



**Πανεπιστήμιο Αιγαίου**

**Τμήμα Μηχανικών Πληροφοριακών και Επικοινωνιακών  
Συστημάτων**

## **Αντικειμενοστραφής Προγραμματισμός II**

Διδάσκων: Δούμα Αναστασία

---

### **Ομαδική Εργασία**

---

<3212014092> Κοτσαράπογλου Ιάσων - Ειρηναίος

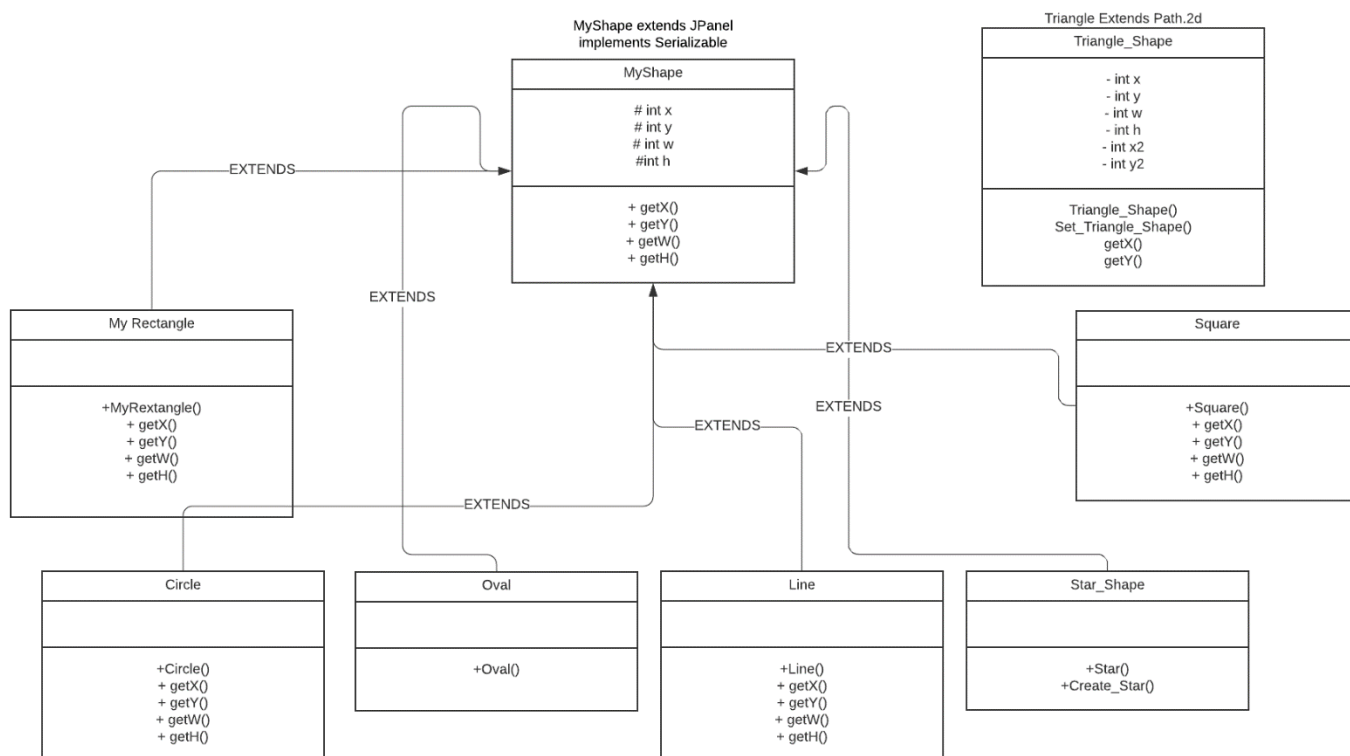
<3212017212> Χαμακιώτη Ελένη

Σάμος, 27/01/2022



## 1<sup>η</sup> φάση υλοποίησης

### UML CLASS DIAGRAM





## Αρχικός Στόχος

Σε πρώτη φάση η ιδέα για την δημιουργία του προγράμματος που μας ζητήθηκε ήταν η εξής :

- ✓ Να φτιάξουμε μια κλάση Paint η οποία θα είναι υπεύθυνη για την δημιουργία του canva (ένα frame δηλαδή) , το οποίο θα είχε ένα menu με τα εργαλεία που θα θέλαμε εμείς να βάλουμε καθώς και μια περιοχή σχεδίασης.
- ✓ Στην συνέχεια θα φτιάχναμε μια κλάση – βάση την οποία ονομάσαμε MyShape η οποία είχε ως ιδιότητες τα κοινά στοιχεία που θα είχαν όλα τα σχήματα (δλδ τις συντεταγμένες , το μήκος και το πλάτος).
- ✓ Έπειτα για κάθε ένα απο τα σχήματα θα κάναμε μια κλάση.
- ✓ Τέλος θα κάναμε και μια κλάση η οποία θα ήταν υπεύθυνη για το save και το load των αρχείων.

Στην συνέχεια παρεθέτουμε τον τρόπο με τον οποίο εν τέλει υλοποιήσαμε το πρόγραμμα , αναλύοντας τα περιεχόμενα κάθε κλάσης και επεξηγώντας.

### Κλάση MyShape:

Για την υλοποίηση των σχημάτων απαιτείται μία βασική κλάση με τα κοινά χαρακτηριστικά όλων των σχημάτων. Τέτοια χαρακτηριστικά είναι:

- int x,y,w,h = οπου αποτελούν τις συντεταγμένες του κάθε σχήματος ,το μήκος και πλάτος του.

