

ΤΕΧΝΟΛΟΓΙΚΟ
ΕΚΠΑΙΔΕΥΤΙΚΟ
Ι Δ Ρ Υ Μ Α



ΠΕΛΟΠΟΝΝΗΣΟΥ

ΤΜΗΜΑ ΜΗΧΑΝΙΚΩΝ ΠΛΗΡΟΦΟΡΙΚΗΣ Τ.Ε.

ΣΧΟΛΗ ΤΕΧΝΟΛΟΓΙΚΩΝ ΕΦΑΡΜΟΓΩΝ (ΕΔΡΑ: ΣΠΑΡΤΗ)

Τ.Ε.Ι. ΠΕΛΟΠΟΝΝΗΣΟΥ

ΑΝΑΠΤΥΞΗ ΔΙΑΔΡΑΣΤΙΚΟΥ ΗΛΕΚΤΡΟΝΙΚΟΥ ΠΑΙΧΝΙΔΙΟΥ ΓΡΙΦΩΝ ΣΕ ΠΕΡΙΒΑΛΛΟΝ ANDROID

ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ

Αναγνωστόπουλος Ορέστης

ΑΜ: 2012123

Επιβλέπων Καθηγητής
Θανόπουλος Αριστομένης

Σπάρτη 2018

ΔΗΛΩΣΗ ΜΗ ΛΟΓΟΚΛΟΠΗΣ ΚΑΙ ΑΝΑΛΗΨΗΣ ΠΡΟΣΩΠΙΚΗΣ ΕΥΘΥΝΗΣ

"Με πλήρη επίγνωση των συνεπειών του νόμου περί πνευματικών δικαιωμάτων, δηλώνω ενυπογράφως ότι είμαι αποκλειστικός συγγραφέας της παρούσας Πτυχιακής Εργασίας, για την ολοκλήρωση της οποίας κάθε βοήθεια είναι πλήρως αναγνωρισμένη και αναφέρεται λεπτομερώς στην εργασία αυτή. Έχω αναφέρει πλήρως και με σαφείς αναφορές, όλες τις πηγές χρήσης δεδομένων, απόψεων, θέσεων και προτάσεων, ιδεών και λεκτικών αναφορών, είτε κατά κυριολεξία είτε βάση επιστημονικής παράφρασης.

Αναλαμβάνω την προσωπική και ατομική ευθύνη ότι σε περίπτωση αποτυχίας στην υλοποίηση των ανωτέρω δηλωθέντων στοιχείων, είμαι υπόλογος έναντι λογοκλοπής, γεγονός που σημαίνει αποτυχία στην Πτυχιακή μου Εργασία και κατά συνέπεια αποτυχία απόκτησης του Τίτλου Σπουδών, πέραν των λοιπών συνεπειών του νόμου περί πνευματικών δικαιωμάτων.

Δηλώνω, συνεπώς, ότι αυτή η Πτυχιακή Εργασία προετοιμάστηκε και ολοκληρώθηκε από εμένα προσωπικά και αποκλειστικά και ότι, αναλαμβάνω πλήρως όλες τις συνέπειες του νόμου στην περίπτωση κατά την οποία αποδειχθεί, διαχρονικά, ότι η εργασία αυτή ή τμήμα της δε μου ανήκει διότι είναι προϊόν λογοκλοπής άλλης πνευματικής ιδιοκτησίας."

Όνομα και Επώνυμο Συγγραφέα (Με Κεφαλαία):

Υπογραφή (Ολογράφως, χωρίς μονογραφή):

Ημερομηνία (Ημέρα – Μήνας – Έτος):

Πίνακας Περιεχομένων

ΠΕΡΙΛΗΨΗ	
ΕΥΧΑΡΙΣΤΙΕΣ	
1. ΕΙΣΑΓΩΓΗ	1
1.1 ΣΤΟΧΟΣ ΤΗΣ ΠΤΥΧΙΑΚΗΣ	1
1.2. ΣΤΟΧΟΣ ΤΗΣ ΕΦΑΡΜΟΓΗΣ	1
1.3. ΔΟΜΗ ΤΗΣ ΠΤΥΧΙΑΚΗΣ	1
2. ΑΝΑΠΤΥΞΗ ΨΥΧΑΓΩΓΙΚΩΝ ΕΦΑΡΜΟΓΩΝ ΣΕ ΠΕΡΙΒΑΛΛΟΝ ANDROID ΜΕ JAVA	2
2.1. ΤΑ MOBILE GAMES	2
2.2. ΤΑ MOBILE PUZZLE GAMES	3
2.3. Η ΓΛΩΣΣΑ ΠΡΟΓΡΑΜΜΑΤΙΣΜΟΥ JAVA	3
2.4. ΤΟ ΛΕΙΤΟΥΡΓΙΚΟ ΣΥΣΤΗΜΑ ANDROID (ANDROID OS)	4
2.5. ΤΟ ANDROID STUDIO	5
2.5.1. ΧΡΗΣΗ ΟΛΟΚΛΗΡΩΜΕΝΩΝ ΠΕΡΙΒΑΛΛΟΝΤΩΝ ΑΝΑΠΤΥΞΗΣ	5
2.5.2. ΤΙ ΕΙΝΑΙ ΤΟ ANDROID STUDIO	5
2.5.3. ΠΛΕΟΝΕΚΤΗΜΑΤΑ ΧΡΗΣΗΣ	6
3. ΣΧΕΔΙΑΣΗ ΚΑΙ ΥΛΟΠΟΙΗΣΗ ΤΗΣ ΕΦΑΡΜΟΓΗΣ	7
3.1. ΓΕΝΙΚΑ: ΤΙ ΘΑ ΧΡΕΙΑΣΤΟΥΜΕ ΓΙΑ ΤΗΝ ΥΛΟΠΟΙΗΣΗ	7
3.2. ΠΕΡΙΓΡΑΦΗ ΤΗΣ ΥΛΟΠΟΙΗΣΗΣ ΒΗΜΑ ΠΡΟΣ ΒΗΜΑ.....	8
3.2.1. ΠΡΟΕΤΟΙΜΑΖΟΝΤΑΣ ΤΙΣ ΕΙΚΟΝΕΣ	8
3.2.2. ΔΗΜΙΟΥΡΓΩΝΤΑΣ ΤΟ PROJECT	11
3.2.3. ΟΡΙΣΜΟΣ ΑΡΧΙΚΩΝ ΤΙΜΩΝ	14
3.2.4. ΔΗΜΙΟΥΡΓΙΑ GestureDetectGridView	20
3.2.5. ΔΗΜΙΟΥΡΓΙΑ CustomAdapter	29
3.2.6. ΔΗΜΙΟΥΡΓΙΑ Start Screen	45
3.2.7. ΔΗΜΙΟΥΡΓΙΑ About	54
3.2.8. ΕΠΕΞΕΡΓΑΣΙΑ AndroidManifest.xml	56
3.2.9. ΕΙΣΑΓΩΓΗ Icon ΕΦΑΡΜΟΓΗΣ	58
3.3. ΕΚΤΕΛΕΣΗ-ΔΟΚΙΜΕΣ-ΑΛΛΑΓΕΣ	61
4. ΣΥΜΠΕΡΑΣΜΑΤΑ	71
ΒΙΒΛΙΟΓΡΑΦΙΑ/ΙΣΤΟΓΡΑΦΙΑ	72
ΑΝΑΦΟΡΕΣ/ΠΑΡΑΠΟΜΠΕΣ	73
ΠΑΡΑΡΤΗΜΑ: ΚΩΔΙΚΑΣ ΕΦΑΡΜΟΓΗΣ	74

Πίνακας Εικόνων

Εικόνα που χρησιμοποιήθηκε για το puzzle	8
Η εικόνα μετά τον τεμαχισμό	9
Εικονίδιο εφαρμογής	10
Background αρχικής οθόνης.....	10
Δημιουργία project	11
Επιλογή API.....	12
Επιλογή Empty Activity.....	13
Ονομασία νέας Activity	13
Drawable μετά την αντιγραφή των εικόνων	14
Αρχική οθόνη της MainActivity.java	15
Δήλωση στηλών & διαστάσεων	16
Αρχικοποίηση λίστας πλακιδίων	17
Δημιουργία ανακατέματος.....	18
Αρχική κλήση και ορισμός εμφάνισης	19
Δημιουργία κλάσης GestureDetectorGridView	20
Δημιουργία constructors GridView 1/2.....	21
Δημιουργία constructors GridView 2/2.....	21
Εισαγωγή API σε τελευταίο constructor	22
Δήλωση ελεγκτή και σταθερών	24
Δήλωση boolean και float	25
Αρχικοποίηση συμβάντων κίνησης ενός ελεγκτή	26
Οι δυο boolean onTouchEvent	27
Δήλωση και αρχικοποίηση GridView	28
Δημιουργία κλάσης CustomAdapter	29
Δημιουργία constructors BaseAdapter 1/2.....	30
Δημιουργία constructors BaseAdapter 2/2.....	30
Τροποποίηση των constructors εκτός του getView	31
Rendering κουμπιών μέσα σε μέθοδο εμφάνισης.....	32
Δημιουργία μεθόδου ορισμού διαστάσεων μέσω ViewTreeObserver	33
Απολαβή πληροφοριών ύψους μπάρας Android αναλόγως συσκευής.....	34
Παραμετροποίηση αντάπτορα 1/2	35
Παραμετροποίηση αντάπτορα 2/2	36
Προσθήκη επιπλέον παραμέτρων.....	37
Δηλώσεις GridView και Strings, καθώς και αφαίρεση κλήσης εμφάνισης από onCreate	38
Μέθοδος εναλλαγής	39
Μέθοδος εναλλαγής πλακιδίων.....	40
Προσθήκη εναλλαγής πλακιδίων στην αρχικοποίηση του ελεγκτή.....	41
Δημιουργία μεθόδου λύσης.....	42
Πριν την ενημέρωση του activity_main.xml.....	43
Μετά την ενημέρωση του activity_main.xml	44
Προσθήκη εικόνας ως background	45

Μετά την προσθήκη background	46
Προσθήκη κουμπιών	47
Επιλογή κουμπιών και δημιουργία συνδέσεων/περιορισμών	48
Μετά τις συνδέσεις	49
Προσθήκη εντολής για να κάνουμε τα κουμπιά αόρατα	50
Μετά τη διαφάνεια των κουμπιών	51
Αρχική κλάση StartMenu.java	52
Προσθήκη προθέσεων ανοίγματος άλλων activities με τα πατήματα των δυο κουμπιών.....	53
Αρχικό drag & drop TextViews	54
Παραμετροποίηση κειμένων και ορισμός συνδέσεων μεταξύ των Views.....	55
AndroidManifest.xml προ επεξεργασίας.....	56
AndroidManifest.xml μετά επεξεργασίας	57
Εισαγωγή νέου Image Asset	58
Επιλογή Icon	59
Τελική οθόνη ενημέρωσης των αλλαγών.....	60
APK Build	61
Prompt εύρεσης τοποθεσίας ή ανάλυσης	62
Αρχική οθόνη	63
Σελίδα About	64
Εισαγωγή σε σελίδα Puzzle	65
Μήνυμα λάθους	66
Επίλυση puzzle	67
Μήνυμα συγχαρητηρίων.....	68
Λανθασμένη στοιχίση TextView.....	69
Πλαγιασμένη Αρχική Οθόνη.....	70

ΠΕΡΙΛΗΨΗ

Στην παρούσα πτυχιακή θα ασχοληθούμε με τη δημιουργία ενός Puzzle Game για λειτουργικό σύστημα Android (έκδοσης 4.4 και άνω) με την χρήση Android Studio IDE και γλώσσα προγραμματισμού Java. Θα αναφερθούμε γενικά στην ιδέα πίσω από το παιχνίδι, αναλύοντας κάθε στάδιο μέχρι και την ολοκλήρωσή του. Θα δούμε λίγα πράγματα για την γλώσσα προγραμματισμού Java, το λειτουργικό σύστημα Android, καθώς και για τα mobile games και τα παιχνίδια παζλ. Επίσης, θα παρουσιαστεί ο κώδικας που χρησιμοποιήθηκε για να δημιουργηθεί το παιχνίδι. Θα αναλυθεί κάθε κλάση και οι λειτουργίες της ξεχωριστά. Τέλος, θα δούμε πως λειτουργεί το παιχνίδι σε περιβάλλον Android.

ΕΥΧΑΡΙΣΤΙΕΣ

Καταρχάς, θα ήθελα να ευχαριστήσω τους γονείς μου για την ευκαιρία που μου έδωσαν να βρίσκομαι εδώ και να πραγματοποιώ τα όνειρά μου, στηρίζοντάς με σε κάθε στροφή της ζωής μου.

Επίσης θα ήθελα να ευχαριστήσω το Τμήμα, τους καθηγητές που εμπλούτισαν τις γνώσεις μου τα τελευταία χρόνια, και τους φίλους μου οι οποίοι ήσαν εκεί στις καλές, αλλά και στις κακές στιγμές μου.

Τέλος, θέλω να ευχαριστήσω ιδιαίτερω τον επιβλέπων καθηγητή μου, Αριστομένη Θανόπουλο, για τη στήριξη και τη βοήθειά του για το πέρας αυτής της Πτυχιακής Εργασίας, αλλά και για τις γνώσεις που μου προσέφερε αυτά τα χρόνια.

1. ΕΙΣΑΓΩΓΗ

1.1. ΣΤΟΧΟΣ ΤΗΣ ΠΤΥΧΙΑΚΗΣ

Ο στόχος της συγκεκριμένης Πτυχιακής Εργασίας είναι να δείξουμε πως δημιουργείται και αναπτύσσεται βήμα-βήμα μια εφαρμογή με χρήση γλώσσας προγραμματισμού Java, για Android περιβάλλοντα. Σκοπό έχει να διδάξει κάποια βασικά βήματα για τη δημιουργία εφαρμογών, καθώς και να δείξει τρόπους επίλυσης ορισμένων διάφορων προβλημάτων τα οποία μπορεί να εμφανιστούν κατά την υλοποίηση ενός τέτοιου project.

1.2. ΣΤΟΧΟΣ ΤΗΣ ΕΦΑΡΜΟΓΗΣ

Ο στόχος της εφαρμογής που θα υλοποιηθεί παρακάτω, είναι η επίλυση ενός παζλ, το οποίο ξεκινάει με 9 ανακατεμένα πλακίδια, τα οποία προσπαθεί ο χρήστης να σύρει, ώστε να ολοκληρώσει την εικόνα η οποία είναι διασκορπισμένη. Εφόσον ο χρήστης νικήσει, εμφανίζεται ένα κείμενο νίκης. Εάν ο χρήστης κάνει κάποια παράνομη κίνηση, εμφανίζεται ένα κείμενο μη έγκυρης κίνησης.

Η αρχική οθόνη της εφαρμογής έχει 2 κουμπιά, το Start και το About. Με το πάτημα του Start αρχίζει η επίλυση του παζλ, ενώ με το πάτημα του About εμφανίζονται πληροφορίες για την εφαρμογή, όπως η έκδοση.

1.3. ΔΟΜΗ ΤΗΣ ΠΤΥΧΙΑΚΗΣ

Αρχικά στην Πτυχιακή Εργασία εξηγούνται μερικές έννοιες, όπως π.χ. τι είναι το Android Studio ή η Java (Κεφάλαιο 2), αναφερόμαστε στο τι θα χρειαστούμε για να δημιουργήσουμε μια εφαρμογή (Κεφάλαιο 3.1), καθώς και υπάρχει επεξήγηση της υλοποίησης της εφαρμογής βήμα-βήμα (Κεφάλαιο 3.2). Εν συνεχεία, υπάρχουν παραδείγματα εκτέλεσης, αλλά και πιθανές αλλαγές/μελλοντικές επεκτάσεις που θα μπορούσαν να υλοποιηθούν (Κεφάλαιο 3.3). Τέλος, αναφέρονται τα συμπεράσματα τα οποία βγήκαν μέσω της εργασίας (Κεφάλαιο 4) και παρατίθενται η Βιβλιογραφία, οι Αναφορές/Παραπομπές καθώς και ο Κώδικας που χρησιμοποιήθηκε για την υλοποίηση της στα ομώνυμα κεφάλαια.

2. ΑΝΑΠΤΥΞΗ ΨΥΧΑΓΩΓΙΚΩΝ ΕΦΑΡΜΟΓΩΝ ΣΕ ΠΕΡΙΒΑΛΛΟΝ ANDROID ΜΕ JAVA

2.1. ΤΑ MOBILE GAMES

Ένα mobile game είναι ένα βιντεοπαιχνίδι το οποίο παίζεται σε τηλέφωνο, smartphone, tablet, smartwatch, ή σε οποιαδήποτε άλλη φορητή συσκευή, εκτός των κονσόλων.

