Game of Pong

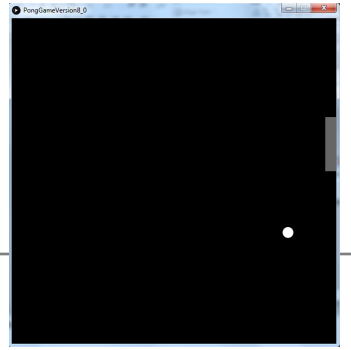
V6 Developing the game further

Produced by:
Dr. Siobhán Drohan
Mr. Colm Dunphy
Mr. Diarmuid O'Connor

Waterford Institute *of* Technology
INSTITIÚID TEICNEOLAÍOCHTA PHORT LÁIRGE

Department of Computing and Mathematics
http://www.wit.ie/

Pong Versions - introduction



v1 - **Ball moving** from left to right of screen. Can bounce off top or bottom

v2 - **Mouse controlling the Paddle**

v3 - **Collision detection** (ball bounces back). Changes made only to PongGame

v4 - **Game Over** (when 3 lives gone), Score (lives Lost). Output to Console. Changes made only to PongGame.

v5 - **Tournament** (no of games per tournament default is 5). Changes made only to PongGame.

v6 - new **Player class using arrays** (no statistics)

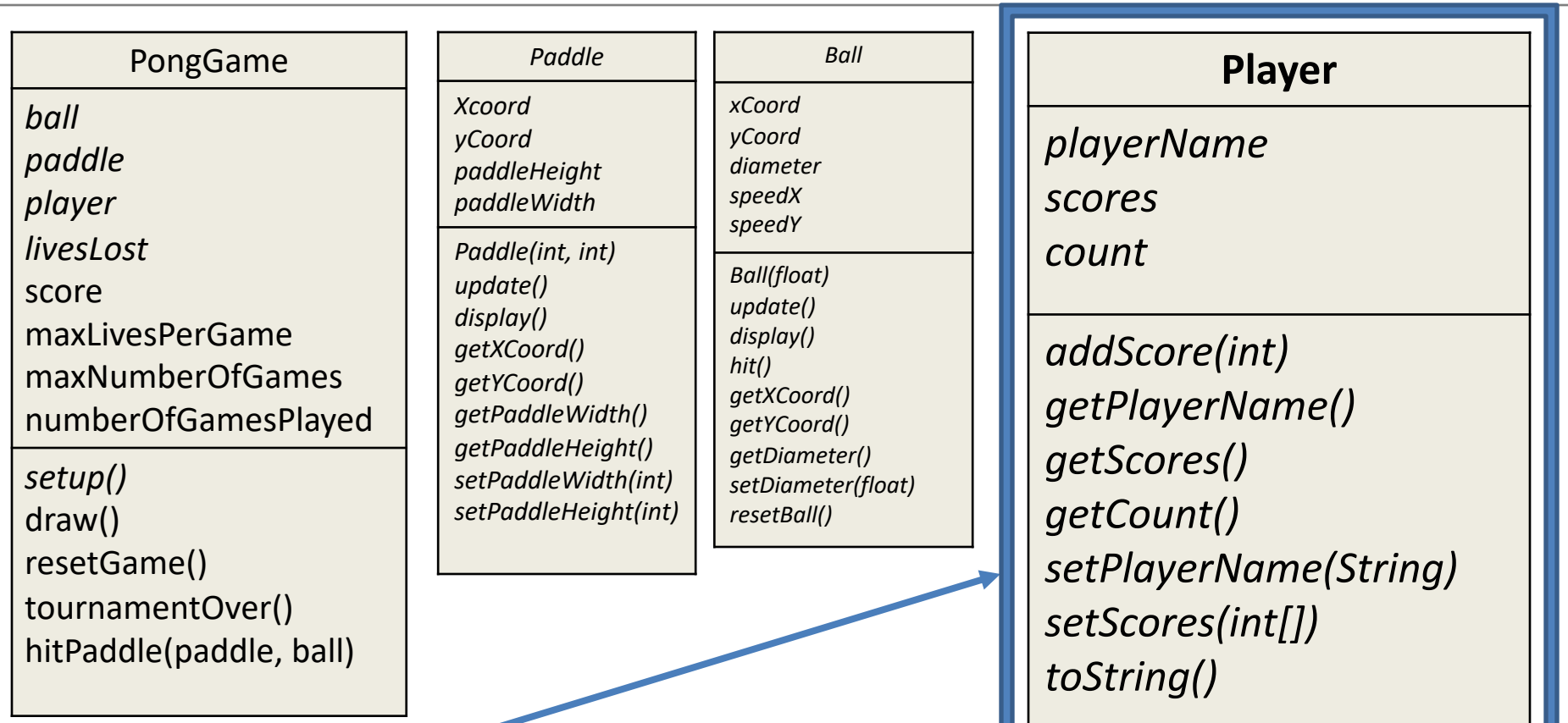
v7 - Player class using arrays (with **statistics** (Tournament Over - highest, lowest, average score))