Επιπλέον οι κοινές μέθοδοι σε όλα τα σχήματα:

- public int getX() = οπου μας επιστρέφει την πρώτη συντεταγμένη για το κάθε σχήμα
- public int getY() = οπου μας επιστρέφει την δεύτερη συντεταγμένη για το κάθε σχήμα
- 

Για να υπάρχει η δυνατότητα αποθήκευσης των σχημάτων ως αντικειμένων σε αρχείο η κλάση MyShape υλοποιεί το Interface Serializable .

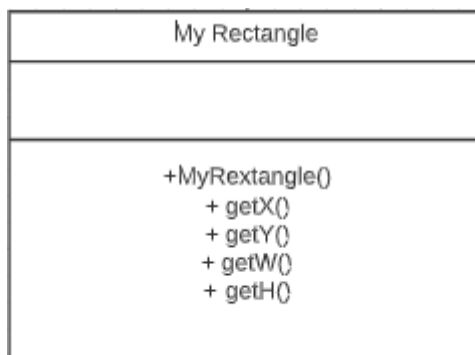
### Κλάση MyRectangle:

Η κλάση αυτή, όπως κάθε σχήμα κληρονομεί από την κλάση MyShape. Περιλαμβάνει τις ιδιότητες οπου έχει κληρονομήσει απο την κλάση βάση :

- int x,y,w,h = οπου αποτελούν τις συντεταγμένες του κάθε σχήματος ,το μήκος και πλάτος του.

και τις μεθόδους :

- int getX()
- int getY()
- int getW()
- int getH()





### Κλάση Square – Κλάση Circle – Κλάση Line – Κλάση Oval:

Οι παραπάνω κλάσεις έχουν υλοποιηθεί με το ίδιο τρόπο όπως και η MyRectangle.

### Κλάση Star Shape:

Όπως και με τις παραπάνω κλάσεις έτσι και αυτή κληρονομεί από την κλάση MyShape. Ωστόσο η Star\_Shape υλοποιεί μια extra μέθοδο αυτήν της Shape createStar(). Είναι μια μέθοδος όπου έχει τύπο επιστροφής Shape και με την βοήθεια της κλάσης Path2D και πιο συγκεκριμένα των μεθόδων moveTo και lineTo μας βοηθάει στον σχηματισμό του αστερίου.

### Κλάση Triangle Shape:

Η κλάση τρίγωνο δεν κληρονομεί από την κλάση MyShape αλλά από την Path2D.double και έχει ως ιδιότητες :

- int x
- int y
- int w
- int h
- int x2
- int y2

Οι ιδιότητες αυτές αναφέρονται στις συντεταγμένες του σχήματος (x,y) το πρώτο σημείο και (x2,y2) το δεύτερο σημείο, ενώ (w,h) αφορούν το μήκος και το πλάτος.

Οι μέθοδοι είναι ίδιοι με όλες τις κλάσεις που αφορούν σχήματα :

- Constructor
- Set
- Get

### Κλάση Paint:

Η κλάση Paint είναι η κλάση στην οποία υλοποιούμε και όλα τα γραφικά του προγράμματος. Η κλάση Paint κάνει extend την κλάση JPanel ( όπου την χρησιμοποιούμε για την δημιουργία του frame, των κουμπιών καθώς και γενικά ότι αφορά το σχεδιαστικό κομμάτι της εργασίας ) καθώς κάνει implement και τα interface MouseListener, MouseMotionListener και ActionListener (όπου είναι υπεύθυνα για την δημιουργία των σχημάτων μέσω του ποντικιού, για να παίρνουμε κάθε φορά τις συντεταγμένες όπου έχουμε το ποντίκι μας (vector) αλλά και για να ξέρουμε πότε ο χρήστης έκανε κλικ και πού ( είτε είναι σε κουμπί είτε είναι σε κάποιο σχήμα ) ).



### Ιδιότητες της κλάσης :

- Color background = αφορά το χρώμα που θέτουμε ως background στον κανβά
- int x1,y1 = Οι συντεταγμένες όπου ο χρήστης πάτησε κλικ (στον κανβά)
- int x2,y2 = Οι συντεταγμένες όπου ο χρήστης άφησε το κλικ (στον κανβά)
- int choice = Μια μεταβλητή που χρησιμοποιούμε για να θέσουμε σε ποιο σχήμα πάτησε ο χρήστης (Βάζουμε ActionListener σε κάθε κουμπί που δημιουργήσαμε και υστερα με την βοήθεια του ActionEvent βλέπουμε ποιο κουμπί πατήσε ο χρήστης και αντίστοιχα θέτουμε το choice)
- Boolean myflag,myflag2 = 2 μεταβλητές που παίζουν τον ρόλο ενός flag. Αν ο χρήστης πατήσε απο τα εργαλεία την επιλογή Paint Border (myflag) ή Fill(myflag2) τότε γίνεται true έτσι ώστε να μπορέσει να γίνει η κατάλληλη λειτουργία
- ArrayList BorderShapes,FillShapes,polys = Λίστες όπου αποθηκεύουμε τα σχήματα ανα κατηγορία . Δηλαδή στα BorderShapes είναι όσα σχήματα έχουν την δυνατότητα να αλλάξουν χρώμα στο περίγραμμα , αντίστοιχα στην λίστα FillShapes είναι όσα σχήματα μπορούν να αλλάξουν χρώμα γεμίσματος και τέλος στην λίστα polys αποθηκεύουμε τα τρίγωνα επειδή δεν κάνουν extend την κλάση MyShape όπως είπαμε παραπάνω .
- HashSet shapeSet = Μια δομή HashSet όπου αποθηκεύουμε μέσα όλα τα σχήματα όπου δημιουργούμε έτσι ώστε αργότερα για την λειτουργία του save δίνουμε την συγκεκριμένη δομή και δημιουργούμε ενα αρχείο, ενώ για το load ανακτούμε απο ένα υπαρχων αρχείο στην συγκεκριμένη δομή αποθήκευσης δεδομένων.