Το πρώτο mobile game ήταν μια παραλλαγή του Tetris σε μια συσκευή του 1994.

Το 1997, η Nokia λανσάρισε το επιτυχημένο Snake(φιδάκι). Το φιδάκι ήταν προεγκατεστημένο στις περισσότερες συσκευές της Nokia, έχει γίνει από τότε ένα από τα πιο παγιμένα βιντεοπαιχνίδια και βρίσκεται σε πάνω από 350 εκατομμύρια συσκευές ανά τον κόσμο. Μια παραλλαγή του Snake χρησιμοποιώντας την θύρα υπέρυθρων, έγινε το πρώτο παιχνίδι 2 παικτών στα κινητά τηλέφωνα.

Σήμερα, τα mobile games συνήθως γίνονται download από καταστήματα εφαρμογών(app stores), αλλά σε μερικές περιπτώσεις έρχονται προεγκατεστημένα στις συσκευές, ή μέσω υπέρυθρων, Bluetooth ή κάρτας μνήμης.

Υπάρχουν τρεις κατηγορίες mobile games σχετικά με την τιμή τους. Πολλά είναι εντελώς δωρεάν, μερικά επί πληρωμή κατά το κατέβασμα, και τέλος πολλά mobile games διατίθενται δωρεάν στο κατέβασμα, αλλά περιέχουν/παρέχουν υπηρεσίες/αντικείμενα επί πληρωμή στα μενού του παιχνιδιού (microtransactions). Μερικά παρέχονται δωρεάν και περιέχουν διαφημίσεις. Και οι δύο παραπάνω περιπτώσεις εντάσσονται στην κατηγορία “Freemium”, δηλαδή δωρεάν εκ πρώτης όψεως, με μερικά μειονεκτήματα σε σχέση με τα εντελώς δωρεάν παιχνίδια. Η συγκεκριμένη κατηγορία είναι αυτή που κυριαρχεί στα downloads, επειδή δεν αναγκάζει τον χρήστη να πληρώσει για το παιχνίδι και παράλληλα προσθέτει βιωσιμότητα στο παιχνίδι λόγω των κερδών από διαφημίσεις και microtransactions.

2.2. TA MOBILE PUZZLE GAMES

Τα βιντεοπαιχνίδια γρίφων(παζλ) είναι ένα ξεχωριστό είδος βιντεοπαιχνιδιών όπου δίνεται έμφαση στην επίλυση γρίφων. Οι τύποι των γρίφων είναι πολλοί και δοκιμάζουν τον χρήστη στην επίλυσή τους χρησιμοποιώντας λογική, αναγνώριση επαναλαμβανόμενων σχεδίων, συμπλήρωση λέξεων καθώς και πολλών άλλων επιδεξιότητων επίλυσης προβλημάτων. Ο χρήστης μπορεί να έχει άπειρες προσπάθειες ή ατελείωτο χρόνο επίλυσης του γρίφου, μπορεί να υπάρχει κάποιο όριο χρόνου ή προσπάθειών, ή η δυσκολία να εμφανίζεται στο ότι ο γρίφος λύνεται σε πραγματικό χρόνο (π.χ. Tetris).

Ένα sliding puzzle, με το οποίο θα ασχοληθούμε στην εργασία, είναι ο τύπος γρίφου/παζλ που θα πρέπει να μπουν τα σωστά κομμάτια στη σωστή θέση, μετακινώντας τα και εναλλάσσοντάς τα(αλλά χωρίς να μπορεί ο χρήστης να σηκώσει τα κομμάτια) με άλλα κομμάτια μέσα στο παζλ, με σκοπό στο τέλος να είναι όλα τα κομμάτια στη σωστή θέση.

2.3. Η ΓΛΩΣΣΑ ΠΡΟΓΡΑΜΜΑΤΙΣΜΟΥ JAVA

Η Java είναι από τις πλέον κυρίαρχες γλώσσες προγραμματισμού στον κόσμο, δεν ήταν όμως πάντοτε γνωστή με αυτό το όνομα. “Πατέρας” της Java θεωρείται ο James Gosling, ο οποίος την εποχή που εργαζόταν στη Sun, το μακρινό 1991, έκανε πειραματισμούς πάνω στη C++, καθώς η εταιρία αναζητούσε το κατάλληλο εργαλείο για να αποτελέσει την πλατφόρμα ανάπτυξης λογισμικού σε μικροσυσκευές. Οι υπάρχουσες γλώσσες δεν ήταν αρκετές, οπότε μετά από πολλούς πειραματισμούς, κατέληξαν στην Oak.

Το συγκεκριμένο όνομα όμως, επειδή ήταν ήδη κατοχυρωμένο, ανάγκασε τους δημιουργούς της να το αλλάξουν σε Java. Η ονομασία εμπνεύστηκε από την αγαπημένη ποικιλία καφέ τους, και επιλέχθηκε μετά από μια από τις πολλές μαζώξεις τους σε κάποιο τοπικό καφέ.

Το 2010, η Sun και οι τεχνολογίες της εξαγοράστηκαν από την Oracle Corporation, έχοντας ως αποτέλεσμα όλα τα πνευματικά δικαιώματα και οι πατέντες της πρώτης να περάσουν στη δεύτερη. Η συμφωνία είναι σημαντική για το μέλλον της Java, καθώς ο έλεγχος περνάει σε άλλα χέρια.

Στα υπέρ της εντάσσεται το ότι, έναντι των περισσότερων άλλων γλωσσών, τα προγράμματα γραμμένα σε Java τρέχουν ακριβώς το ίδιο ανεξαρτήτου λειτουργικού συστήματος και πλατφόρμας, χωρίς να χρειαστεί να ξαναγίνει μεταγλώττιση ή να αλλάξει ο πηγαίος κώδικας για κάθε διαφορετικό λειτουργικό σύστημα. Ένα ακόμα υπέρ της είναι ο Garbage Collector, ο οποίος ελευθερώνει τμήματα μνήμης που δε χρησιμοποιούνται πλέον.

Στα μείον της εντάσσεται το ότι είναι πιο αργή σε σχέση με άλλες γλώσσες υψηλού επιπέδου(π.χ. C++). Ωστόσο, γίνονται συνεχείς προσπάθειες από την Oracle για βελτιστοποίηση της εικονικής μηχανής, ενώ υπάρχουν και υλοποιήσεις τρίτων, που μπορεί να φέρουν καλύτερα ή χειρότερα αποτελέσματα.

Η Java, καθώς είναι αντικειμενοστραφής γλώσσα, βασίζεται στην χρήση αντικειμένων. Τα αντικείμενα είναι συλλογές πεδίων πληροφορίας και μεθόδων επεξεργασίας και προβολής πληροφορίας. Τα αντικείμενα με τη σειρά τους ανήκουν σε κλάσεις, οι οποίες δηλώνουν τον τύπο των αντικειμένων.

2.4. ΤΟ ΛΕΙΤΟΥΡΓΙΚΟ ΣΥΣΤΗΜΑ ANDROID(ANDROID OS)

Το Android είναι ένα λειτουργικό σύστημα που τρέχει σε πυρήνα(kernel) του Linux. Το συγκεκριμένο λειτουργικό σύστημα χρησιμοποιείται σε συσκευές κινητής τηλεφωνίας, tablets, smartwatches, media players, activity trackers, τηλεοράσεις κλπ. Ανήκει στην Google, και αρχικά αναπτύχθηκε από την ίδια· αργότερα από την Open Handset Alliance. Οι κατασκευαστές λογισμικού συνθέτουν κώδικα με τη χρήση της γλώσσας προγραμματισμού Java, με βιβλιοθήκες λογισμικού ανεπτυγμένες από την Google. Το Android, παρότι είναι σχεδιασμένο κυρίως για συσκευές με οθόνη αφής, έχει χρησιμοποιηθεί και σε άλλες ηλεκτρονικές συσκευές, όπως κονσόλες παιχνιδιών, ηλεκτρονικούς υπολογιστές κ.α.

Το Android είναι το πιο ευρέως διαδεδομένο λογισμικό στον κόσμο, καθώς οι συσκευές με Android έχουν περισσότερες πωλήσεις από όλες τις συσκευές Windows, iOS και Mac OS μαζί.

Το Android παρουσιάστηκε για πρώτη φορά στις 5 Νοεμβρίου του 2007. Η Google δημοσίευσε το μεγαλύτερο μέρος του κώδικα υπό τους όρους της Apache License, μιας ελεύθερης άδειας λογισμικού. Το λογότυπο του είναι ένα πράσινο ανδροϊδές ρομπότ.

Το Android λαμβάνει συχνές ενημερώσεις, με νέες εκδόσεις να παραδίδονται συνήθως ετήσια. Η τελευταία έκδοση την στιγμή εγγραφής της συγκεκριμένης πτυχιακής εργασίας είναι η 8.1 με κωδική ονομασία Oreo.

2.5. TO ANDROID STUDIO

2.5.1. ΧΡΗΣΗ ΟΛΟΚΛΗΡΩΜΕΝΩΝ ΠΕΡΙΒΑΛΛΟΝΤΩΝ ΑΝΑΠΤΥΞΗΣ

Για να μπορέσουμε να αναπτύξουμε αποτελεσματικά μια εφαρμογή για το Android λογισμικό μας, κρίνεται επιθυμητή η χρήση ενός ολοκληρωμένου περιβάλλοντος ανάπτυξης (Integrated Development Environment, IDE), αν όχι απαραίτητη. Ένα ολοκληρωμένο περιβάλλον ανάπτυξης είναι ένα σύνολο εργαλείων εφαρμοσμένων σε ένα λογισμικό, το οποίο μας επιτρέπει την καλύτερη δυνατή εμπειρία καθώς και έλεγχο κατά την υλοποίηση της εφαρμογής. Για την συγκεκριμένη πτυχιακή, θα χρησιμοποιήσουμε το Android Studio, μια σουίτα αναπτυγμένη από την Google.

2.5.2. ΤΙ ΕΙΝΑΙ ΤΟ ANDROID STUDIO

Όπως αναφέραμε και προηγουμένως, το Android Studio αποτελεί την επίσημη πλατφόρμα υλοποίησης εφαρμογών η οποία προσφέρεται από την Google. Έχει υλοποιηθεί πάνω στο λογισμικό JetBrains' IntelliJ IDEA και έχει σχεδιαστεί συγκεκριμένα για την ανάπτυξη εφαρμογών για το Android. Το Android Studio ανακοινώθηκε για πρώτη φορά τον Μάιο του 2013 στη συνεδρία της Google με όνομα Google I/O, και στόχος του ήταν να αντικαταστήσει την πλατφόρμα Eclipse, η οποία μέχρι τότε χρησιμοποιούταν για την ανάπτυξη λογισμικού Android με τη χρήση των Εργαλείων Ανάπτυξης Android (Eclipse Android Development Tools).

Παράλληλα, επειδή υπήρχε ο παραπάνω στόχος, έγινε η προσπάθεια να αναπτυχθεί για όλα τα λειτουργικά συστήματα (Windows, MacOS, Linux) ώστε να υπάρχει η δυνατότητα ανάπτυξης λογισμικού ανεξαρτήτως λειτουργικού συστήματος, κάτι που επετεύχθη, μιας και το Android Studio είναι διαθέσιμο προς λήψη και εγκατάσταση για όλα τα παραπάνω λειτουργικά.

2.5.3. ΠΛΕΟΝΕΚΤΗΜΑΤΑ ΧΡΗΣΗΣ

Το Android Studio παρέχει κάποια προτερήματα σε σχέση με κάποιο ολοκληρωμένο περιβάλλον ανάπτυξη κάποιου τρίτου κατασκευαστή. Η χρήση του προσφέρει εύκολη και αποτελεσματική διόρθωση σφαλμάτων καθώς και υλοποίηση αναπροσαρμοσμένου κώδικα στοχευμένο συγκεκριμένα στο Android, ενώ τα εργαλεία Lint τα οποία διαθέτει μας επιτρέπουν να περιορίσουμε τυχόν προβλήματα, καθώς και να επιλύσουμε θέματα επιδόσεων, χρήσης, συμβατότητας του λογισμικού που αναπτύσσουμε. Επίσης, η πλατφόρμα μας δίνει την δυνατότητα να υπογράψουμε τις εφαρμογές μας, ώστε να διατηρήσουμε την μοναδικότητα της. Ακόμη, το Android Studio μας προσφέρει έναν αρκετά αποτελεσματικό Layout Editor ώστε να δημιουργήσουμε μια κατάλληλη πρόσοψη για την εφαρμογή μας χωρίς την χρήση κώδικα, ενώ παράλληλα περιέχει και ικανοποιητικά Templates για γρήγορη δημιουργία κοινών σχεδίων (designs) για μία εφαρμογή. Παράλληλα, υποστηρίζει την δημιουργία κώδικα για συσκευές Android Wear καθώς επίσης μας δίνει και την δυνατότητα να χρησιμοποιήσουμε την υποστήριξη της Google Cloud πλατφόρμας, ώστε να υλοποιήσουμε την Google App Engine, μια πλατφόρμα η οποία είναι σχεδιασμένη για την φιλοξενία εφαρμογών web. Τέλος, ως τα πιο σημαντικά πλεονεκτήματα μπορούμε να θεωρήσουμε το Gradle-Based build Support το οποίο είναι ένα σύστημα για κατασκευή των εφαρμογών (build system) τύπου JVM (Java Virtual Machine) το οποίο μας επιτρέπει την χρήση δικών μας script γλώσσας Java, καθώς και την άμεση χρήση του Android Virtual Device, ενός εξομοιωτή ο οποίος μας επιτρέπει την γρήγορη εκτέλεση και δοκιμή των εφαρμογών μας, ώστε να μπορέσουμε να κάνουμε την καλύτερη δυνατή αποσφαλμάτωση(debug) τους χρησιμοποιώντας την εισαγωγή δεδομένων μέσω αυτού.

3. ΣΧΕΔΙΑΣΗ ΚΑΙ ΥΛΟΠΟΙΗΣΗ ΤΗΣ ΕΦΑΡΜΟΓΗΣ

3.1. ΓΕΝΙΚΑ: ΤΙ ΘΑ ΧΡΕΙΑΣΤΟΥΜΕ ΓΙΑ ΤΗΝ ΥΛΟΠΟΙΗΣΗ

Για τη δημιουργία ενός παιχνιδιού γρίφων σε περιβάλλον Android θα χρειαστούμε:

- Λογισμικό Android Studio
- Εικόνα που θα χρησιμοποιηθεί για το παζλ
- Εικόνα icon εφαρμογής
- Crop tool για ισομερές κόψιμο εικόνας
- Adobe Photoshop CS6 για τη δημιουργία του icon
- Γνώσεις Java

3.2. ΠΕΡΙΓΡΑΦΗ ΤΗΣ ΥΛΟΠΟΙΗΣΗΣ ΒΗΜΑ ΠΡΟΣ ΒΗΜΑ

3.2.1. ΠΡΟΕΤΟΙΜΑΖΟΝΤΑΣ ΤΙΣ ΕΙΚΟΝΕΣ

Αρχικά, θα χρειαστούμε μια εικόνα για να αποτελέσει τη βάση του ruzzle μας. Η εικόνα που χρησιμοποιήθηκε δεν υπόκειται σε πνευματικά δικαιώματα, καθώς έχει φωτογραφηθεί από τον δημιουργό της εφαρμογής.

Η εικόνα που χρησιμοποιήθηκε είναι η παρακάτω:



Εικόνα που χρησιμοποιήθηκε για το ruzzle

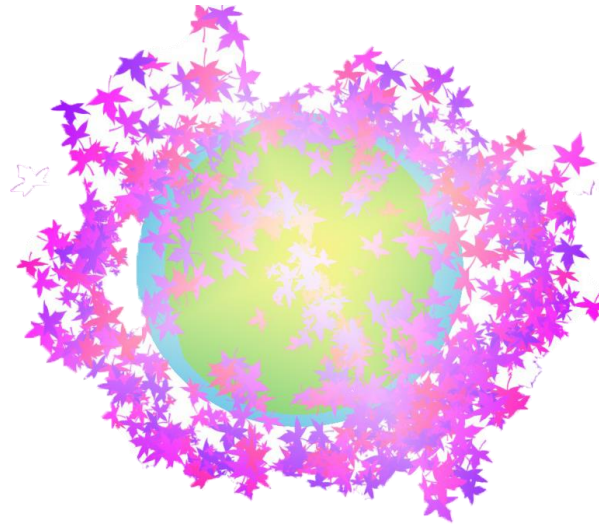
Στη συνέχεια, μέσω του διαδικτυακού εργαλείου [IMGonline.com.ua](https://www.imgonline.com.ua) (<https://www.imgonline.com.ua/eng/cut-photo-into-pieces.php>) διαιρέθηκε η εικόνα σε 9 (3*3) ίσα κομμάτια, τα οποία φαίνονται παρακάτω:



Η εικόνα μετά τον τεμαχισμό

Μετάπειτα χρησιμοποιήσαμε την εφαρμογή Adobe Photoshop CS6 για να δημιουργήσουμε το logo της εφαρμογής, καθώς και το background του αρχικού της μενού, τα οποία παρατίθενται στην επόμενη σελίδα.

