Υ325 - Αντικειμενοστρεφής Τεχνολογία

2021

Τμ. ΗΜ&ΤΥ - Πανεπιστήμιο Πατρών

Εβδομάδα 8 - 02/12/2021

Java GUI programming - Astra game

Eclipse clone project

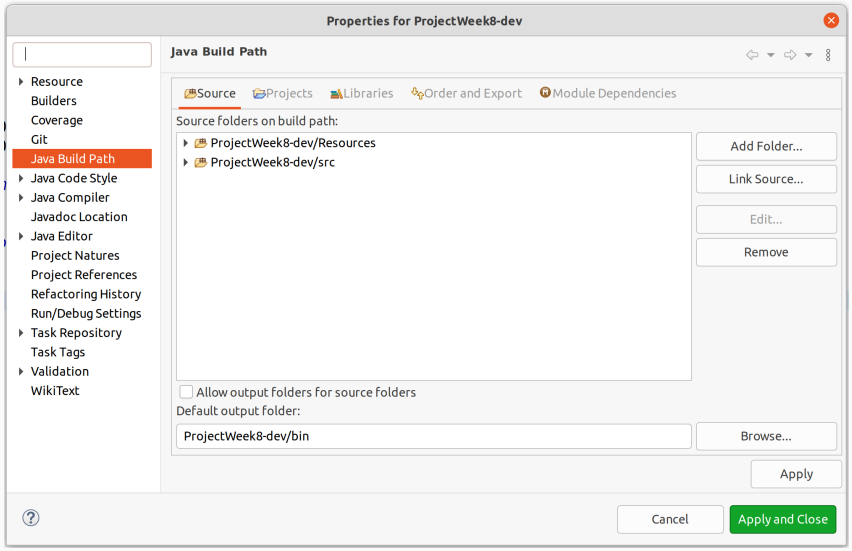
Ξεκινάμε από το παραδοτέο μας (ή την ενδεικτική λύση) της εβδομάδας #7

- Αντιγράφουμε (`Ctrl-C`) το `ProjectWeek7` και κάνουμε επικόλληση
  - Η επικόλληση γίνεται στον `Project Explorer` του Eclipse
- Ονομάστε το νέο πρότζεκτ `ProjectWeek7-1099999` (αν 1099999 είναι ο ΑΜ σας)
- `F11` για Compile and Run για να βεβαιωθείτε πως το πρότζεκτ τρέχει.

Resources και φόρτωμα εικόνων

Στα Project properties (προσβάσιμα από το μενού `Project ->Properties` ή με δεξί κλικ στο πρότζεκ μας) μεταβείτε στην καρτέλα `Java Build Path ->Source`.

Παρατηρήστε πως έχει δηλωθεί σαν πηγή και ο φάκελος `Resources`.

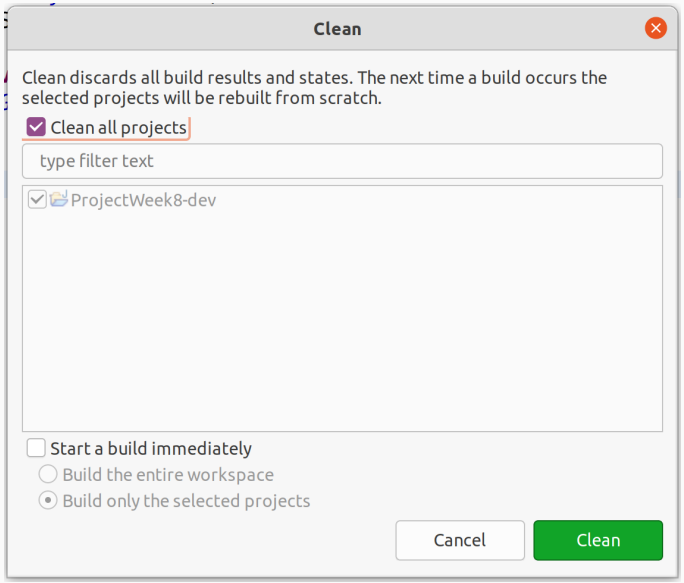


Εικόνα 1: `Project ->Properties`

Resources και φόρτωμα εικόνων 2

Κάντε το εξής πείραμα:

1. Καθαρίστε τον φάκελο `bin`/, που περιέχει τα αποτελέσματα της προηγούμενης μεταγλώττισης, δηλ. τα class files.
    - Από το μενού `Project ->Clean...` (απενεργοποιήστε την επιλογή Start a build immediately)



Εικόνα 2: `Project ->Clean...`

Resources και φόρτωμα εικόνων 3

2. Ανοίξτε το φάκελο `bin/` από τον εξερευνητή αρχείων για να επιβεβαιώσετε πως είναι πλέον άδειος.
3. Στη συνέχεια μεταγλωττίστε ξανά το πρότζεκτ, είτε με `Project ->Build project` είτε με `Ctrl-F11` (ή `Run ->Run`),
4. Δείτε πάλι τον φάκελο `bin/` από τον εξερευνητή αρχείων. Θα παρατηρήσετε πως υπάρχουν πλέον εκεί τα αποτελέσματα της μεταγλώττισης (τα αρχεία `.class`) και τα περιεχόμενα του φακέλου `Resources/`.

Resources και φόρτωμα εικόνων 4

Συμπέρασμα: για τον κώδικα που βρίσκεται π.χ. στο αρχείο `spaceships/SpaceShipALPHA.class`, το αρχείο `images/ALPHA.png` βρίσκεται στο σχετικό φάκελο `../images/ALPHA.png`.

Οπότε στην κλάση `spaceships.SpaceShipALPHA.java` μπορούμε να γράψουμε:

```
    SpaceShipALPHA.img = ImageIO.read(MainClass.class.
        getResource("../images/ALPHA.png"));
```

Painting

Painting

Τα γραφικά στοιχεία AWT και Swing εμφανίζονται στην οθόνη με μια ενέργεια που ονομάζεται *painting*: Ζωγραφίζουν τον εαυτό τους.

Η διαδικασία αυτή μπορεί να προκληθεί με δύο τρόπους (περιγράφεται στο Painting in AWT and Swing):

1. Το σύστημα ζητάει από το γρ. στοιχείο να ξαναζωγραφιστεί είτε γιατί εμφανίζεται στην οθόνη για πρώτη φορά, είτε γιατί αλλάζει μέγεθος, είτε γιατί κάποιο τμήμα του δεν εμφανίζεται σωστά (π.χ. ένα τμήμα του καλυπτόταν από κάποιο άλλο παράθυρο το οποίο μετακινήθηκε).
2. Η ίδια η εφαρμογή ζητά να ξαναζωγραφιστεί το γραφικό στοιχείο, γιατί κάτι άλλαξε στην εσωτερική κατάσταση (state) της εφαρμογής (π.χ. το διαστημόπλοιο άλλαξε θέση, ή πέρασε κάποιος χρόνος και το λέιζερ πλησίασε τον εχθρό).

Painting - η μέθοδος paint()

Το ζωγράφισμα του γρ. στοιχείου το αναλαμβάνει η μέθοδος `paint()`. Όλες οι κλάσεις που κληρονομούν από την `java.awt.Component` μπορούν να αναιρέσουν (*override*) τη μέθοδο `paint()`.

- Αν το σύστημα διαπιστώσει πως χρειάζεται το γραφικό στοιχείο να ζωγραφίσει τον εαυτό του, τότε το ίδιο το σύστημα καλεί την `paint()` αυτού του στοιχείου.
- Αν ο προγραμματιστής αλλάξει την εσωτερική κατάσταση (state) της εφαρμογής, μπορεί να καλέσει την `paint()` ρητά. Αλλά:
  > "It is not recommended that programs invoke paint() directly." *(Painting in AWT and Swing)*

Painting - repaint()

Για να προκαλέσουμε ρητά ένα repaint, δηλ. για να ζητήσει ο προγραμματιστής από το γραφικό στοιχείο να ξαναζωγραφιστεί, καλεί τη μέθοδο `repaint()`. Η `repaint()` έρχεται σε διάφορες εκδοχές:

- `public void repaint()` *`Repaints this component`*.
- `public void repaint(long tm)` *`Repaints the component within tm msec`*.
- `public void repaint(int x, int y, int width, int height)`
  `Repaints the specified rectangle of this component`.
- `public void repaint(long tm, int x, int y, int width, int height)`
  `Repaints the specified rectangle of this component within tm msec`.