### Μέθοδοι της κλάσης :

- Paint() = ο constructor της κλάσης όπου μέσα δημιουργούμε το frame , ένα JMenuBar καθώς και τα κουμπία αλλα και γενικά το σχεδιαστικό κομμάτι του προγράμματος .
- paintComponent() = μέθοδος η οποία καλείτε μόνη της και είναι υπεύθυνη για τον γραφικό σχεδιασμό των σχημάτων.
- MouseClicked() = μεθοδός που καλείτε οταν ο χρήστης κάνει κάποιο κλικ.
- MousePressed() = μέθοδος για να πάρουμε τις συντεταγμένες όπου ο χρήστης έκανε το κλικ.
- MouseReleased() = μέθοδος για να πάρουμε τις συντεταγμένες όπου ο χρήστης άφησε το κλικ.
- ActionPerformed() = μέθοδος για να δουμε σε ποιά λειτουργία (κουμπί – actionListener) έκανε κλικ ο χρήστης.
- draw() = η μέθοδος η οποία καλείτε για τον σχεδιασμό των σχημάτων.

### Κλάση FileHandler:

Η κλάση FileHandler είναι υπεύθυνη για τις λειτουργίες του save και του load. Περιέχει 2 μεθόδους :

- writecanvas() = μέθοδος όπου δέχεται μια δομή αποθήκευσης δεδομένων HashSet και την μετατρέπει σε αρχείο.
- readcanvas() = μέθοδος η οποία επιστρέφει μια δομή αποθήκευσης δεδομένων HashSet . Η συγκεκριμένη μέθοδος διαβάζει αρχεία απο ενα υπάρχων αρχείο και τα αποθηκεύει μεσα στην HashSet και υστερα την επιστρέφει.



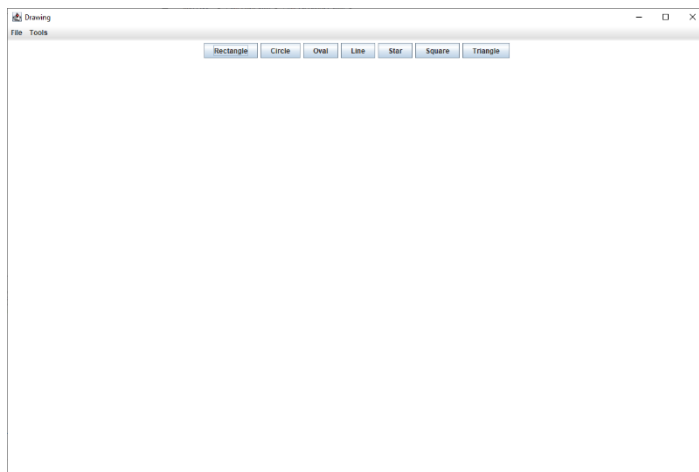
## 2<sup>η</sup> φάση υλοποίησης

### Παραθυρικό Περιβάλλον

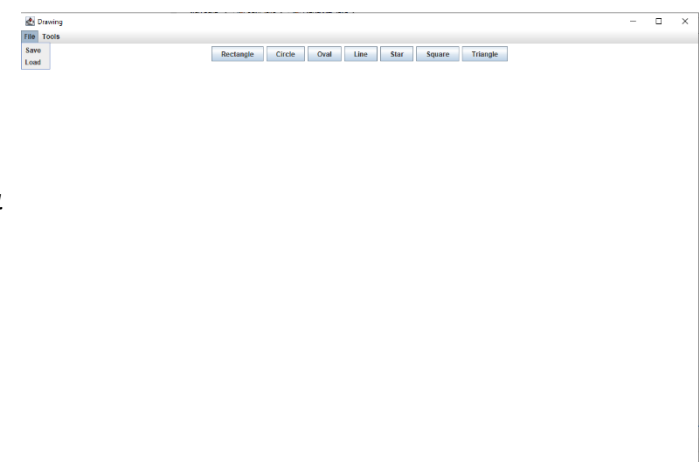
#### Frame:

Το παραθυρικό περιβάλλον υλοποιείται στην κλάση Frame η οποία περιλαμβάνει τα επόμενα βασικά Panel:

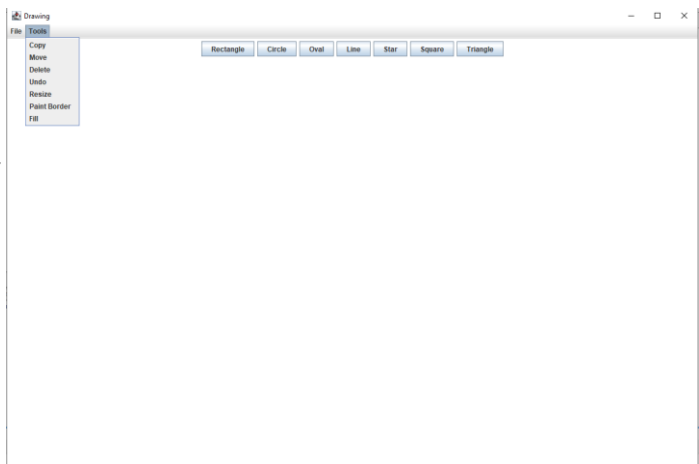
- **Canvas:** Όλη η λευκή περιοχή στο παράθυρο στην οποία γίνεται ο σχεδιασμός των σχημάτων.
- **JMenuBar:** Τα μενού επιλογών για την πρόσβαση στις λειτουργίες των αρχείων ή την πρόσβαση στα εργαλεία.
- **Shape Buttons:** Τα μενού επιλογών για τα σχήματα που μπορεί να διαλέξει ο χρήστης. Στο πάνω μισό υπάρχει μία λίστα στην οποία φαίνονται και μπορούν να επιλεγούν όλα τα σχήματα που έχουν προστεθεί. Ανάλογα με το σχήμα που έχει επιλεγεί, ο χρήστης μπορεί να τροποποιήσει τα χαρακτηριστικά του και πατώντας το αντίστοιχο κουμπί (**tools**) και να εφαρμόσει τις αλλαγές αυτές. Οι αλλαγές περιλαμβάνουν αλλαγή χρώματος, μεγέθους και σημείων. Επιλέγοντας **Αντιγραφή** θα δημιουργηθεί ένα ακριβές αντίγραφο που αρχικά δεν θα φαίνεται γιατί θα είναι πάνω στο άλλο αλλά μπορεί να αυτό να τροποποιηθεί ανάλογα. Η **διαγραφή** διαγράφει το επιλεγμένο σχήμα. Τα τελευταία κουμπιά αφορούν την **αποθήκευση** και **ανάκτηση** από αρχείο. Το παράθυρο διαλόγου στο 3<sup>ο</sup> στιγμιότυπο εμφανίζεται για να βρεθεί το αρχείο για άνοιγμα ή για την επιλογή φακέλου και ονόματος αποθήκευσης.



Εικόνα 1



Εικόνα 2




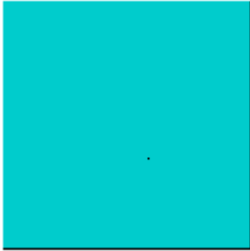
Εικόνα 3



## Σχεδιασμός σχημάτων

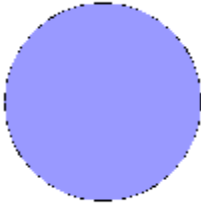

### Ορθογώνιο – Τετράγωνο:

Τα σχήματα ορθογώνιο, τετράγωνο και τρίγωνο αποτελούν πολύγωνα με 4 και 3 γωνίες αντίστοιχα με την διαφορά στο τετράγωνο ότι όλες οι πλευρές είναι ίσες ενώ στο ορθογώνιο οι δύο απέναντι ίσες. Ο τρόπος σχεδιασμού τους είναι ο ίδιος.

<pre>MyRectangle rec = new MyRectangle(x1, y1, w, h); FillShapes.add(rec); BorderShapes.add(rec); shapeSet.add(rec); gc.drawRect(x1, y1, w, h); repaint();</pre>	<pre>... Square square = new Square(x1, y1, w, w); FillShapes.add(square); BorderShapes.add(square); shapeSet.add(square); gc.drawRect(x1, y1, w, w); repaint();</pre>
	
Ορθογώνιο	Τετράγωνο

### Έλλειψη – Κύκλος:

Τα σχήματα έλλειψη και κύκλος σχεδιάζονται με την βοήθεια της μεθόδου drawOval με τον κύκλο να έχει ίσο πλάτος και ύψος.

<pre>gc.drawOval(x1, y1, w, h); gc.drawOval(x1, y1, w, w);</pre>
 



## Σχεδιασμός σχημάτων

```
public void paintComponent(Graphics g) {  
    super.paintComponent(g);  
    Graphics2D g2 = (Graphics2D) g;  
    if (grid == null) {  
        int w = this.getWidth();  
        int h = this.getHeight();  
        grid = (BufferedImage) (this.createImage(w, h));  
        gc = grid.createGraphics();  
        gc.setColor(Color.BLACK);  
    }  
  
    g2.drawImage(grid, null, 0, 0);  
    check();  
}
```

Για τον σχεδιασμό όλων των σχημάτων, καλείται η `paintComponent()` ενώ έχουμε θέσει ως default αρχικό χρώμα το Μαύρο.

## Αρχεία

Όλα τα σχήματα αποθηκεύονται σε ένα αρχείο με το όνομα που θα δώσει ο χρήστης. Αυτός είναι και ο λόγος που υλοποιούν την διεπαφή `Serializable`.

Για το γράψιμο σε αρχείο αποθηκεύονται τα σχήματα όπως αυτά έχουν αποθηκευτεί στο `HashSet`.

```
public static void writecanvas(HashSet<MyShape> set) {  
    //για kathe sxima to grafei sto arxeio  
    String filename = JOptionPane.showInputDialog("Give file name to save");  
    try (FileOutputStream fos = new FileOutputStream(filename);  
        ObjectOutputStream oos = new ObjectOutputStream(fos)) {  
        // write object to file  
        oos.writeObject(set);  
    } catch (IOException ex) {  
        ex.printStackTrace();  
    }  
}
```

Γράψιμο σε αρχείο





Στην περίπτωση ανοίγματος αρχείου, ζητάμε από τον χρήστη το όνομα του αρχείου και αν βρεθεί τότε τα σχήματα προστίθενται ένα ένα όπως αυτά διαβάζονται από το αρχείο. Τέλος επιστρέφουμε το HashSet .

```
public static HashSet<MyShape> readcanvas() {  
    //zitaai apto xristi to arxeio  
    String filename = JOptionPane.showInputDialog("Give file name to read");  
  
    try (InputStream is = new FileInputStream(filename);  
        ObjectInputStream ois = new ObjectInputStream(is)) {  
        return (HashSet<MyShape>) ois.readObject();  
    } catch (IOException | ClassNotFoundException ioe) {  
        ioe.printStackTrace();  
    }  
    //allws epistrefei mia adeia  
    return new HashSet<>();  
}
```