Εικονίδιο Εφαρμογής:



Εικονίδιο εφαρμογής

Background Αρχικής οθόνης εφαρμογής:

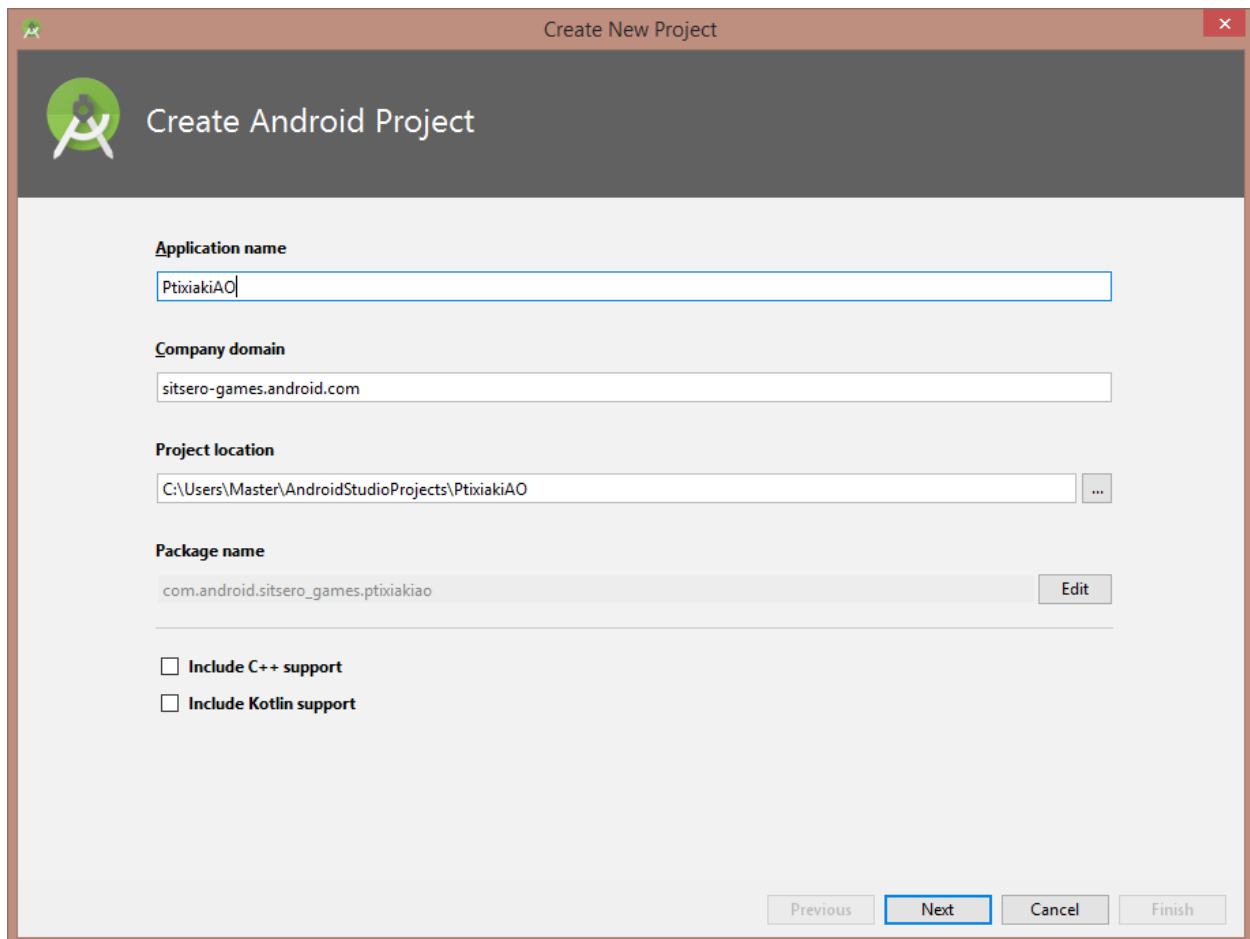


ABOUT

Background αρχικής οθόνης

3.2.2. ΔΗΜΙΟΥΡΓΩΝΤΑΣ ΤΟ PROJECT

Ξεκινώντας, ανοίγουμε το Android Studio, και δημιουργούμε νέο project. Βάζουμε όποιο όνομα και όνομα εταιρίας θέλουμε, καθώς και ορίζουμε την τοποθεσία στην οποία θα αποθηκευτεί το project. Πατάμε Next.



The screenshot shows the 'Create New Project' dialog in Android Studio. The dialog is titled 'Create New Project' and contains the following fields:

- Application name:** PtixiakIAO
- Company domain:** sitsero-games.android.com
- Project location:** C:\Users\Master\AndroidStudioProjects\PtixiakIAO
- Package name:** com.android.sitsero_games.ptixiakiao

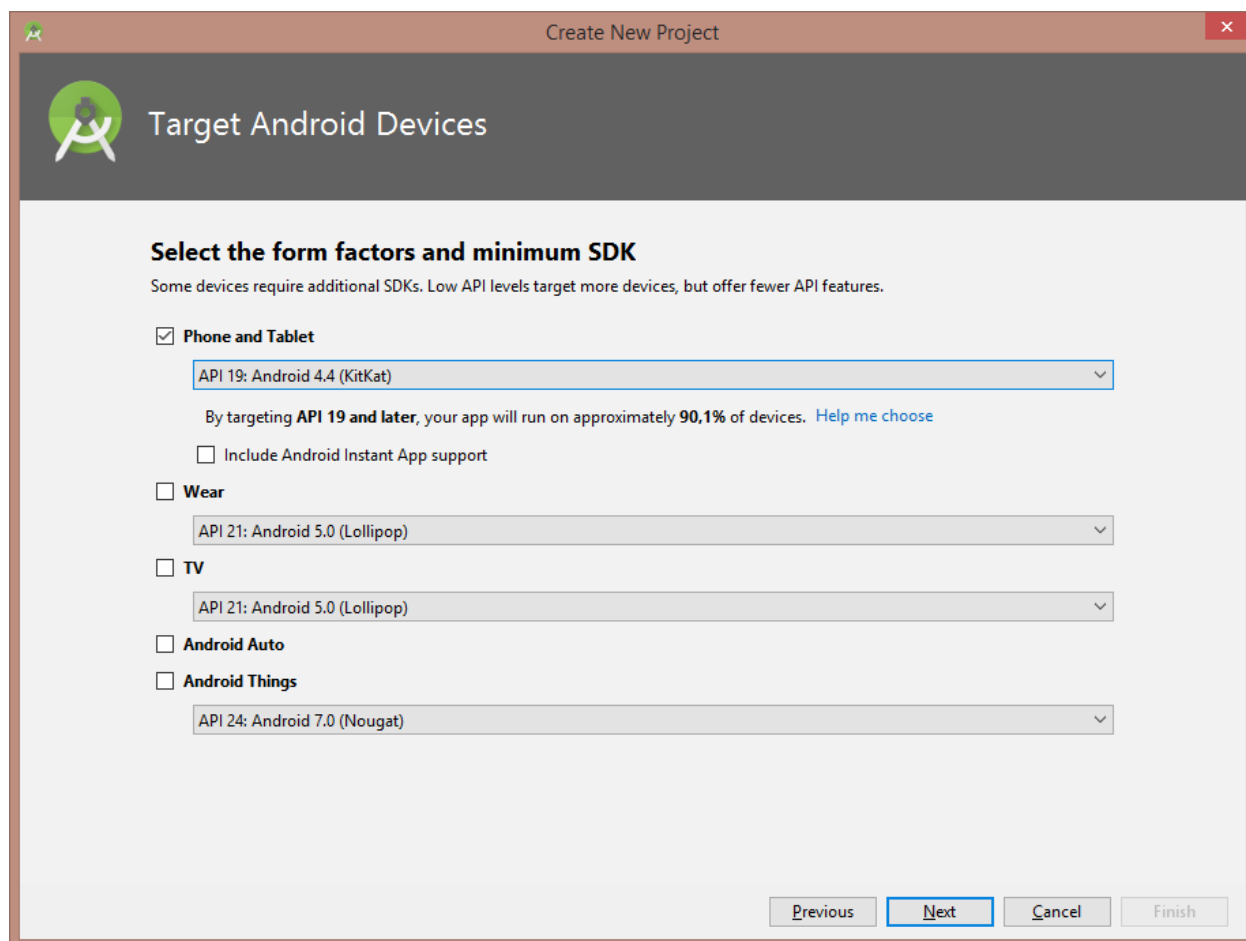
There are also two checkboxes:

- Include C++ support
- Include Kotlin support

At the bottom, there are four buttons: Previous, Next, Cancel, and Finish. The 'Next' button is highlighted with a blue border.

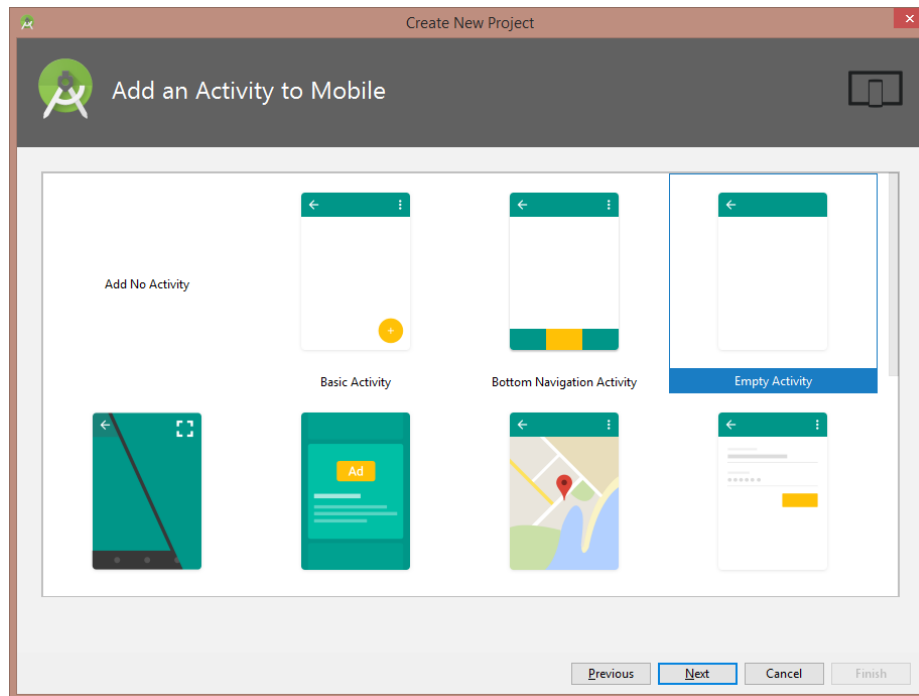
Δημιουργία project

Στην επόμενη οθόνη επιλέγουμε για ποια λειτουργικά θα είναι compatible η εφαρμογή μας. Στο συγκεκριμένο project επέλεξα API19(Android KitKat και πάνω). Μόλις είμαστε χαρούμενοι με τις επιλογές μας πατάμε Next.

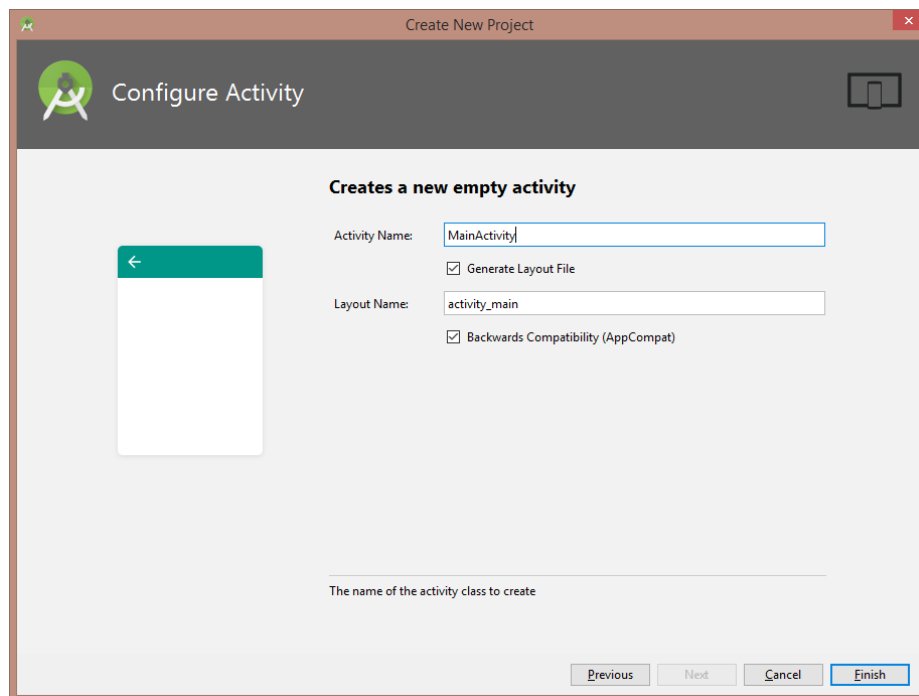


Επιλογή API

Επιλέγουμε να ξεκινήσουμε με Empty Activity, προχωράμε, και εάν θέλουμε αλλάζουμε το όνομα της κύριας Activity (στην περίπτωση μας το αφήσαμε στο default MainActivity). Πατώντας Finish αρχίζουμε το χτίσιμο του project μας.



