MemoryGame

Originally a card game, this is a Java Program of a Card-Matching game.
All cards are face down when the game begins, and you need to flip 2 distinct cards to see if their shapes match. if they don't, when you flip up a third card, the 2 previous will flip back down. you need to remember those shapes to match them in the future.

Currently, if 2 shapes are showing and you press one of the showing shapes again, and proceed to make a new match you will confuse the game. *You have to make a match with 2 distinct cards, and if they're not right,* **press an empty card.** I'm trying to fix that.

How it works

Randomizing the board

First, `initShuffleTiles()` is called when the start button is pressed. `initShuffleTiles()` loads the numbers 0-15 (in 4x4) or 0-35 (in6x6) into an array, then shuffles the array. Then the variables from index 0 and 1 are assigned to the first shape, index 2 and 3 to the next shape, and so on. Those variables become the ID's or locations of each of the shapes on the board. Since circle takes the first 2 indeces of the array, and the numbers in those 2 slots are a random number each time, it will place 2 circles in random places on the board.

in the `tileControl` class, there are globals at the top that go to each shape. The data in there is the ID of the shapes.

Shape Codes

Any code that deals with shapes has to deal with them in a specific order, because each shape is assigned a code. For example, if they're in an array, the index in the array corresponds to their shape code, so you would just pass `type`, or the shape code to read from the array and get the desired shape.

the idea of having codes that correspond to shapes, is that they are used to easily compare and deal with shapes, because a "shape" is more or less intagible in code. The shapes are always in alphabetical order in this code. Here are all the shape codes and the shape they correspond to:

4x4 codes

Shape	Code
circle	0
cross	1
diamond	2
donut	3
eclipse	4
square	5
star	6
x	7

6x6 codes

Shape	Code
circle	0
cross	1
diamond	2
dice	3
donut	4
eclipse	5
eclipse2	6
grapes	7
heart	8
L	9
rectangle	10
snowflake	11
sqDonut	12
square	13
star	14
tear	15
triangle	16
X	17

Shape codes are given when a button press asks a `getTileType()` sub in the `tileControl` class what shape is at the given ID. each button is assigned a unique ID in a local variable called `ID`. The ID serves as a location code. `tileControl` saves the ID's of each shape in its global variables, so when you pass an ID to `getTileType()`, it will return a shape code. the ID would have to equal the ID of a shape.

When a button is pressed

This is the code from a game button on GameBoard2, but it is the same on gameBoard.

```
//button 0, row 1, column 1.
private void Tile11ActionPerformed(java.awt.event.ActionEvent evt) {
        int ID = 0;    //<the unique ID of this particular button.
        int type;      //<holds returned shape code from getTileType(int ID).

        System.out.print("1,1 -- ");             //Print to console what button this is.
        type = tileControl.get6x6TileType(ID);  //asking what shape is at this button.
        buttonPress(ID, type);                   //make game decisions.
    }
```

The calling button will pass its ID or location code to `tilecontrol.getTileType()`. Because the global variables in tileControl are the ID's of each shape, all this does is compare them using if statements. If the button sends its ID of 4, and the local variable for circle1 OR circle2 were assigned an ID of 4 by `initShuffleTiles()`, it gets a shape code `type` of 0 returned.

This is code from the `getTileType()` subroutine in the `tileControl` class.

```
int circle1, circle2;  //<hold the ID of the 2 circles
int cross1, cross2;    //<hold the ID of the 2 crosses
...

public int get4x4TileType(int ID){
        int type = 100;

        if (ID == circle1 || ID == circle2){
            type = 0;  //shape code
            System.out.print(type + " - Circle");
        }
        ...
```

How to Play

How To Win

- flip up all matching pairs of shapes on the game board
- Preferably have a score higher than 0 by the time you match all shapes.
- Cheating off the console.

How To Lose

- you really can't lose, you sort of keep going till you get them all right.
- Having a score of 0 or less is considered losing though the program doesn't let you know you technically lost.

I'm not a developer, I just want to play.

I built the program into an executable .jar. All you need to do is download this repository, go to the `dist` folder, and run `MemoryGame.jar` with your JRE. (which you need to have installed first).