**ThunderBird Team**

Dian Penchev (Penchev)

Georgi Kostadinov (georgikostadinov)

Georgi Yakimov (Santyago)

Hristo Tsekov (H.Tsekov)

Vladislav Budnikov (vladislav.budnikov)

Vladislava Atanasova (Vladka)

<https://thunderbird.codeplex.com/>

**“Match-u-Picture”**

**• Game play**

The objective of the game is to find all the pairs in the deck. The size of the deck depends on the level you choose:

1) Easy - 4x4 cards

2) Medium - 6x6 cards

3) Hard - 8x8 cards

Move the cursor with the arrow keys on the keyboard until you reach the card you want to open and then press Enter. Do the same for the second card. If they match you have found a pair and the cards will remain open. If they don't match the cards will be closed.

Do the same until you find all the pairs. Try to do it for the least possible time with the least possible moves.

**• Summary**

**The game starts** with Welcome screen. Then the main menu shows up.

• Play (Leads you to submenu – you have to choose level difficulty)

• Baby (4 x 4 cards)

• Not Hard (6 x 6 cards)

• You Crazy? (8 x 8 cards)

After you choose a level, the game asks for player name.

• Options (You can choose to on/off the sounds of the game).

• How to play (Description of the game rules and the keys you use to play with).

• Leave us (Exits the game).

**The game ends** when you open all the cards. On the screen you can see the number of moves you have done to solve the puzzle and the time of your game play. On the next lines you can see all of the results of your playing for the current level until now.

• **Behind the curtains:**

For saving the symbols for the deck is used ***two-dimensional array***. When the player choose a level, a number of symbols are read from external .txt file and a Shuffle method distributes the symbols to a matrix. Every symbol from the file coрresponds to a simple picture which are described in a *switch-case* structure.

To check the cards that are already opened is used ***two-dimensional* *bool*** array.

***List<>*** is used a couple of times - for reading and displaying time and movements results, reading symbols.

Use ***external files*** for reading a number of symbols, display the menus and save (write) the results of every player.

The game consists of a bunch of ***classes and methods*** used for:

**Class Field**

• Print cards backs, symbols (simple combination of symbols).

• Coloring the cards over the deck while moving the cursor through the cards.

• Shuffle the symbols in the matrix.

• Comparing opened cards.

• Following player moves

**Class Cell**

• Constructor for creating objects of type *Cell* that fills the matrix.

• Displaying opened cards.

**Class Player**

• Couple of methods that gets data about current player – time, moves, name

• Displaying player results

**Class FrontPage**

• Displaying all the menus

• Follows player’s choice

• Sound options

***Existing .NET classes*** used in the game:

Char, Bool, Int, String, Console, IO, Collection.Generic, Thread, ect.

A couple of ***exception handling:***

* If Symbol.txt file is not found
* If RulesAndControls.txt file is not found
* If the player do not enter a name

As an extra-contribution to the project there used object-oriented approach, sound effects, randomization.