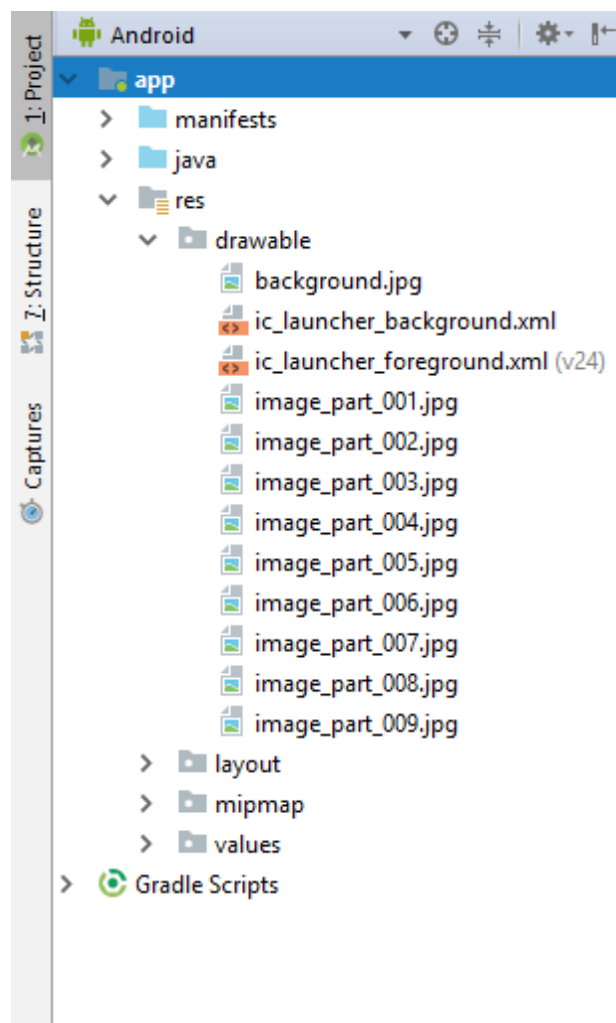
Επιλογή Empty Activity



Ονομασία νέας Activity

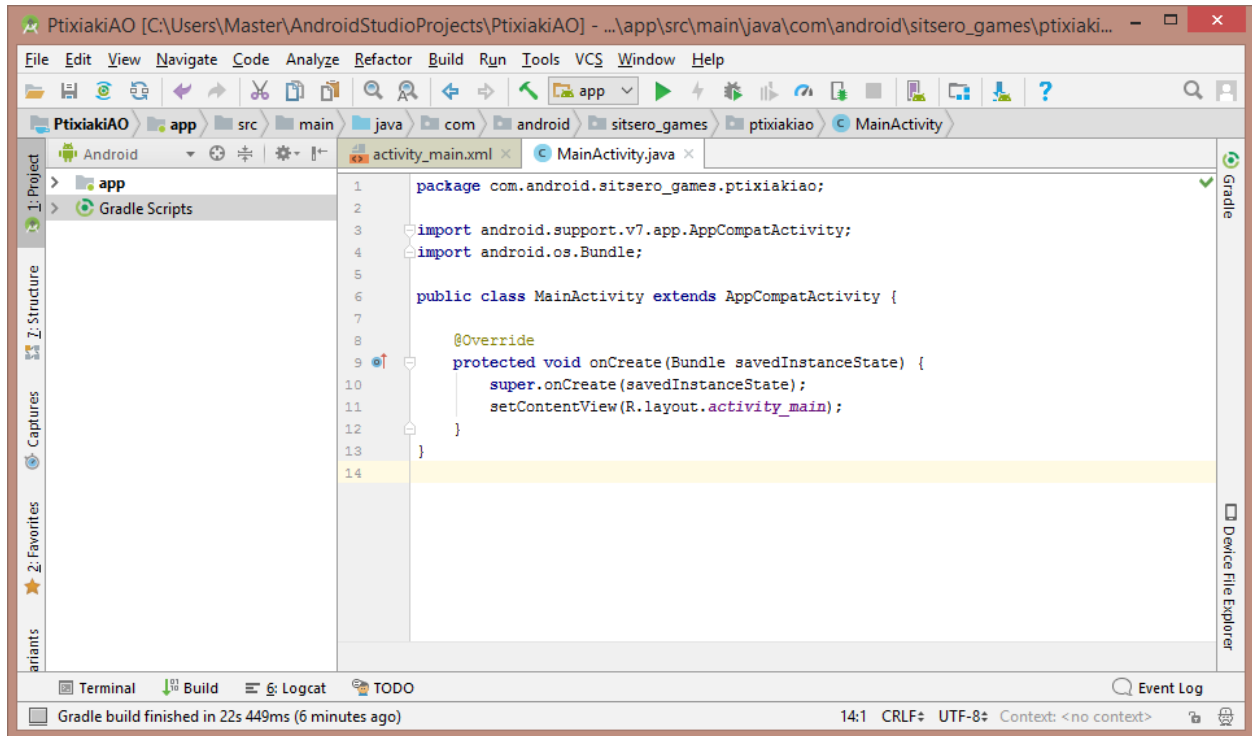
3.2.3. ΟΡΙΣΜΟΣ ΑΡΧΙΚΩΝ ΤΙΜΩΝ

Πριν προχωρήσουμε με οτιδήποτε άλλο, ξεκινάμε τοποθετώντας τις εικόνες που θα χρειαστούμε, δηλαδή τα 9 κομμάτια του παζλ και την εικόνα του background της Αρχικής οθόνης στον φάκελο drawable, τα οποία θα χρησιμοποιηθούν αργότερα. Η εικόνα του logo της εφαρμογής θα μπει σε μεταγενέστερο στάδιο με άλλη διαδικασία. Μετά την αντιγραφή των εικόνων, πρέπει να έχουμε αυτό το αποτέλεσμα κάτω από τον drawable.



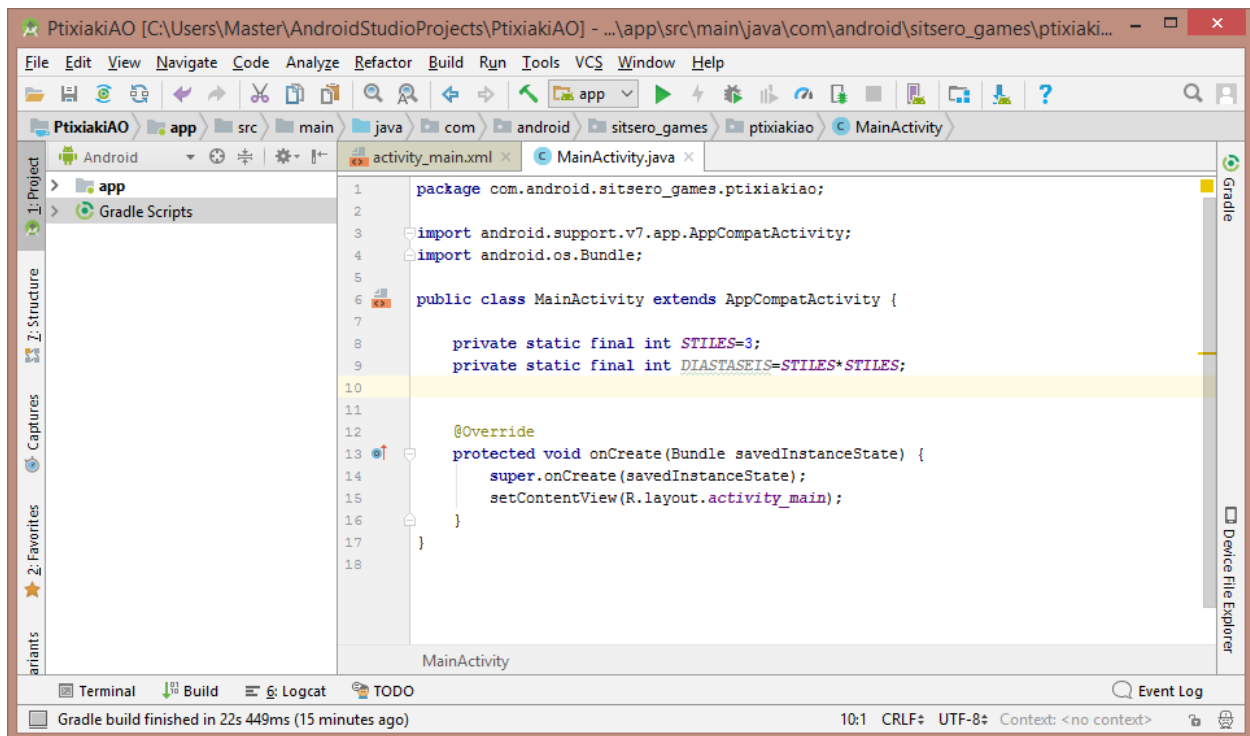
Drawable μετά την αντιγραφή των εικόνων

Αφού πατήσουμε Finish, το Android Studio μας αφήνει στην οθόνη της MainActivity κλάσης.



Αρχική οθόνη της MainActivity.java

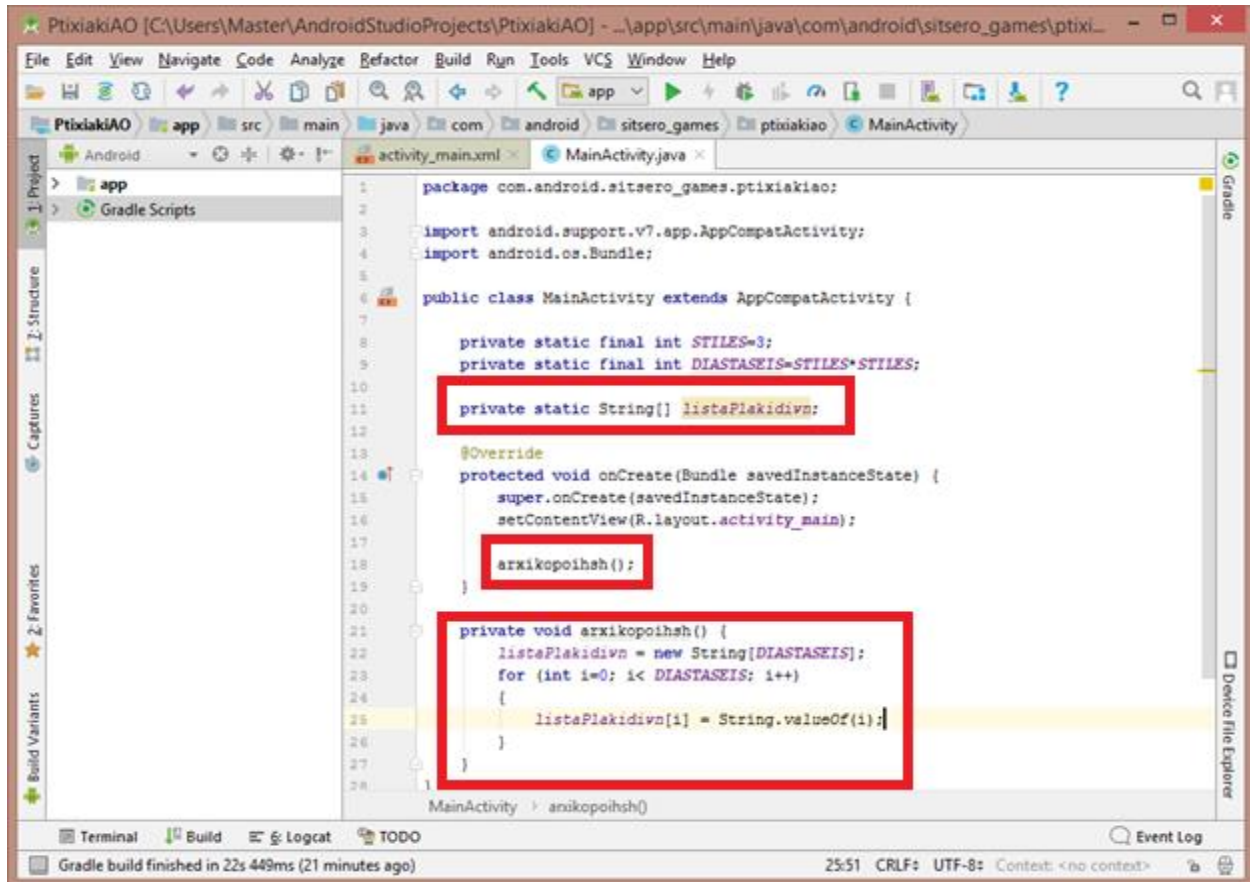
Αρχικά, θέλουμε να ορίσουμε τις διαστάσεις του ruzzle, οπότε αρχικοποιούμε τις στήλες, αλλά και τις ίδιες τις διαστάσεις του.



```
1 package com.android.sitsero_games.ptixiakiao;
2
3 import android.support.v7.app.AppCompatActivity;
4 import android.os.Bundle;
5
6 public class MainActivity extends AppCompatActivity {
7
8     private static final int STILES=3;
9     private static final int DIASTASEIS=STILES*STILES;
10
11
12     @Override
13     protected void onCreate(Bundle savedInstanceState) {
14         super.onCreate(savedInstanceState);
15         setContentView(R.layout.activity_main);
16     }
17 }
18
```

Δήλωση στηλών & διαστάσεων

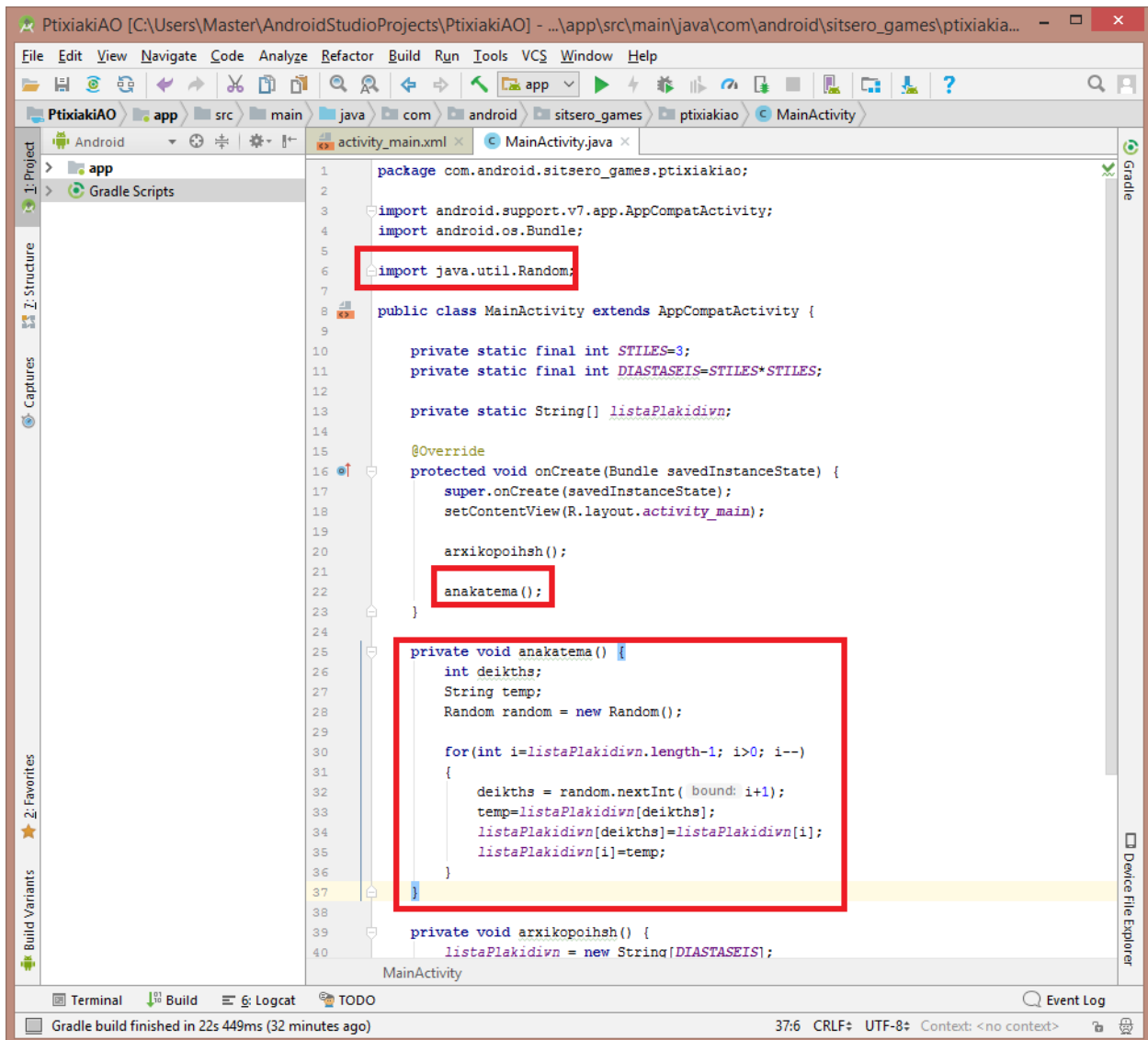
Εν συνεχεία, ορίζουμε μια λίστα πλακιδίων και την αρχικοποιούμε. Προσθέτουμε κλήση της αρχικοποίησης στην onCreate.



```
1 package com.android.sitsero_games.ptixiakiao;
2
3 import android.support.v7.app.AppCompatActivity;
4 import android.os.Bundle;
5
6 public class MainActivity extends AppCompatActivity {
7
8     private static final int STILES=3;
9     private static final int DIASTASEIS=STILES*STILES;
10
11     private static String[] listaPlekidiwn;
12
13     @Override
14     protected void onCreate(Bundle savedInstanceState) {
15         super.onCreate(savedInstanceState);
16         setContentView(R.layout.activity_main);
17
18         arxikopoihsh();
19     }
20
21     private void arxikopoihsh() {
22         listaPlekidiwn = new String[DIASTASEIS];
23         for (int i=0; i< DIASTASEIS; i++)
24         {
25             listaPlekidiwn[i] = String.valueOf(i);
26         }
27     }
28 }
```