Διάβασμα  
απο αρχείο



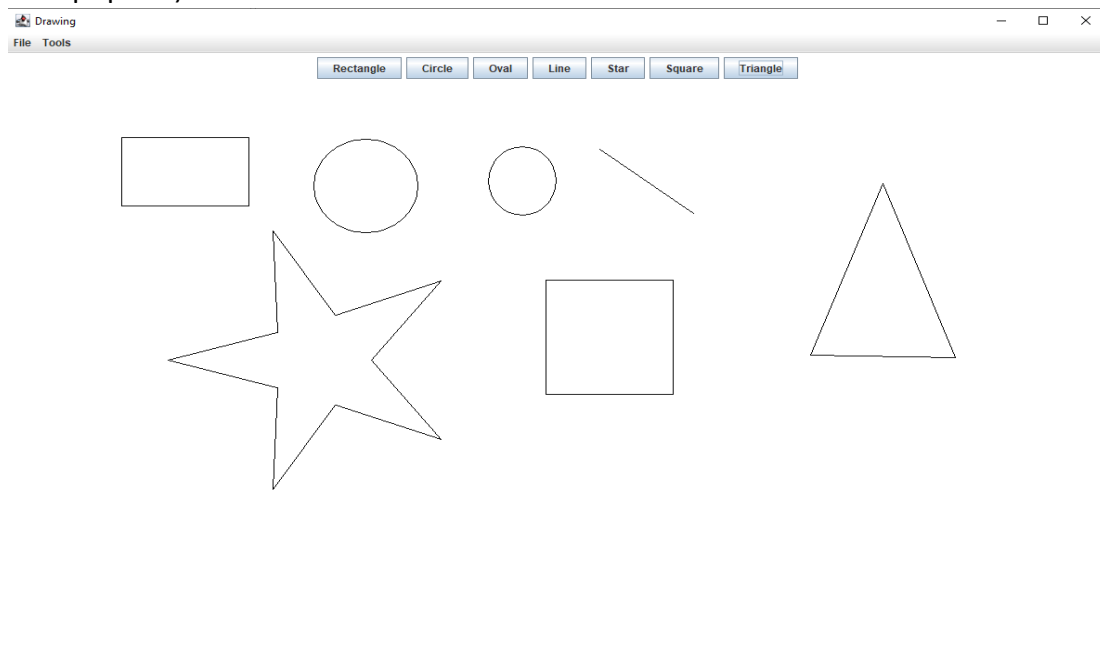
## Ομαδική Εργασία

Τίτλος Μελέτης: Αναφορά Εργασίας

Κοτσαράπογλου Ιάσων - Ειρηανίος, Χαμακιώτη Ελένη

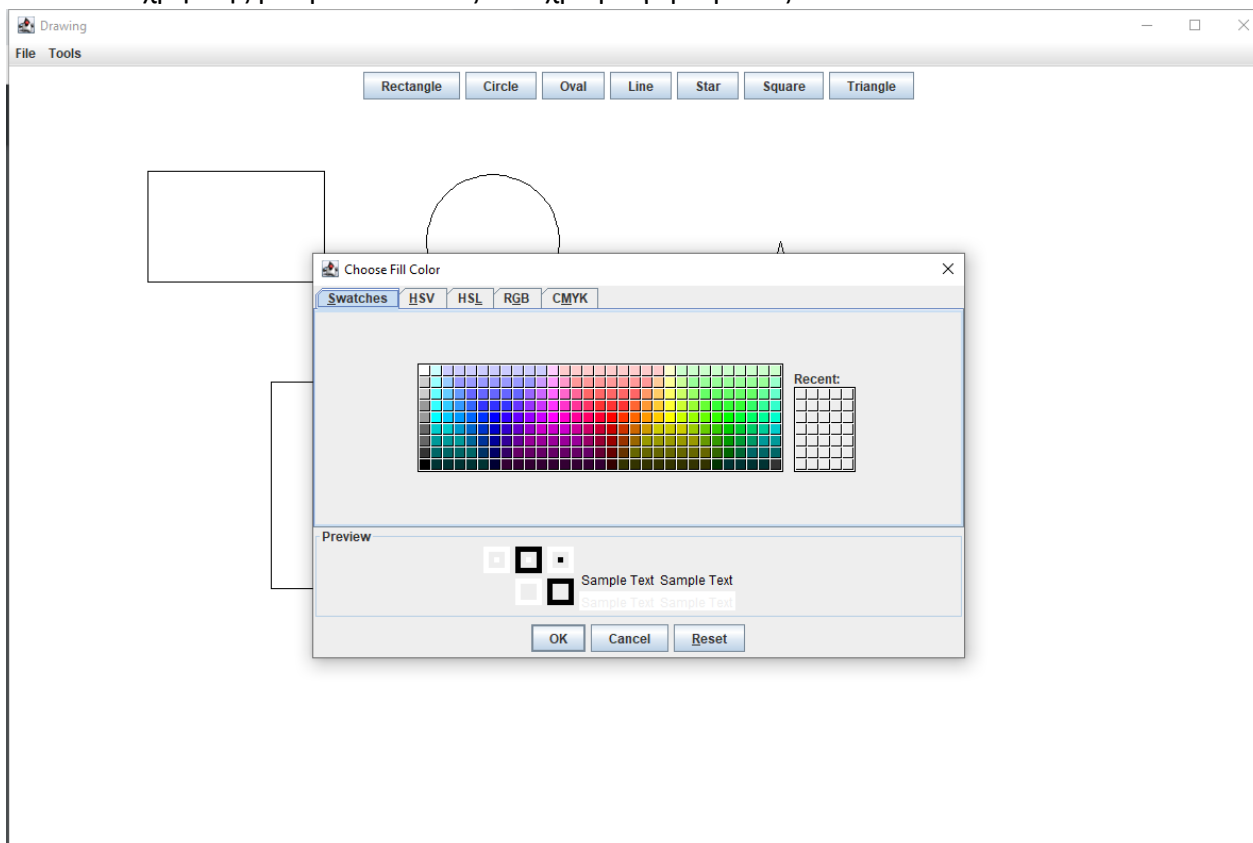
### Σενάριο χρήσης

1. Ο χρήστης ανοίγει την εφαρμογή και μπορεί να αρχίσει κατευθείαν να σχεδιάζει. Επιλέγοντας από το menu με τα σχήματα το σχήμα που θέλει μπορεί κανόντας κλικ στον κανβά και σέρνοντας το ποντίκι του , να σχεδιάζει το σχήμα που θέλει αλλά να καθορίσει και το σημείο που το θέλει και το μέγεθος του .

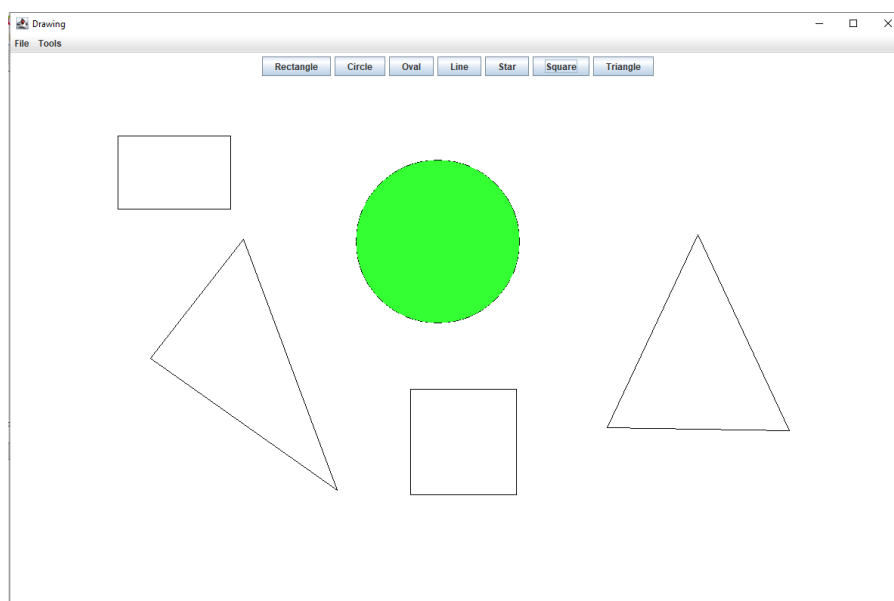




