLukasek.GamePlan.i
```

Converts a button to grid coordinates.

Parameters

btn	The button to convert.

Returns

The grid coordinates as a tuple (i, j).

5.2.5 Property Documentation

5.2.5.1 hitCounter

```
int Ships_JosefLukasek.GamePlan.hitCounter = 0  [get]
```

Indicates how many ship squares have been hit.

5.2.5.2 IsReady

```
bool Ships_JosefLukasek.GamePlan.IsReady = false  [get]
```

Indicates whether the game plan is ready to be used because all ships have been placed.

5.2.5.3 state

```
PlanState Ships_JosefLukasek.GamePlan.state  [get]
```

The current state of the game plan.

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

- Ships-JosefLukasek/Ships-JosefLukasek/GamePlan.cs

5.3 Ships_JosefLukasek.Ship Class Reference

Represents a game plan grid.

Public Types

- enum class ShipDir { **Down** , **Up** , **Left** , **Right** }
  Enumeration to specify the direction of the ship.

Public Member Functions

- Ship (int length)

  Initializes a new ship with the specified length.
- ShipDir Rotate ()

  Rotates the ship 90 degrees clockwise.
- void Remove ()

  Removes the ship from the grid.

Properties

- ShipDir dir `[get]`

  Current direction of the ship.
- int Length `[get]`

  Length of the ship.
- Square[ ] occupiedSquares `[get]`

  Array to store the squares occupied by the ship.

5.3.1 Detailed Description

Represents a game plan grid.

5.3.2 Member Enumeration Documentation

5.3.2.1 ShipDir

```
enum Ships_JosefLukasek.Ship.ShipDir  [strong]
```

Enumeration to specify the direction of the ship.

5.3.3 Constructor & Destructor Documentation

5.3.3.1 Ship()

```
Ships_JosefLukasek.Ship.Ship (
            int length )  [inline]
```

Initializes a new ship with the specified length.

Parameters

length	The length of the ship.

5.3.4 Member Function Documentation

5.3.4.1 Remove()

```
void Ships_JosefLukasek.Ship.Remove ( )  [inline]
```

Removes the ship from the grid.

5.3.4.2 Rotate()

```
ShipDir Ships_JosefLukasek.Ship.Rotate ( )  [inline]
```

Rotates the ship 90 degrees clockwise.

Returns

The new direction of the ship.

5.3.5 Property Documentation

5.3.5.1 dir

```
ShipDir Ships_JosefLukasek.Ship.dir  [get]
```

Current direction of the ship.

5.3.5.2 Length

```
int Ships_JosefLukasek.Ship.Length  [get]
```

Length of the ship.

5.3.5.3 occupiedSquares

```
Square [] Ships_JosefLukasek.Ship.occupiedSquares  [get]
```

Array to store the squares occupied by the ship.

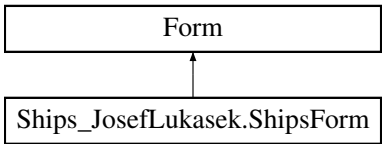
The documentation for this class was generated from the following file:

- Ships-JosefLukasek/Ships-JosefLukasek/GamePlan.cs

5.4 Ships_JosefLukasek.ShipsForm Class Reference

This class is responsible for the GUI of the game.

Inheritance diagram for Ships_JosefLukasek.ShipsForm:



Classes

- class **StateController**

  This class is responsible for controlling the state of the game.
- class **Translator**

  This class is responsible for translating messages from network to actions in game

Public Member Functions

- ShipsForm ()

  Initializes a new instance of the ShipsForm class.
- void ReceiveMessage (string message)

  Receives message from network and passes it to translator in GUI thread

Protected Member Functions

- override void Dispose (bool disposing)

  Clean up any resources being used.

5.4.1 Detailed Description

This class is responsible for the GUI of the game.

5.4.2 Constructor & Destructor Documentation

5.4.2.1 ShipsForm()

```
Ships_JosefLukasek.ShipsForm.ShipsForm ( )  [inline]
```

Initializes a new instance of the ShipsForm class.

5.4.3 Member Function Documentation

5.4.3.1 Dispose()

```
override void Ships_JosefLukasek.ShipsForm.Dispose (
            bool disposing )  [inline], [protected]
```

Clean up any resources being used.

Parameters

disposing	true if managed resources should be disposed; otherwise, false.

5.4.3.2 ReceiveMessage()

```
void Ships_JosefLukasek.ShipsForm.ReceiveMessage (
            string message )  [inline]
```

Receives message from network and passes it to translator in GUI thread

Parameters

message	The message.

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

- Ships-JosefLukasek/Ships-JosefLukasek/Form1.cs
- Ships-JosefLukasek/Ships-JosefLukasek/Form1.Designer.cs
- Ships-JosefLukasek/Ships-JosefLukasek/Translator.cs

5.5 Ships_JosefLukasek.Square Class Reference

Represents a square on the game plan grid.

Public Member Functions

- Square (SquareState state, Button button)

  Initializes a new square with the specified state and associated button.

Properties

- SquareState State `[get, set]`

  The current state of the square. Setting the state to Occupied increases the occupation counter and setting it to Free decreases it.
- Square **Up** `[get, set]`
- Square **Down** `[get, set]`
- Square **Right** `[get, set]`
- Square **Left** `[get, set]`
- Button button `[get]`

  The button associated with the square.
- Ship? ship `[get, set]`

  The ship occupying the square.

5.5.1 Detailed Description

Represents a square on the game plan grid.

5.5.2 Constructor & Destructor Documentation

5.5.2.1 Square()

```
Ships_JosefLukasek.Square.Square (
            SquareState state,
            Button button )  [inline]
```

Initializes a new square with the specified state and associated button.

Parameters

state	The initial state of the square.
button	The button associated with the square.

5.5.3 Property Documentation

5.5.3.1 button

```
Button Ships_JosefLukasek.Square.button  [get]
```

The button associated with the square.

5.5.3.2 ship

```
Ship?  Ships_JosefLukasek.Square.ship  [get], [set]
```

The ship occupying the square.

5.5.3.3 State

```
SquareState Ships_JosefLukasek.Square.State  [get], [set]
```

The current state of the square. Setting the state to Occupied increases the occupation counter and setting it to Free decreases it.

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

- Ships-JosefLukasek/Ships-JosefLukasek/GamePlan.cs

Index