Αρχικοποίηση λίστας πλακιδίων

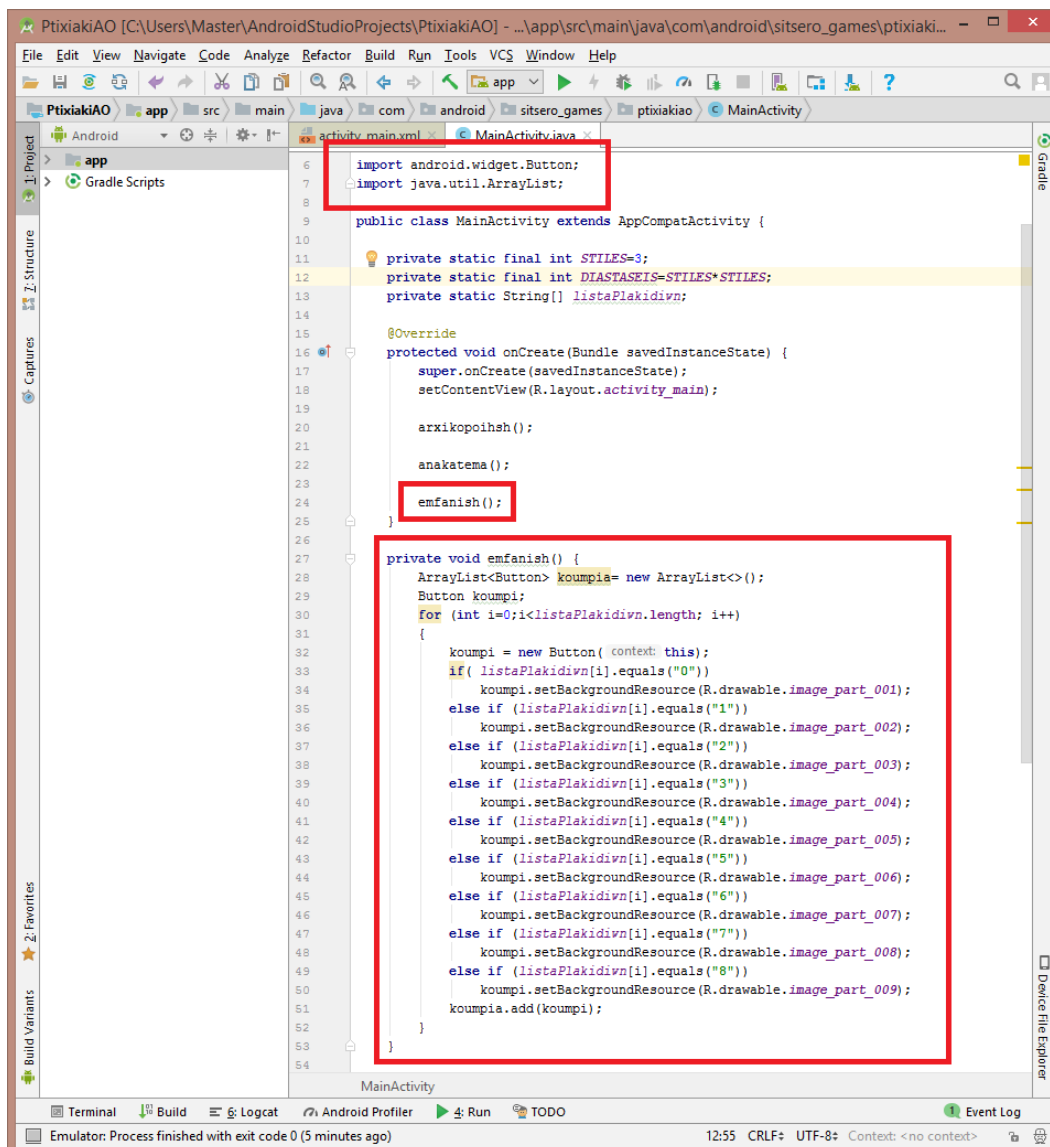
Η συγκεκριμένη μέθοδος όμως αρχικοποιεί στη σειρά, κι εμείς θέλουμε τα πλακίδια να είναι ανακατεμένα μόλις αρχίζει το παιχνίδι. Οπότε δημιουργούμε μια μέθοδο ανακατέματος, και την ορίζουμε. Προσθέτουμε κλήση του ανακατέματος στην onCreate. Μέσα στο ανακάτεμα καλούμε την κλάση Random και δημιουργούμε μια τυχαία μεταβλητή τύπου Random. Αυτομάτως προσθέτουμε και την Random στα imports της MainActivity. Μετά ανακατεύουμε την λίστα πλακιδίων χρησιμοποιώντας την μεταβλητή random που δημιουργήσαμε, δίνοντας την τιμή στον δείκτη, με τη χρήση του random.nextInt(i+1) το οποίο παίρνει έναν αριθμό ανάμεσα από το 0 και το (i+1), στην προκειμένη περίπτωση ανάμεσα στο 0 και το μέγεθος/μήκος της λίστας πλακιδίων. Στη συνέχεια ανακατεύουμε τις τιμές με μια τοπική swap.



```
1 package com.android.sitsero_games.ptixiakiao;
2
3 import android.support.v7.app.AppCompatActivity;
4 import android.os.Bundle;
5
6 import java.util.Random;
7
8 public class MainActivity extends AppCompatActivity {
9
10     private static final int STILES=3;
11     private static final int DIASTASEIS=STILES*STILES;
12
13     private static String[] listaPlakidiwn;
14
15     @Override
16     protected void onCreate(Bundle savedInstanceState) {
17         super.onCreate(savedInstanceState);
18         setContentView(R.layout.activity_main);
19
20         arxikopoihsh();
21         anakatema();
22     }
23
24     private void anakatema() {
25         int deikths;
26         String temp;
27         Random random = new Random();
28
29         for(int i=listaPlakidiwn.length-1; i>0; i--){
30             {
31                 deikths = random.nextInt( bound: i+1);
32                 temp=listaPlakidiwn[deikths];
33                 listaPlakidiwn[deikths]=listaPlakidiwn[i];
34                 listaPlakidiwn[i]=temp;
35             }
36         }
37     }
38
39     private void arxikopoihsh() {
40         listaPlakidiwn = new String[DIASTASEIS];
```

Δημιουργία ανακατέματος

Στη συνέχεια δημιουργούμε τη μέθοδο Εμφάνιση. Προσθέτουμε μια λίστα κουμπιών, κάνοντας import αυτομάτως τις ArrayList και Button κλάσεις. Η ArrayList θα πρέπει να αρχικοποιηθεί μέσα στην Εμφάνιση, διότι θα αλλάζει δυναμικά κάθε φορά που γίνεται μια κίνηση στην εφαρμογή. Ονομάζουμε την λίστα μας «κουμπιά» και το κουμπί μας «κουμπί». Εν συνεχεία συνδέουμε τα πλακίδια με τις εικόνες, μετατρέποντάς τα σε κουμπιά, ορίζοντάς τους τις εικόνες ως background images και προσθέτοντας τα κουμπιά στη λίστα. (Υποσημείωση: στην θέση του κουμπι=new Button(this); θα προστεθεί αργότερα μεταβλητή τύπου Context αντί του “this”).

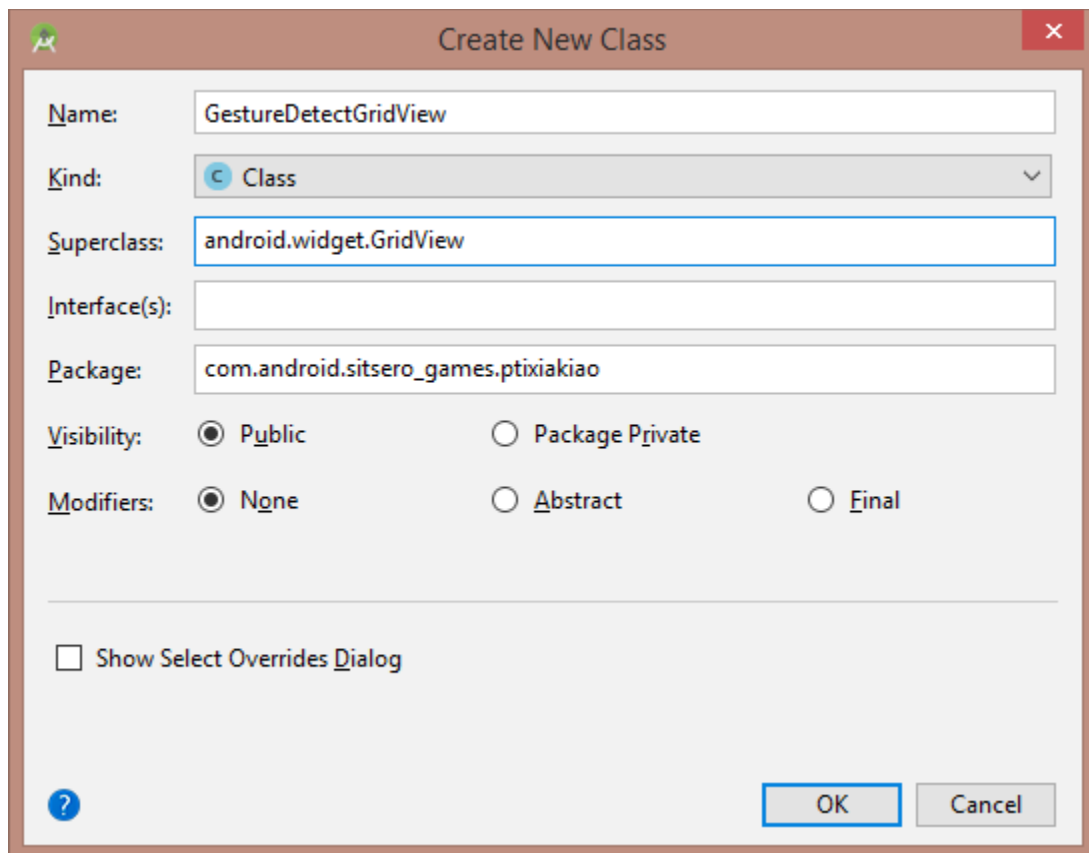


```
6 import android.widget.Button;
7 import java.util.ArrayList;
8
9 public class MainActivity extends AppCompatActivity {
10
11     private static final int STILES=3;
12     private static final int DIASTASEIS=STILES*STILES;
13     private static String[] listaPlakidiwn;
14
15     @Override
16     protected void onCreate(Bundle savedInstanceState) {
17         super.onCreate(savedInstanceState);
18         setContentView(R.layout.activity_main);
19
20         arxikopoihsh();
21
22         anakatema();
23
24         emfanish();
25     }
26
27     private void emfanish() {
28         ArrayList<Button> koumpia= new ArrayList<>();
29         Button koumpi;
30         for (int i=0;i<listaPlakidiwn.length; i++)
31         {
32             koumpi = new Button( context: this);
33             if ( listaPlakidiwn[i].equals("0"))
34                 koumpi.setBackgroundResource(R.drawable.image_part_001);
35             else if ( listaPlakidiwn[i].equals("1"))
36                 koumpi.setBackgroundResource(R.drawable.image_part_002);
37             else if ( listaPlakidiwn[i].equals("2"))
38                 koumpi.setBackgroundResource(R.drawable.image_part_003);
39             else if ( listaPlakidiwn[i].equals("3"))
40                 koumpi.setBackgroundResource(R.drawable.image_part_004);
41             else if ( listaPlakidiwn[i].equals("4"))
42                 koumpi.setBackgroundResource(R.drawable.image_part_005);
43             else if ( listaPlakidiwn[i].equals("5"))
44                 koumpi.setBackgroundResource(R.drawable.image_part_006);
45             else if ( listaPlakidiwn[i].equals("6"))
46                 koumpi.setBackgroundResource(R.drawable.image_part_007);
47             else if ( listaPlakidiwn[i].equals("7"))
48                 koumpi.setBackgroundResource(R.drawable.image_part_008);
49             else if ( listaPlakidiwn[i].equals("8"))
50                 koumpi.setBackgroundResource(R.drawable.image_part_009);
51             koumpia.add(koumpi);
52         }
53     }
54 }
```

Αρχική κλήση και ορισμός εμφάνισης

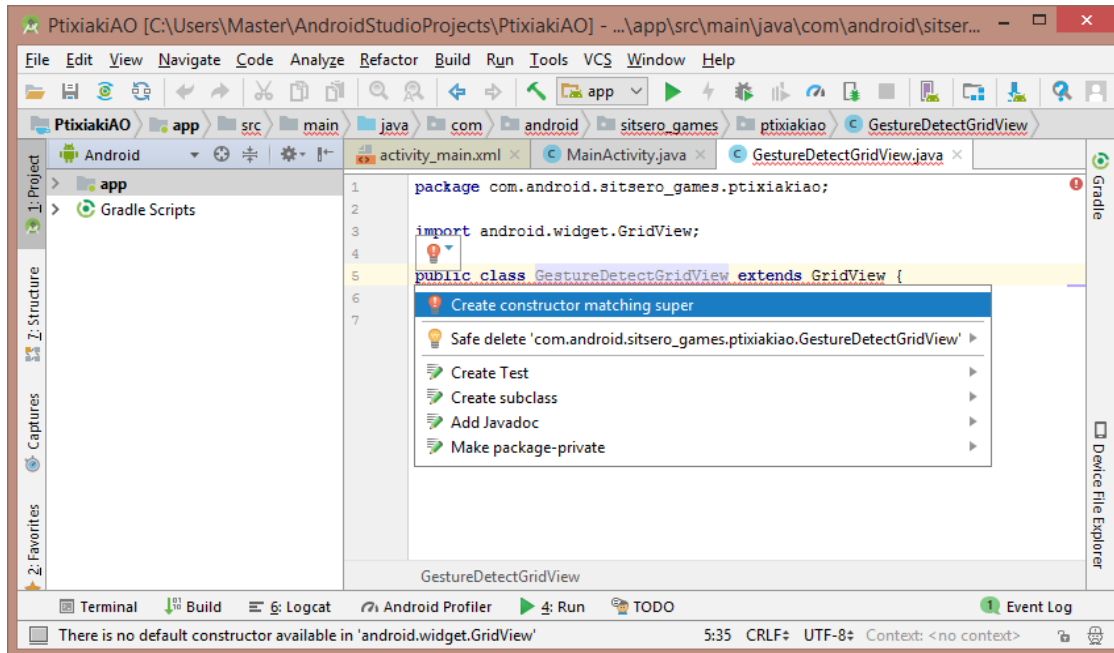
3.2.4. ΔΗΜΙΟΥΡΓΙΑ GestureDetectGridView

Οπότε πατάμε δεξί κλικ στο όνομα εταιρίας μας->New->Java Class, και ονομάζουμε τη νέα κλάση GestureDetectGridView, η οποία θα κάνει extend στη GridView.(Υποσημείωση: Μέσω της GridView χωρίζουμε την οθόνη σε Grids). Πατάμε OK.

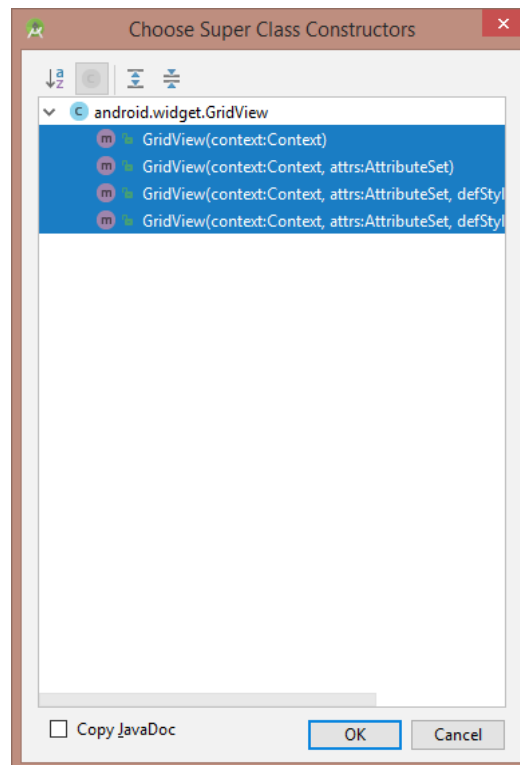


Δημιουργία κλάσης GestureDetectGridView

Εν συνεχεία θα δούμε ότι η κλάση θα χρειαστεί τους constructors της GridView, οπότε τους προσθέτουμε όλους.