2. Μέσα από το menu εργαλείων (tools) και την επιλογή fill (γέμισμα) , ο χρήστης μπορεί να επιλέξει το χρώμα γεμίσματος.



3. Στην συνέχεια αφού επιλέξει χρώμα μπορεί να διαλέξει ποιο σχήμα θελει να γεμίσει.(Προσοχή! Η επιλογή ισχύει μόνο για ένα σχήμα , αν ο χρήστης θέλει και να γεμίσει και άλλο σχήμα τότε θα πρέπει να επιλέξει πάλι το εργαλείο. )



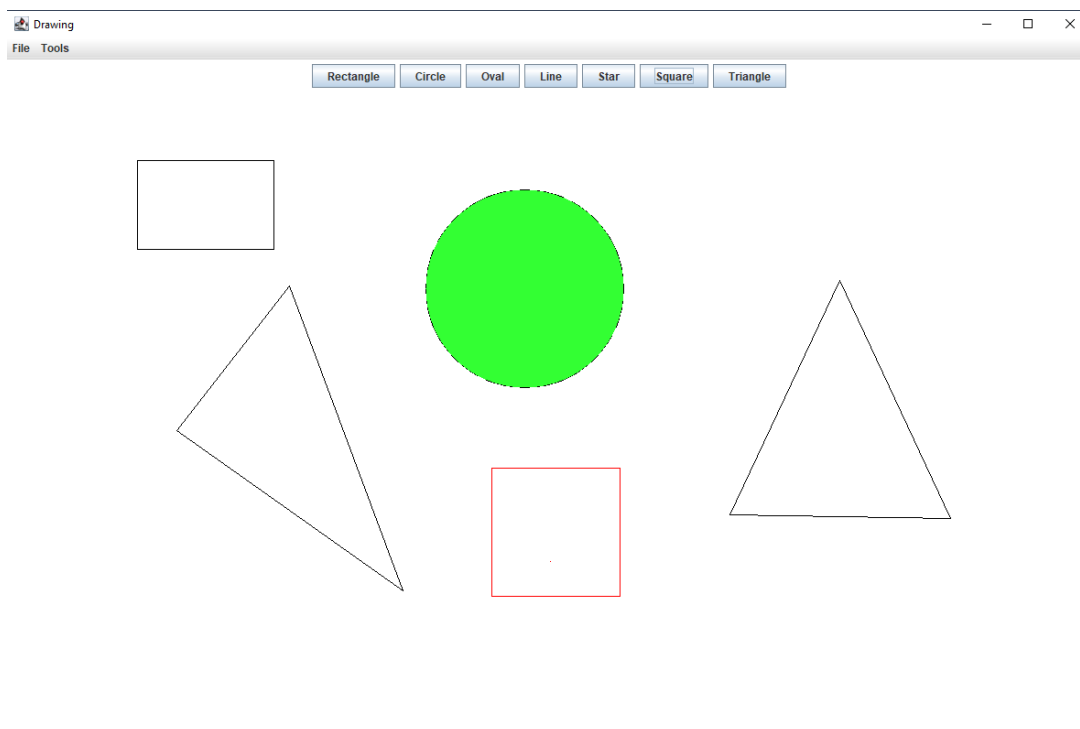


## Ομαδική Εργασία

Τίτλος Μελέτης: Αναφορά Εργασίας

Κοτσαράπογλου Ιάσων - Ειρηανίος, Χαμακιώτη Ελένη

4. Επιλέγοντας την επιλογή Paint Border, ο χρήστης μπορεί να διαλέξει χρώμα περιγράμματος αυτήν την φορά.