v8 - **JOptionPane for I/O** instead of console

v9 - alternative algorithm using **Pythagoras Theorem**

Demo of
Pong Game V6.0

Classes in the PongGameV6.0



We have a new **Player** class.

This stores the **score** of the current player in an array

Use of **Arrays** in Player

- We use an array of integers
  - to hold the **scores** for the games.
- declare at start:

```
private int[] scores;
```

- and in constructor:

```
scores = new int[numOfGames]
```

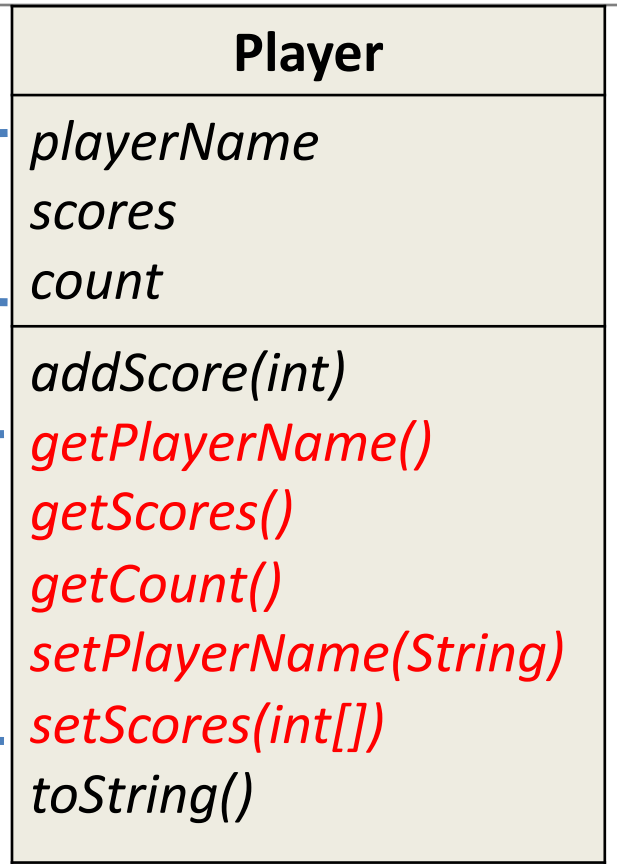
- The **addScore()** method
  - adds a score to this array when called (by PongGame).