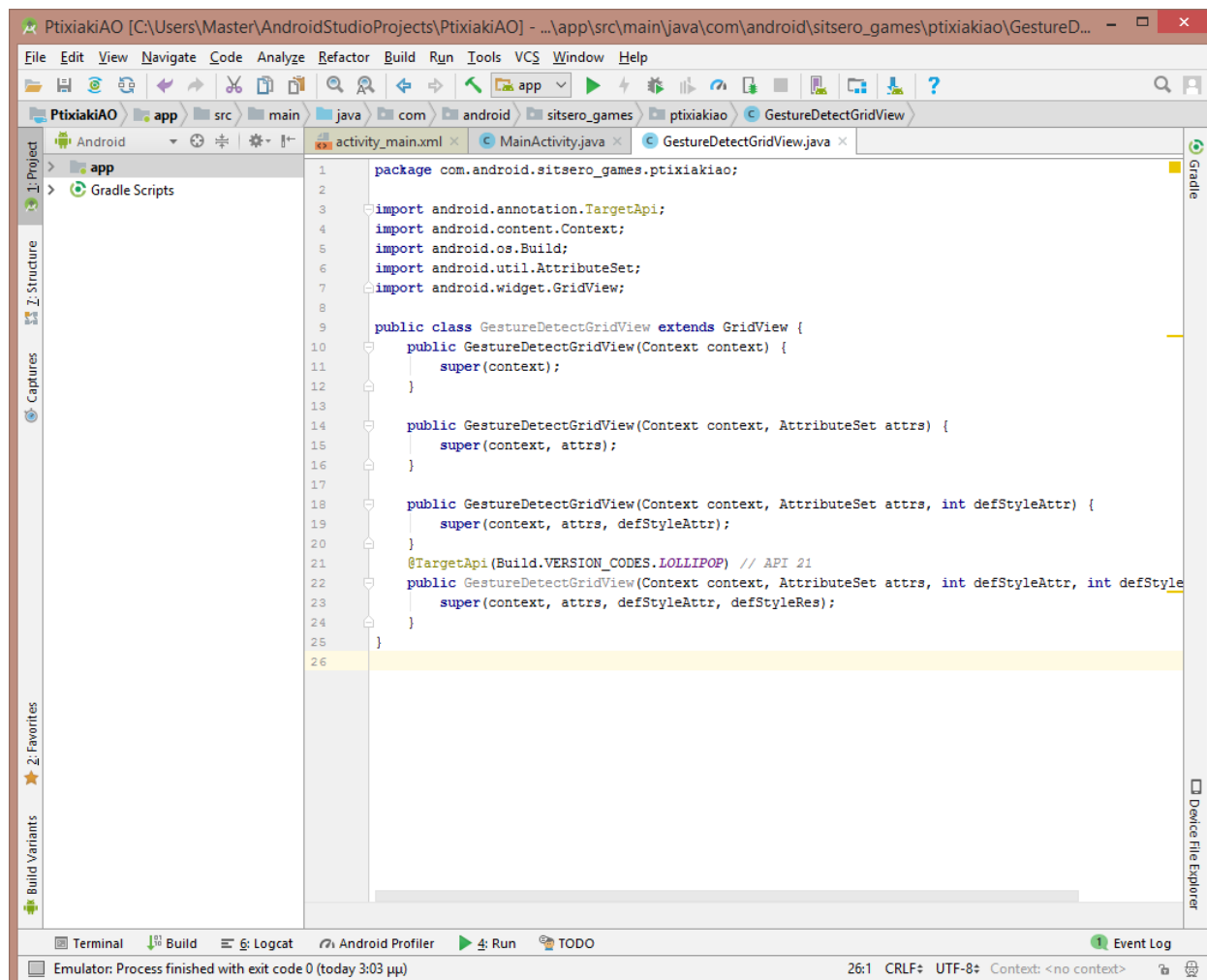


Δημιουργία constructors GridView 1/2



Δημιουργία constructors GridView 2/2

Προσθέτουμε το “@TargetApi(Build.VERSION_CODES.LOLLIPOP)” στον τελευταίο constructor για να εντάξουμε και τα API21, διότι αλλιώς έβγαζε σφάλμα.



```
1 package com.android.sitsero_games.ptixiakiao;
2
3 import android.annotation.TargetApi;
4 import android.content.Context;
5 import android.os.Build;
6 import android.util.AttributeSet;
7 import android.widget.GridView;
8
9 public class GestureDetectGridView extends GridView {
10     public GestureDetectGridView(Context context) {
11         super(context);
12     }
13
14     public GestureDetectGridView(Context context, AttributeSet attrs) {
15         super(context, attrs);
16     }
17
18     public GestureDetectGridView(Context context, AttributeSet attrs, int defStyleAttr) {
19         super(context, attrs, defStyleAttr);
20     }
21     @TargetApi(Build.VERSION_CODES.LOLLIPOP) // API 21
22     public GestureDetectGridView(Context context, AttributeSet attrs, int defStyleAttr, int defStyleRes) {
23         super(context, attrs, defStyleAttr, defStyleRes);
24     }
25 }
26
```

Εισαγωγή API σε τελευταίο constructor

Στη συνέχεια αρχικοποιούμε έναν `GestureDetector`^[1] ο οποίος θα μας χρειαστεί για τις κινήσεις/χειρονομίες που θα κάνουμε στην οθόνη^[2]. Η αρχικοποίηση θα λαμβάνει τιμή από την μεταβλητή `context` τύπου `Context`, η οποία χρησιμοποιείται για κουμπιά. Μέσα σε αυτή δημιουργούμε και custom `Listener` για τις χειρονομίες. Μέσα στον `Listener` αρχικά ορίζουμε ως default να αποδεχόμαστε όλες τις χειρονομίες. Στη συνέχεια δημιουργούμε μια `Boolean onFling` για τους περιορισμούς που θέλουμε να θέσουμε, καθώς και τις ενέργειες που θα επιστρέφουμε. Δηλώνουμε στο `GestureDetectGridView` τις αποστάσεις (σε dpi) που θέλουμε να ορίσουμε ως επαρκείς, δηλαδή την μικρότερη απόσταση χειρονομίας, την μέγιστη απόσταση εκτός πορείας χειρονομίας, καθώς ένα όριο επιτάχυνσης χειρονομίας. Στη συγκεκριμένη εργασία χρησιμοποιήσαμε όριο 100dpi και για τις 3 αυτές σταθερές, καθώς είναι το standard στις λίγο παλαιότερες συσκευές. Αν θέλουμε να αυξήσουμε τις σταθερές αυτές, θα είναι μόνο για συσκευές με πολύ περισσότερα pixels. Παράλληλα κάνουμε `import` και τα κατάλληλα classes. Μέσα στον `Listener` θα χρειαστούμε μια `boolean onFling`, η οποία ορίζεται ως εξής:

```
onFling (MotionEvent e1, MotionEvent e2, float velocityX, float velocityY)  
Notified of a fling event when it occurs with the initial on down MotionEvent and the matching up MotionEvent.[3]
```

Οπότε μέσα σε αυτή συγκρίνουμε τις 2 κινήσεις, τις οποίες περνάμε και από τους τρεις παραπάνω ελέγχους των σταθερών που ορίσαμε, επιστρέφοντας την `onFling` με τις μεταβλητές που πέρασαν τον έλεγχο, στον `Listener`. Μέσα στην `onFling` έχουμε ορίσει μια μέθοδο την οποία θα δημιουργήσουμε και θα δούμε αργότερα στην `MainActivity`, η οποία για τώρα παραμένει ως σχόλιο.

```
1 package com.android.sitsero_games.ptixiakiao;
2
3 import android.annotation.TargetApi;
4 import android.content.Context;
5 import android.os.Build;
6 import android.util.AttributeSet;
7 import android.view.GestureDetector;
8 import android.view.MotionEvent;
9 import android.widget.GridView;
10
11 public class GestureDetectGridView extends GridView {
12
13     private GestureDetector elegkths;
14
15     private static final int SWIPE_MIN_DISTANCE = 100;
16     private static final int SWIPE_MAX_OFF_PATH = 100;
17     private static final int SWIPE_THRESHOLD_VELOCITY = 100;
18
19     public GestureDetectGridView(Context context) {
20         super(context);
21     }
22
23     public GestureDetectGridView(Context context, AttributeSet attrs) {
24         super(context, attrs);
25     }
26
27     public GestureDetectGridView(Context context, AttributeSet attrs, int defStyleAttr) {
28         super(context, attrs, defStyleAttr);
29     }
30     @TargetApi(Build.VERSION_CODES.LOLLIPOP) // API 21
31     public GestureDetectGridView(Context context, AttributeSet attrs, int defStyleAttr, int defStyleRes) {
32         super(context, attrs, defStyleAttr, defStyleRes);
33     }
34 }
```

Δήλωση ελεγκτή και σταθερών

Μετάπειτα προσθέτουμε 2 νέες boolean μεθόδους[4]. Θα χρειαστούμε μια επιπλέον boolean μεταβλητή, και δύο float μεταβλητές. Ο στόχος είναι να κάνουμε override τα κουμπιά. Χωρίς αυτές τις 2 επιπλέον μεθόδους, εάν μετακινούσαμε τα κουμπιά, από πίσω τα πλακίδια θα μετακινούνταν, αλλά δε θα υπήρχε εμφανής αλλαγή στον χρήστη, θα έβλεπε τις εικόνες στην αρχική τους ανακατεμένη θέση. Ενώ, εφόσον η νέα boolean μεταβλητή μας πάρει την τιμή **true**, τότε και η πρώτη boolean μέθοδος θα επιστρέψει **true** και θα προχωρήσει στην δεύτερη boolean μέθοδο, η οποία με τη σειρά της θα επιστρέψει **true**. Ορίζουμε σε μια μεταβλητή int το

“[getActionMasked](#) ()

Return the masked action being performed, without pointer index information.”[5]

το οποίο όπως αναφέρεται επιστρέφει την πράξη χωρίς πληροφορίες του δείκτη. Δίνουμε στον ελεγκτή την τιμή του συμβάντος, και πράττουμε αναλόγως. Ο κώδικας που είναι σε μορφή σχολίου θα γίνει implement αργότερα.

```
1 package com.android.sitsero_games.ptixiakiao;
2
3 import android.annotation.TargetApi;
4 import android.content.Context;
5 import android.os.Build;
6 import android.util.AttributeSet;
7 import android.view.GestureDetector;
8 import android.view.MotionEvent;
9 import android.widget.GridView;
10
11 public class GestureDetectGridView extends GridView {
12
13     private GestureDetector elegkths;
14
15     private static final int SWIPE_MIN_DISTANCE = 100;
16     private static final int SWIPE_MAX_OFF_PATH = 100;
17     private static final int SWIPE_THRESHOLD_VELOCITY = 100;
18
19     private boolean flingConfirmed = false;
20
21     private float touchX;
22     private float touchY;
23
24     public GestureDetectGridView(Context context) {
25         super(context);
26     }
27
28     public GestureDetectGridView(Context context, AttributeSet attrs) {
```

Δήλωση boolean και float

```
35
36
37 public GestureDetectGridView(Context context, AttributeSet attrs, int defStyleAttr, int defStyleRes) {
38     super(context, attrs, defStyleAttr, defStyleRes);
39 }
40
41 // Αρχικοποίηση των συμβάντων κίνησης μέσω ενός Gesture Detector
42 private void init(final Context context) {
43     mListener = new GestureDetector(context, new GestureDetector.SimpleOnGestureListener() {
44         @Override
45         // Αποδοχή όλων των συμβάντων κίνησης
46         public boolean onDown(MotionEvent event) { return true; }
47
48         @Override
49         public boolean onFling(MotionEvent e1, MotionEvent e2, float velocityX,
50                               float velocityY) {
51             final int position = GestureDetectGridView.this.pointToPosition(
52                 Math.round(e1.getX()), Math.round(e1.getY()));
53
54             if (Math.abs(e1.getY() - e2.getY()) > SWIPE_MAX_OFF_PATH) {
55                 if (Math.abs(e1.getX() - e2.getX()) > SWIPE_MAX_OFF_PATH
56                     || Math.abs(velocityY) < SWIPE_THRESHOLD_VELOCITY) {
57                     return false;
58                 }
59                 if (e1.getY() - e2.getY() > SWIPE_MIN_DISTANCE) {
60                     //MainActivity.metakinshshPlakidiwn(context, MainActivity.UP, position);
61                 } else if (e2.getY() - e1.getY() > SWIPE_MIN_DISTANCE) {
62                     //MainActivity.metakinshshPlakidiwn(context, MainActivity.DOWN, position);
63                 }
64             } else {
65                 if (Math.abs(velocityX) < SWIPE_THRESHOLD_VELOCITY) {
66                     return false;
67                 }
68                 if (e1.getX() - e2.getX() > SWIPE_MIN_DISTANCE) {
69                     //MainActivity.metakinshshPlakidiwn(context, MainActivity.LEFT, position);
70                 } else if (e2.getX() - e1.getX() > SWIPE_MIN_DISTANCE) {
71                     //MainActivity.metakinshshPlakidiwn(context, MainActivity.RIGHT, position);
72                 }
73             }
74
75             return super.onFling(e1, e2, velocityX, velocityY);
76         }
77     });
78 }
79
```

Αρχικοποίηση συμβάντων κίνησης ενός ελεγκτή

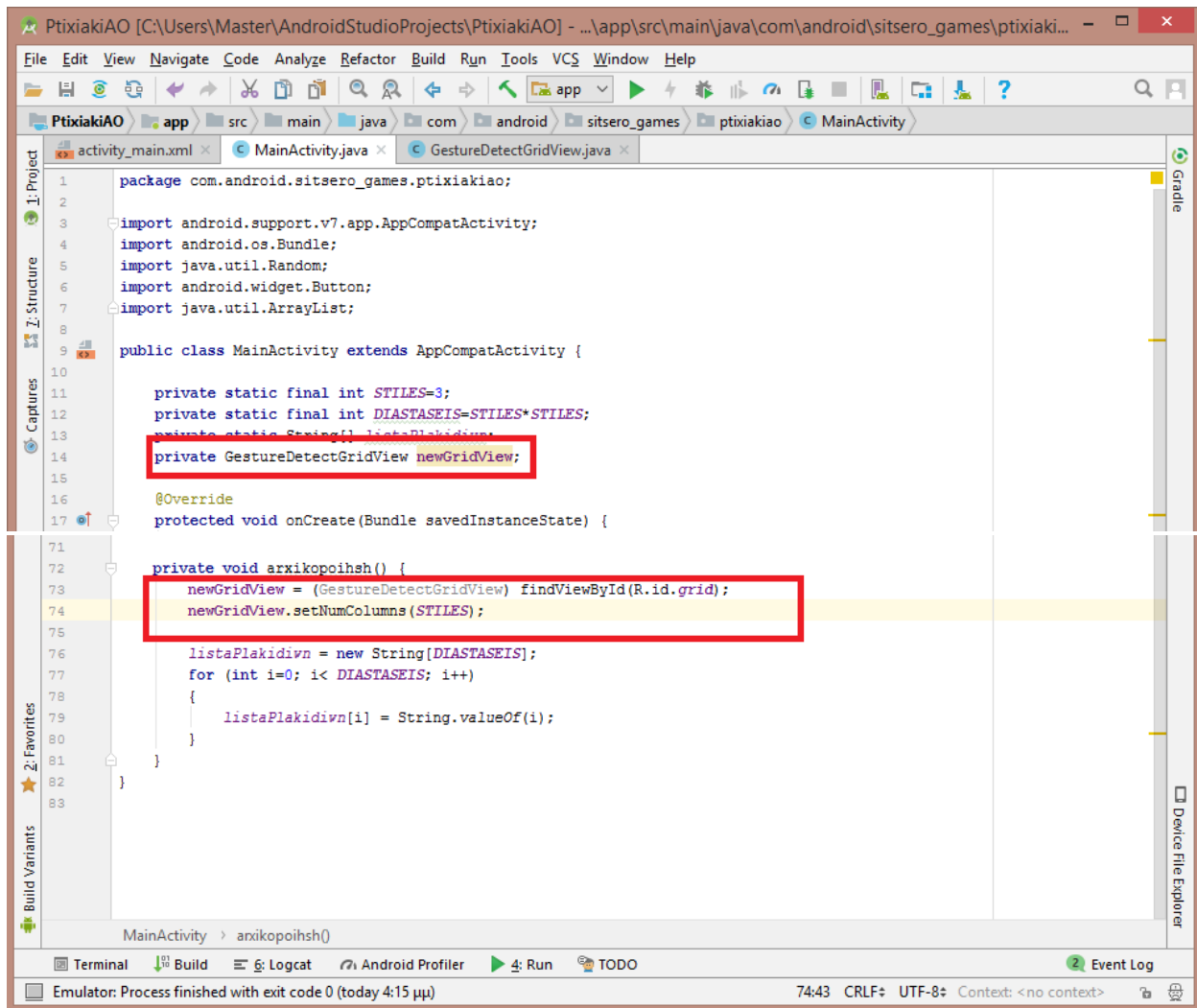
```
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115

@Override
public boolean onInterceptTouchEvent(MotionEvent ev) {
    int action = ev.getActionMasked();
    elegkths.onTouchEvent(ev);
    //Αν η χειρονομία ακυρώθηκε ή τελείωσε
    if (action == MotionEvent.ACTION_CANCEL || action == MotionEvent.ACTION_UP) {
        flingConfirmed = false;
        //Αλλιώς αν έχει ξεκινήσει χειρονομία(ACTION_DOWN παίρνει την μεταβλητή της αρχικής τοποθεσίας
        //της χειρονομίας) παίρνουμε τις μεταβλητές X και Y της οθόνης
    } else if (action == MotionEvent.ACTION_DOWN) {
        touchX = ev.getX();
        touchY = ev.getY();
    } else {
        //εαν τελειώσει η κίνηση
        if (flingConfirmed) {
            return true;
        }
        //παίρνουμε τις απόλυτες αποστάσεις του swipe, και αν είναι μέσα στα επιτρεπτά όρια προχωράμε κανονικά
        float dX = (Math.abs(ev.getX() - touchX));
        float dY = (Math.abs(ev.getY() - touchY));
        if ((dX > SWIPE_MIN_DISTANCE) || (dY > SWIPE_MIN_DISTANCE)) {
            flingConfirmed = true;
            return true;
        }
    }
    //Επιστρέφουμε τη μεταβλητή συμβάντος
    return super.onInterceptTouchEvent(ev);
}

@Override
public boolean onTouchEvent(MotionEvent ev)
{
    return elegkths.onTouchEvent(ev);
}
```

Οι δυο boolean onTouchEvent

Επιστρέφοντας στην MainActivity, ορίζουμε μια μεταβλητή τύπου GestureDetectGridView και την αρχικοποιούμε.