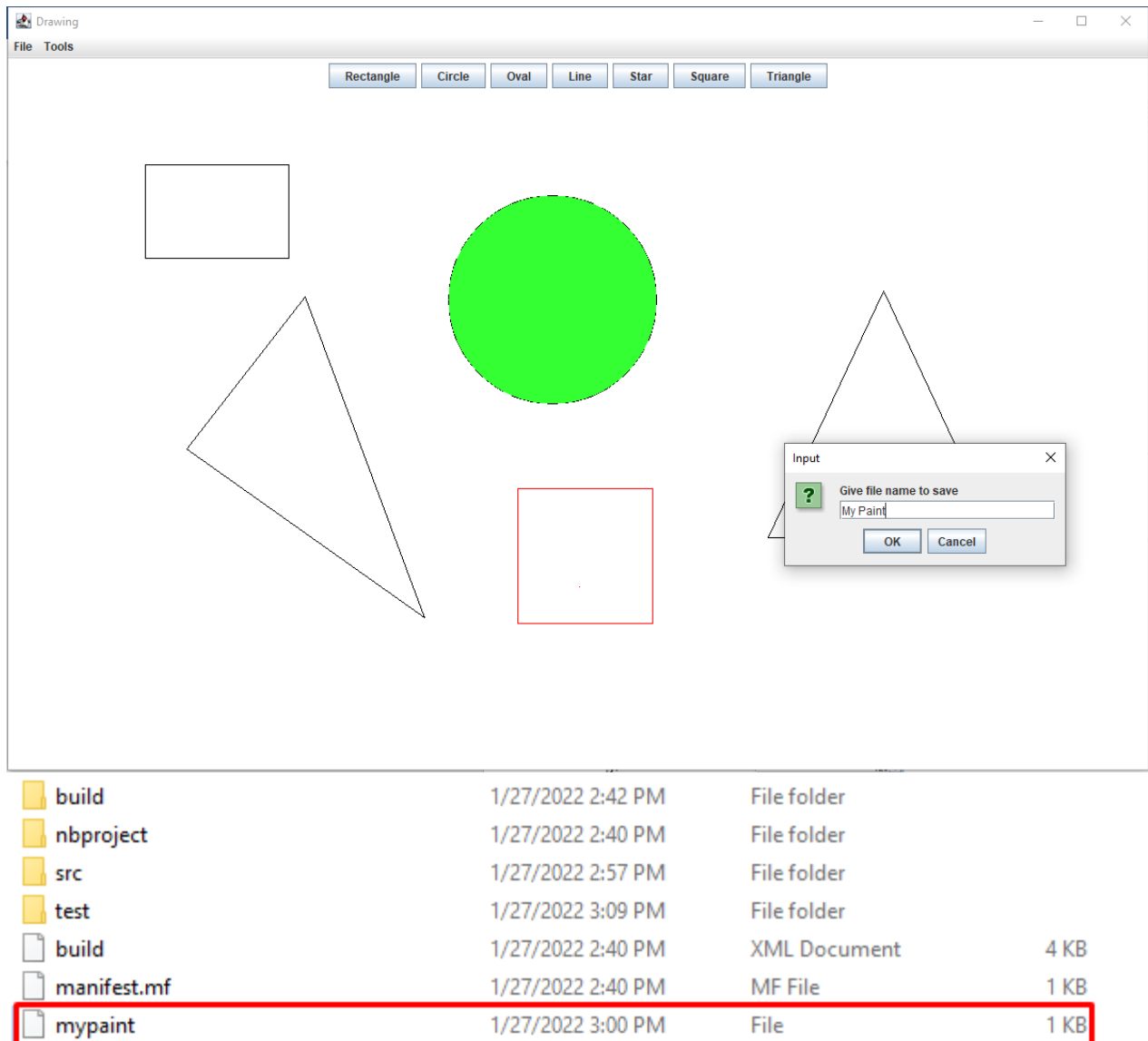


## Ομαδική Εργασία

Τίτλος Μελέτης: Αναφορά Εργασίας

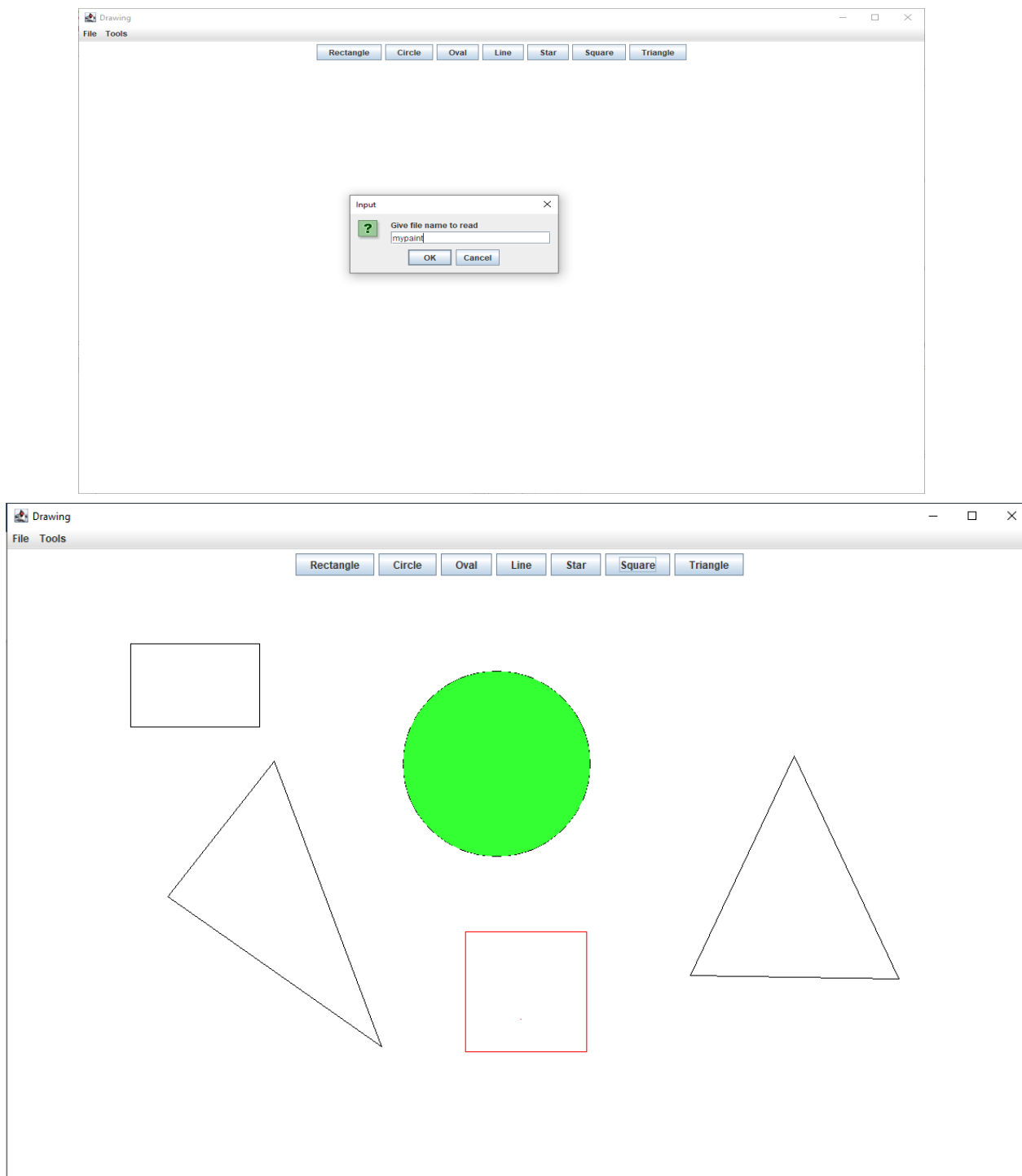
Κοτσαράπογλου Ιάσων - Ειρηανίος, Χαμακιώτη Ελένη

5. Στη συνέχεια ο χρήστης μπορεί να αποθηκεύσει την ζώγραφιά του επιλέγοντας απο το μενού την επιλογή file -> Save. Τότε θα του ζητηθεί ένα όνομα για το αρχείο και αφού πατήσει το ok το αρχείο θα είναι πλέον αποθηκευμένο.





6. Ο χρήστης έχει την δυνατότητα να κάνει και load μια ήδη αποθηκευμένη ζωγραφία πατώντας στο menu και στην συνέχεια file->Load.(Αφού πρώτα του ζητηθεί το όνομα του αρχείου που θέλει να κάνει load)





## Αλλαγές και Προβλήματα

Κατα την δημιουργία του προγράμματος δεν αντιμετωπίσαμε κάποιο πρόβλημα που να αφορά το προγραμματιστικό κομμάτι. Ωστόσο αφού είχαμε υλοποιήσει το πρόγραμμα σε ένα σημαντικό βαθμό ( είχαμε υλοποιήσει ήδη των σχεδίασμο όλων των σχημάτων καθώς και το save και το load) και προχωρώντας στην υλοποίηση των λειτουργιών που μας ζητήθηκαν καταλάβαμε