Player class

private **fields**

```
private String playerName;
private int[] scores;          //array to hold scores
private int count;             //holds current position
                               //in array
```

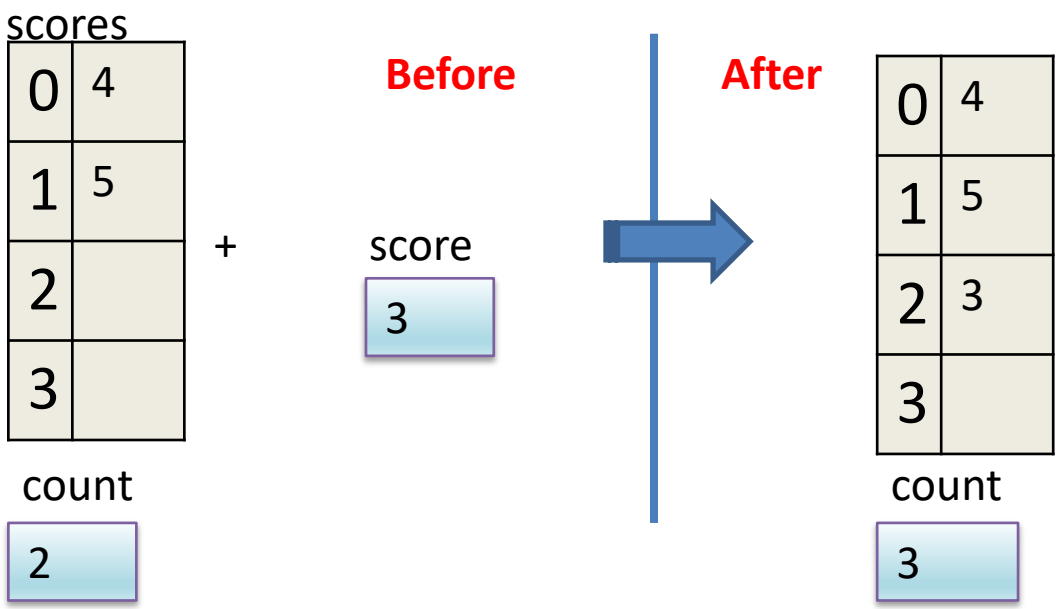
getters and **set**ters for **fields** above



Player class – **addScore()** method

```
public void addScore (int score)  {
  if (score >= 0){
          scores[count] = score;
          count++;
     }
}
```

addScore (int**)**
- takes in the new *score* as a parameter.
- adds the new score to the *array*
- increments the *counts* variable



Player
playerName *scores* *count*
***addScore** (int)* *getPlayerName()* *getScores()* *getCount()* *setPlayerName(String)* *setScores(int[])* *toString()*

Player class – **toString()** method

```
public String toString ()  {
    String str = "Scores for " + playerName + "\n";
    for(int i = 0; i < count; i++){
            str = str + "     Score " + (i+1) + ": " +
                        scores[i] + "\n";
   }
   return str;
}
```

toString()

- **returns a string version of an object**.
- useful method
  - we will have a **toString()** method in most classes.

scores

0	4
1	5
2	3
3	4

toString() returns

"Score 1 : 4 \n"+
"Score 2 : 5 \n"+
"Score 3 : 3 \n" +
"Score 4 : 4\n"

count

4

Player
playerName *scores* *count*
addScore *getPlayerName()* *getScores()* *getCount()* *setPlayerName(String)* *setScores(int[])* ***toString()***

When is the **Player** object used?

```
Ball ball;
Paddle paddle;
Player player;
:
void setup(){
  size(600,600);
  noCursor();
  //setting up ball and paddle with hard-coded sizes.
  ball = new Ball(20.0);
  paddle = new Paddle(20,100);
   //create a player object
   player = new Player("  PongMaster  ", maxNumberOfGames);
 }
```

Need to declare and setup Player

When is the **Player** object used?

‘Sends a message to the player object to add a new score to its scores array.’
i.e.
calls the **addScores()** method.

```
void draw(){
background(0);
paddle.update();
:
 //If the player has no lives left in the current game
else{
   player.addScore(score);
   numberOfGamesPlayed++;.
   if (numberOfGamesPlayed < maxNumberOfGames){
            resetGame();
    }
    else
     tournamentOver();
 }
```

Questions?

References

- Reas, C. & Fry, B. (2014) Processing – A Programming Handbook for Visual Designers and Artists, 2nd Edition, MIT Press, London.