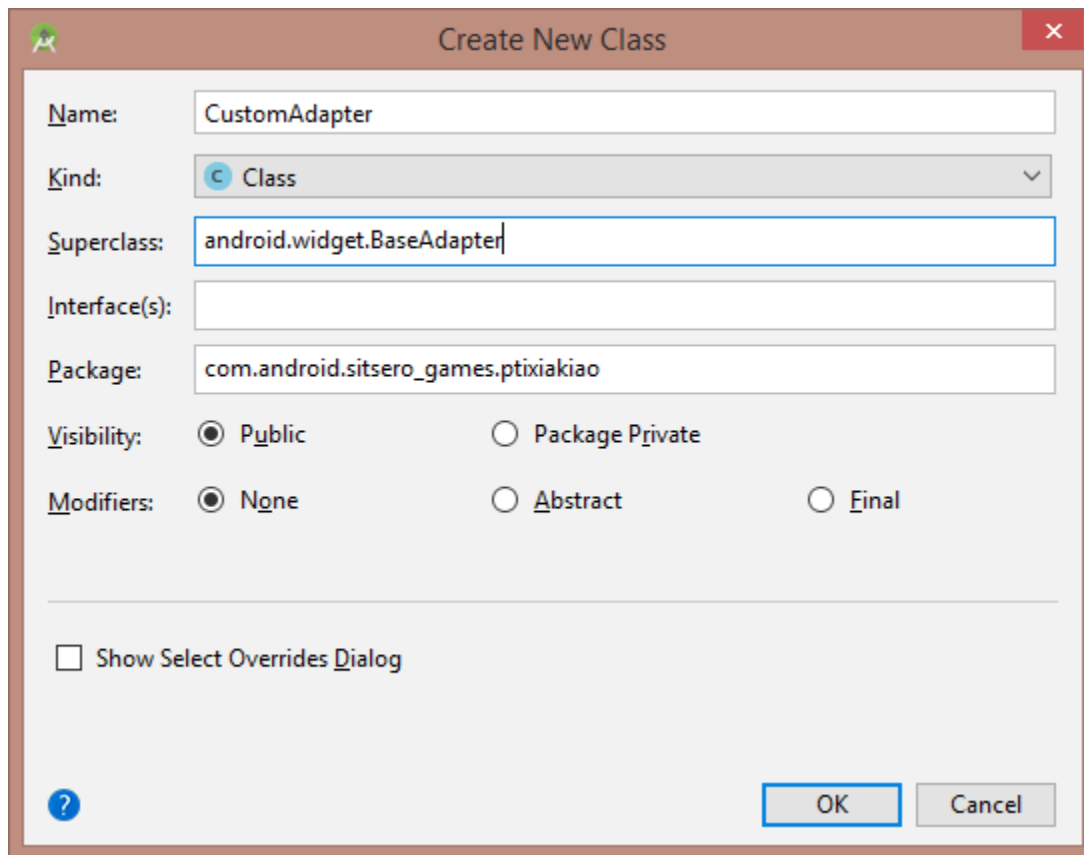
```
1 package com.android.sitsero_games.ptixiakiao;
2
3 import android.support.v7.app.AppCompatActivity;
4 import android.os.Bundle;
5 import java.util.Random;
6 import android.widget.Button;
7 import java.util.ArrayList;
8
9 public class MainActivity extends AppCompatActivity {
10
11     private static final int STILES=3;
12     private static final int DIASTASEIS=STILES*STILES;
13     private static String[] listaPlakidiwn;
14     private GestureDetectGridView newGridView;
15
16     @Override
17     protected void onCreate(Bundle savedInstanceState) {
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72     private void arxikopoihsh() {
73         newGridView = (GestureDetectGridView) findViewById(R.id.grid);
74         newGridView.setNumColumns(STILES);
75
76         listaPlakidiwn = new String[DIASTASEIS];
77         for (int i=0; i< DIASTASEIS; i++)
78         {
79             listaPlakidiwn[i] = String.valueOf(i);
80         }
81     }
82 }
83
```

Δήλωση και αρχικοποίηση GridView

3.2.5. ΔΗΜΙΟΥΡΓΙΑ CustomAdapter

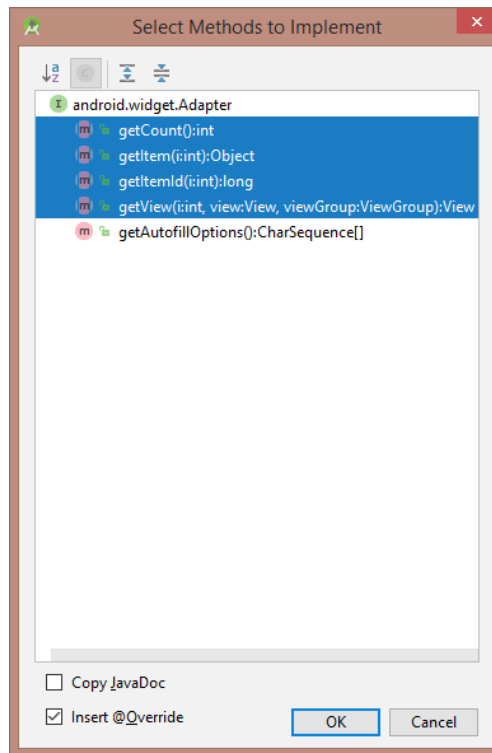
Θα χρειαστούμε, εκτός από το GestureDetectorGridView, κι έναν αντάπτορα για να συνδέει τα κουμπιά με τα πλακίδια.

Πατάμε δεξί κλικ στο όνομα εταιρίας μας->New->Java Class, και ονομάζουμε τη νέα κλάση CustomAdapter, η οποία θα κάνει extend στην BaseAdapter κλάση. Πατάμε OK.

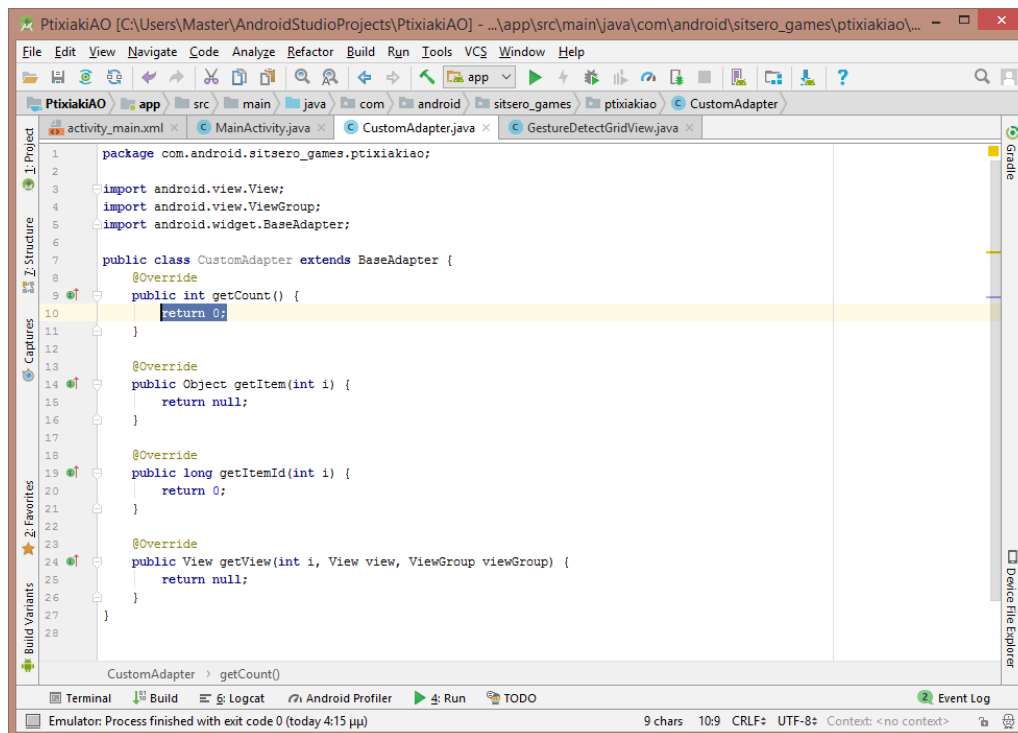


Δημιουργία κλάσης CustomAdapter

Στη συνέχεια θα δούμε ότι η κλάση θα χρειαστεί τους constructors της BaseAdapter, οπότε προσθέτουμε τους παρακάτω.

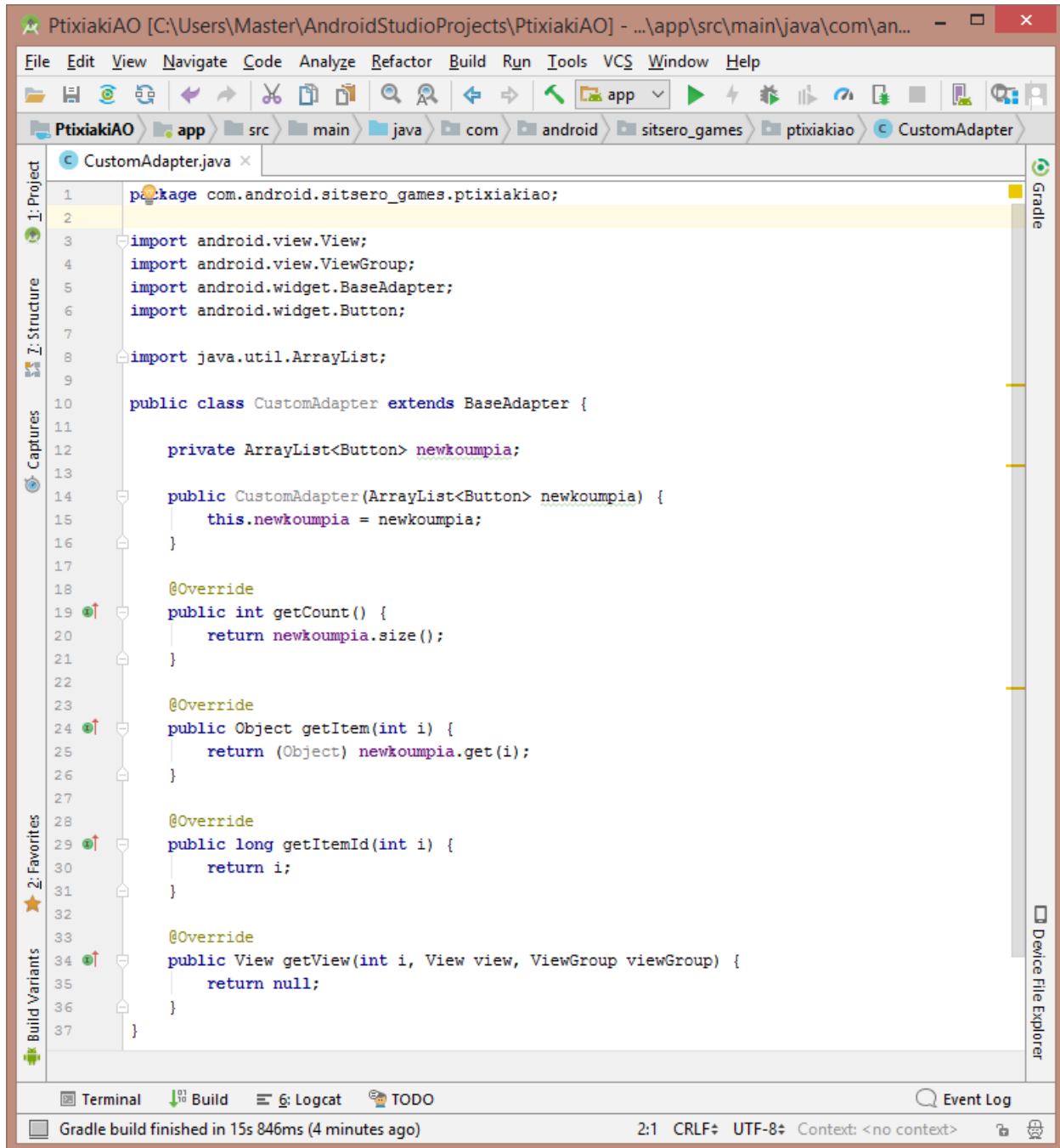


Δημιουργία constructors BaseAdapter 1/2



Δημιουργία constructors BaseAdapter 2/2

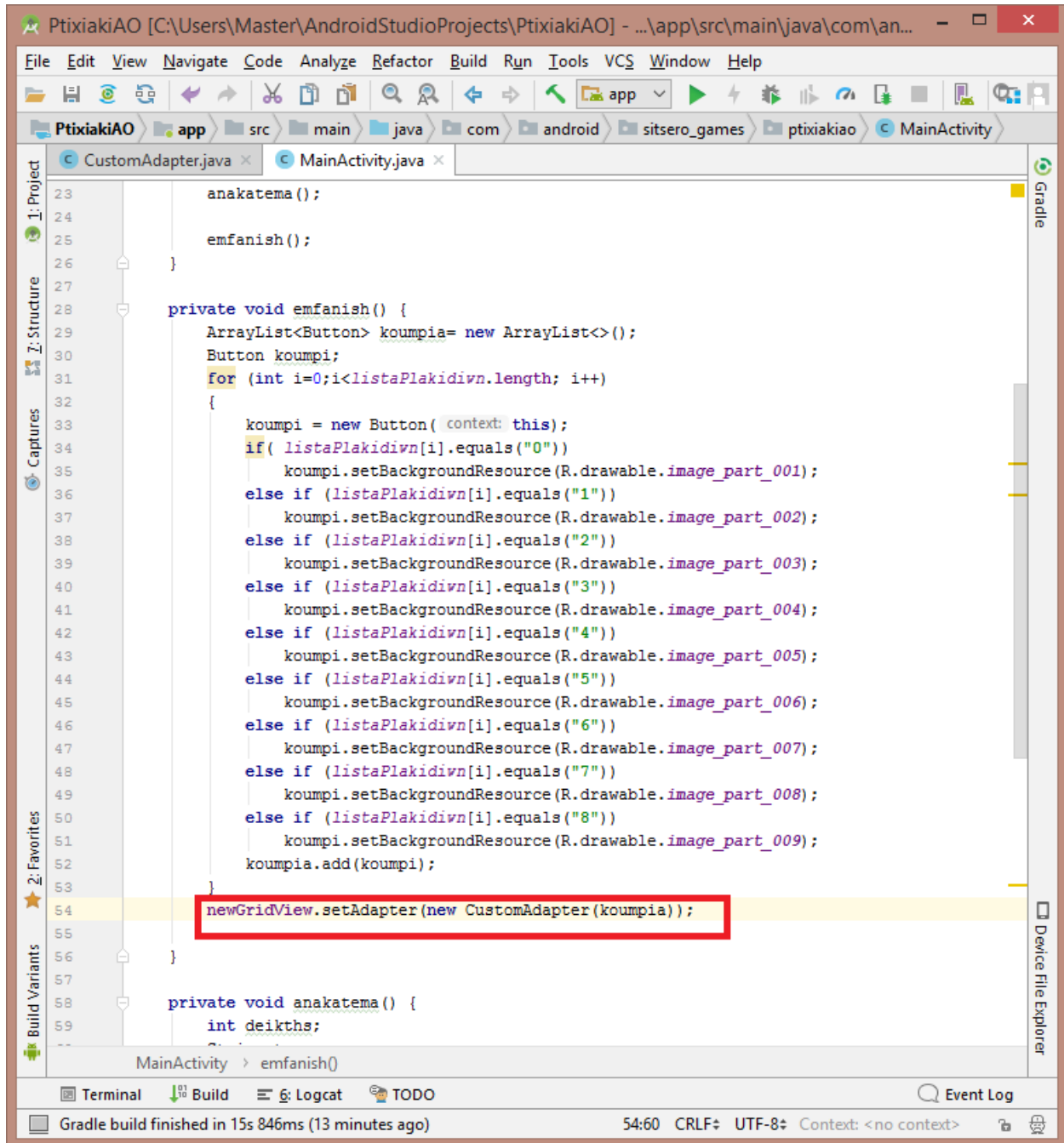
Επειδή θέλουμε να αναφέρεται στην λίστα που δημιουργήσαμε προηγουμένως, μέσα στη μέθοδο εμφάνισης, θα χρειαστεί να ορίσουμε μια λίστα και μέσα στη κλάση CustomAdapter. Εν συνεχεία δημιουργούμε και τον constructor της νέας λίστας και τροποποιούμε τους υπόλοιπους constructors έτσι ώστε να επιστρέφουν στοιχεία της λίστας, εκτός του τελευταίου (getView), τον οποίο θα αλλάξουμε αργότερα με μερικές ακόμη προσθήκες.



```
1 package com.android.sitsero_games.ptixiakiao;
2
3 import android.view.View;
4 import android.view.ViewGroup;
5 import android.widget.BaseAdapter;
6 import android.widget.Button;
7
8 import java.util.ArrayList;
9
10 public class CustomAdapter extends BaseAdapter {
11
12     private ArrayList<Button> newkoumpia;
13
14     public CustomAdapter(ArrayList<Button> newkoumpia) {
15         this.newkoumpia = newkoumpia;
16     }
17
18     @Override
19     public int getCount() {
20         return newkoumpia.size();
21     }
22
23     @Override
24     public Object getItem(int i) {
25         return (Object) newkoumpia.get(i);
26     }
27
28     @Override
29     public long getItemId(int i) {
30         return i;
31     }
32
33     @Override
34     public View getView(int i, View view, ViewGroup viewGroup) {
35         return null;
36     }
37 }
```