Έτσι μπορούμε να ζητήσουμε να ζωγραφιστεί ξανά μόνο μια μικρή περιοχή του στοιχείου, πχ. με την
`repaint(int x, int y, int width, int height)`, και όχι όλο από την αρχή, όπως με τη `repaint()`.

Painting - η μέθοδος paintComponent()

Αν και μπορούμε να αναιρέσουμε (*override*) την `paint()`, αυτό δε συνίσταται - εκτός αν ξέρουμε πολύ καλά τί κάνουμε.

> "Swing programs should override paintComponent() instead of overriding paint()." *(Painting in AWT and Swing)*

Μπορούμε να αναιρέσουμε (*override*) την `paintComponent()`, η οποία είναι μια από τις μεθόδους που θα καλέσει η `paint()`. Η `paint()` με τη σειρά της καλεί τις μεθόδους:

- `protected void paintComponent(Graphics g)`
- `protected void paintBorder(Graphics g)`
- `protected void paintChildren(Graphics g)`

Τις δύο τελευταίες συνήθως δεν τις αναιρούμε (*override*), εκτός αν ξέρουμε πολύ καλά τί κάνουμε.

Painting - παράδειγμα

```
public class DemoPanel extends JPanel {
    // το Graphics object περιέχει πληροφορία για τις βασικές
    // δυνατότητες ζωγραφικής της Java (γραμματοσειρά, χρώμα...)
    @Override
    public void paintComponent(Graphics g) {
        super.paintComponent(g);
        g.setFont(new Font("Arial", Font.PLAIN, 22));
        // Ζωγράφισε ένα κείμενο
        g.drawString("Το δικόμου  Panel!", 10, 20);
    }
}
```

Painting - παράδειγμα 2

```
public class DemoPanel extends JPanel {
    @Override
    public void paintComponent(Graphics g) {
        super.paintComponent(g);
        // Η Dimension αφορά το μέγεθος ενός γραφικού στοιχείου
        Dimension size = getSize(); // το μέγεθος του παραθύρου
        int d = Math.min(size.width, size.height);
        //στο κέντρο του παραθύρου και στους δύο άξονες
        int x = (size.width - d) / 2;
        int y = (size.height - d) / 2;
        // Ζωγραφίζουμε έναν κύκλο
        g.setColor(Color.black);
        g.fillOval(x, y, d, d);
        g.setColor(Color.red);
        //το Graphics2D μπορεί να κάνει περισσότερα
        Graphics2D g2 = (Graphics2D) g;
        g2.setStroke(new BasicStroke(10));
        g2.drawOval(x, y, d, d);
    }
}
```

Painting - παράδειγμα 3

Στο eclass της εβδομάδας 8, υλοποιούμε τη διεπαφή `KeyListener` για να ξαναζωγραφίζεται όταν πατάμε κουμπιά στο πληκτρολόγιο.

Timer

Timer και TimerTask

Με τις δύο κλάσεις `Timer` και `TimerTask` μπορούμε να χρονοπρογραμματίσουμε εργασίες που θα γίνουν στο υπόβαθρο (σε ξεχωριστό νήμα)

- `TimerTask` είναι η εργασία που θέλουμε να γίνει
- `Timer` είναι ο χρόνος στον οποίο θέλουμε να γίνει (ο χρονοπρογραμματισμός)

Timer και TimerTask #2

```
//η επαναλαμβανόμενη εργασία θα είναι να τυπώνει ένα 'τικ'
class MonitorDeamonGame extends TimerTask {
    public void run() {
        repaint();
    }
}
private void createDaemon() {
    TimerTask task = new MonitorDeamonGame();
    Timer timer = new Timer();
    //ξεκίνα σε 500ms και μετά τρέξε κάθε 100ms
    timer.schedule(task, 500, 100);
}
```

Painting - παράδειγμα 4

Στο eclass της εβδομάδας 8, ξαναζωγραφίζουμε χρησιμοποιώντας τα `TimerTask` και `Timer`.

Demo