ότι δεν ήταν η σωστή λύση του προγράμματος καθώς δεν μπορούσαμε να υλοποιήσουμε τις υπόλοιπες λειτουργίες . Βάση όσον είδαμε γνωρίζουμε ότι ο σωστός τρόπος υλοποίησης θα ήταν η κλάση MyShape να ήταν abstract και ενδεχομένως η μέθοδος paintComponent() να υλοποιήτε σε κάθε κλάση σχήματος. Ενδεχόμενος και τα σχήματα να έπρεπε να έχουν και 2 μεταβλητές εξτρά (αυτές για το χρώμα σχεδίασης και το χρώμα γεμίσματος) σε όποιο σχήμα χρειαζόταν η κάθε μια . Εμείς λόγω έλλειψης χρόνου ( εμείς φταίμε που το αφήσαμε για τελευταία στιγμή) και λόγω ότι όταν πήγαμε να την κάνουμε τελευταία στιγμή κολλήσαμε και οι 2 covid (έχουμε τα πιστοποιητικά νόσησης) δεν προλαβαίναμε να την υλοποιήσουμε. Οπότε εν τέλει υλοποιήσαμε και τις 2 λειτουργίες paint border και fill πανώ σε αυτό το πρόγραμμα που ήδη είχαμε .

## Επέκταση Μελλοντικά

Η προφανής επέκταση μελλοντικά θα ήταν να υλοποιήσουμε το πρόγραμμα σωστά απο την αρχή και έτσι να υλοποιήσουμε και τις υπολοιπές λειτουργίες που μας λείπουν.