Τροποποίηση των constructors εκτός του getView

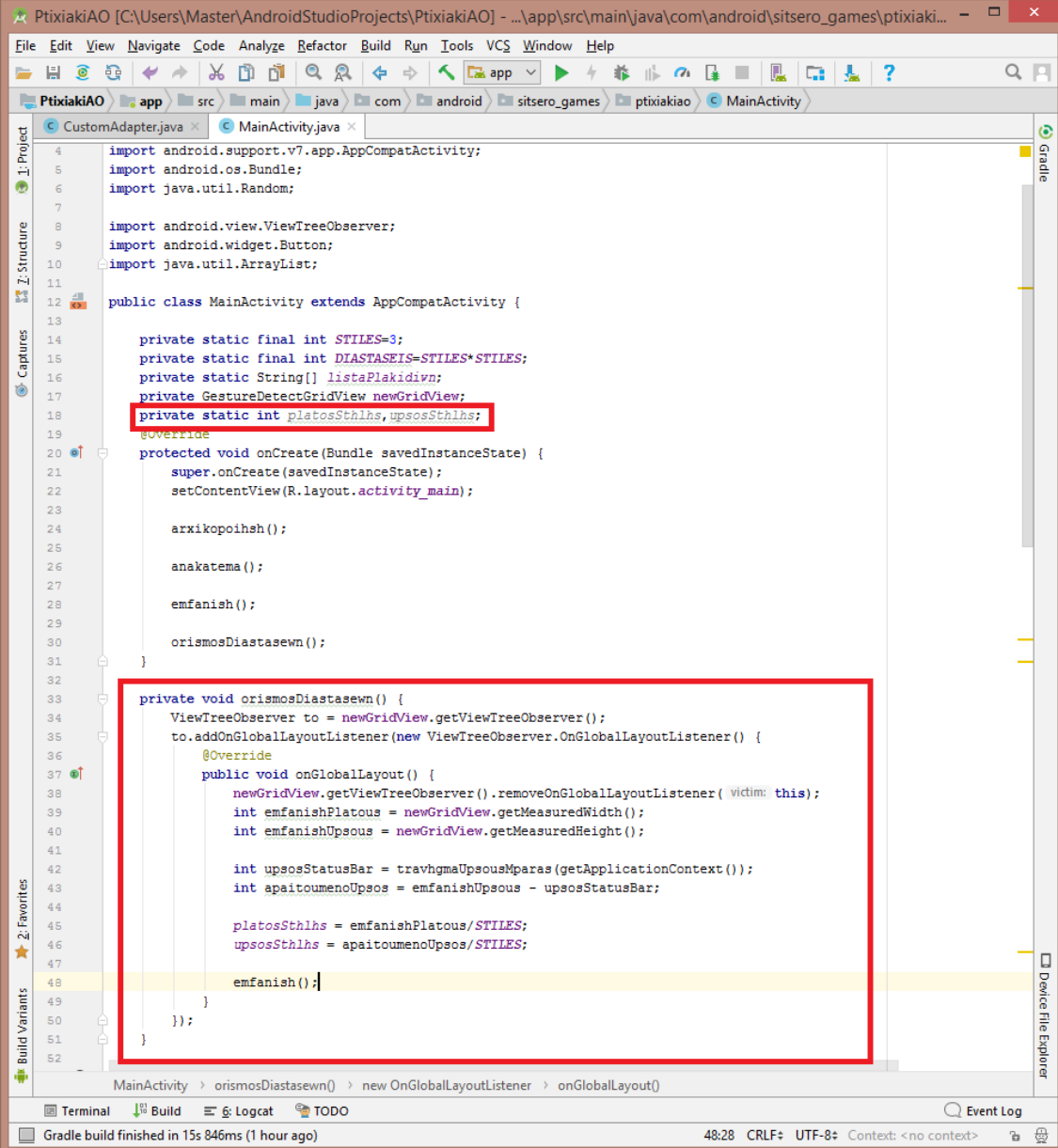
Προχωρώντας στην MainActivity.java, προσθέτουμε έναν αντάπτορα της GridView για το rendering των κουμπιών, μέσα στη μέθοδο εμφάνισης.



```
23     anakatema();
24
25     emfanish();
26 }
27
28 private void emfanish() {
29     ArrayList<Button> koumpia= new ArrayList<>();
30     Button koumpi;
31     for (int i=0;i<listaPlakidiwn.length; i++)
32     {
33         koumpi = new Button( context: this);
34         if( listaPlakidiwn[i].equals("0"))
35             koumpi.setBackgroundResource(R.drawable.image_part_001);
36         else if (listaPlakidiwn[i].equals("1"))
37             koumpi.setBackgroundResource(R.drawable.image_part_002);
38         else if (listaPlakidiwn[i].equals("2"))
39             koumpi.setBackgroundResource(R.drawable.image_part_003);
40         else if (listaPlakidiwn[i].equals("3"))
41             koumpi.setBackgroundResource(R.drawable.image_part_004);
42         else if (listaPlakidiwn[i].equals("4"))
43             koumpi.setBackgroundResource(R.drawable.image_part_005);
44         else if (listaPlakidiwn[i].equals("5"))
45             koumpi.setBackgroundResource(R.drawable.image_part_006);
46         else if (listaPlakidiwn[i].equals("6"))
47             koumpi.setBackgroundResource(R.drawable.image_part_007);
48         else if (listaPlakidiwn[i].equals("7"))
49             koumpi.setBackgroundResource(R.drawable.image_part_008);
50         else if (listaPlakidiwn[i].equals("8"))
51             koumpi.setBackgroundResource(R.drawable.image_part_009);
52         koumpia.add(koumpi);
53     }
54     newGridView.setAdapter(new CustomAdapter(koumpia));
55 }
56
57 private void anakatema() {
58     int deikths;
59     ...
60 }
```

Rendering κουμπιών μέσα σε μέθοδο εμφάνισης

Στη συνέχεια θα χρειαστεί να μετρήσουμε τις διαστάσεις κάθε κουμπιού στην οθόνη, αναλόγως της συσκευής που χρησιμοποιούμε. Για να το κάνουμε αυτό θα χρειαστούμε κλήση ενός `ViewTreeObserver`^[6]. Δηλώνουμε τις `int` σταθερές “πλάτος στήλης” και “ύψος στήλης”, καθώς θα τις χρειαστούμε για την μέθοδο εμφάνισης αλλά και για τον αντάπτορα. Ορίζουμε μια μέθοδο για τις διαστάσεις, η οποία καλείται στην `OnCreate`, και μέσα σε αυτή καλούμε έναν `ViewTreeObserver` ο οποίος θα αναφέρεται στο `GridView`. Μετά θα προσθέσουμε έναν `Listener` στον `TreeObserver`, ο οποίος θα παίρνει τις διαστάσεις κάθε κουμπιού του `Grid`. Στη συνέχεια ο `TreeObserver` ορίζει τις διαστάσεις ανάλογα με τη συσκευή, το μοντέλο και το ύψος της `status bar` κάθε συσκευής.



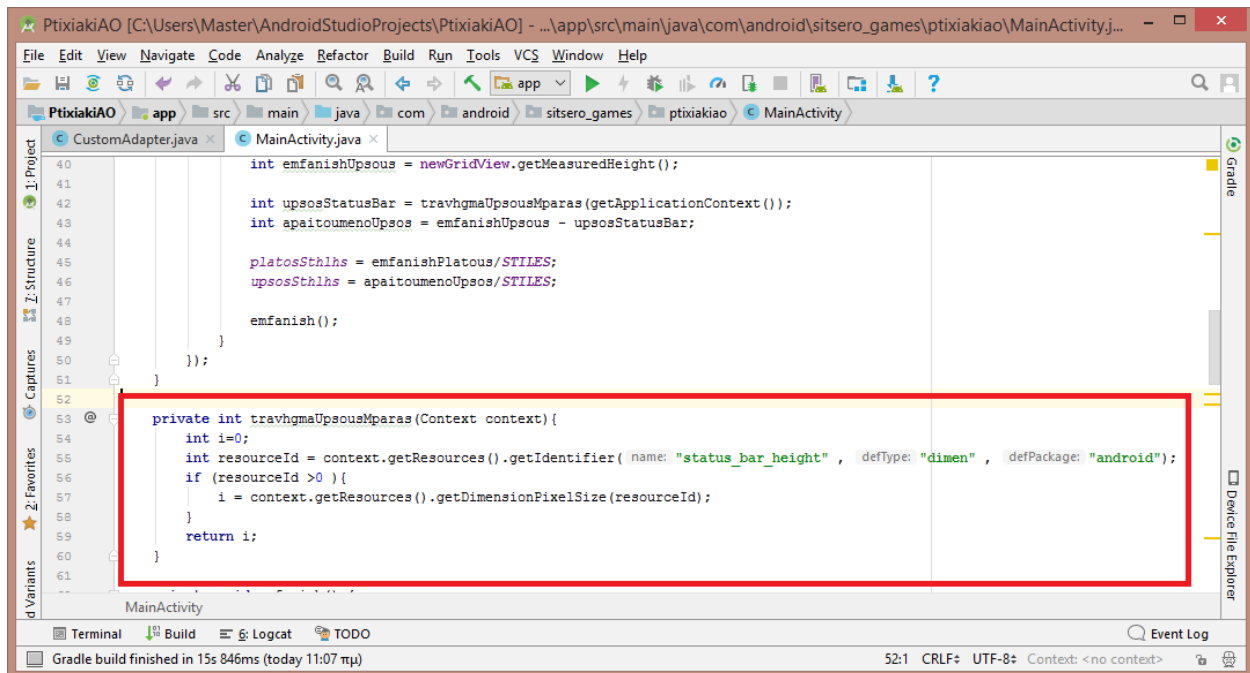
```

4 import android.support.v7.app.AppCompatActivity;
5 import android.os.Bundle;
6 import java.util.Random;
7
8 import android.view.ViewTreeObserver;
9 import android.widget.Button;
10 import java.util.ArrayList;
11
12 public class MainActivity extends AppCompatActivity {
13
14     private static final int STILES=3;
15     private static final int DIASTASEIS=STILES*STILES;
16     private static String[] listaPlakidiwn;
17     private GestureDetector newGridView;
18     private static int platosSthlhs,upsosSthlhs;
19
20     @Override
21     protected void onCreate(Bundle savedInstanceState) {
22         super.onCreate(savedInstanceState);
23         setContentView(R.layout.activity_main);
24
25         arxikopoihsh();
26
27         anakatema();
28
29         emfanish();
30
31         orismosDiastasewn();
32     }
33
34     private void orismosDiastasewn() {
35         ViewTreeObserver to = newGridView.getViewTreeObserver();
36         to.addOnGlobalLayoutListener(new ViewTreeObserver.OnGlobalLayoutListener() {
37             @Override
38             public void onGlobalLayout() {
39                 newGridView.getViewTreeObserver().removeOnGlobalLayoutListener( victim: this);
40                 int emfanishPlatous = newGridView.getMeasuredWidth();
41                 int emfanishUpsous = newGridView.getMeasuredHeight();
42
43                 int upsosStatusBar = travhgmaUpsousMparas(getApplicationContext());
44                 int apaitoumenoUpsos = emfanishUpsous - upsosStatusBar;
45
46                 platosSthlhs = emfanishPlatous/STILES;
47                 upsosSthlhs = apaitoumenoUpsos/STILES;
48
49                 emfanish();
50             }
51         });
52     }

```

Δημιουργία μεθόδου ορισμού διαστάσεων μέσω `ViewTreeObserver`

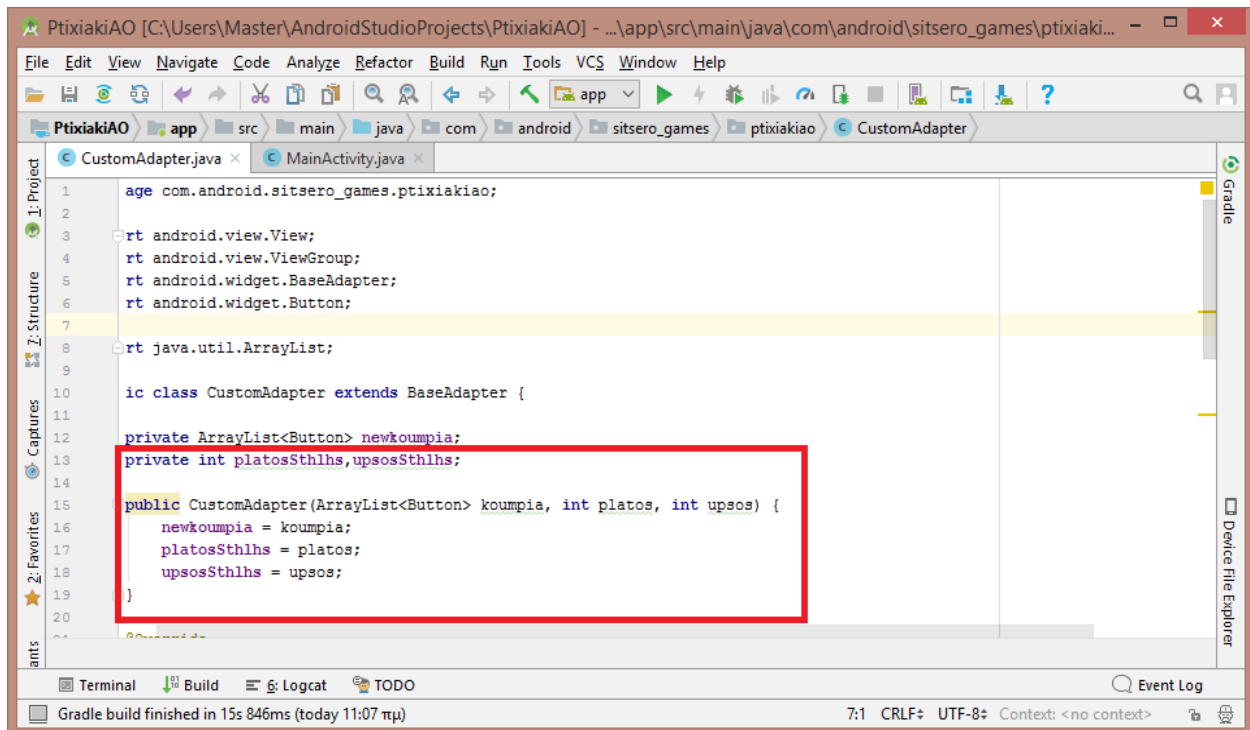
Όπως βλέπουμε γίνεται αναφορά σε μια μέθοδο “τραβήγματος” του ύψους της μπάρας. Μέσα σε αυτή τραβάμε το ύψος της μπάρας τύπου διαστάσεων από το πακέτο android, και επιστρέφουμε το μέγεθος σε pixels. Η συγκεκριμένη μέθοδος παρατίθεται παρακάτω:



```
40 int emfanishUpsous = newGridView.getMeasuredHeight();
41
42 int upsosStatusBar = travhgmaUpsousMparas(getApplicationContext());
43 int apaitoumenoUpsos = emfanishUpsous - upsosStatusBar;
44
45 platosSthlhs = emfanishPlatous/STILES;
46 upsosSthlhs = apaitoumenoUpsos/STILES;
47
48 emfanish();
49
50 }
51
52 }
53
54 private int travhgmaUpsousMparas(Context context) {
55     int i=0;
56     int resourceId = context.getResources().getIdentifier("status_bar_height", "dimen", "android");
57     if (resourceId > 0) {
58         i = context.getResources().getDimensionPixelSize(resourceId);
59     }
60     return i;
61 }
```

Απολαβή πληροφοριών ύψους μπάρας Android αναλόγως συσκευής

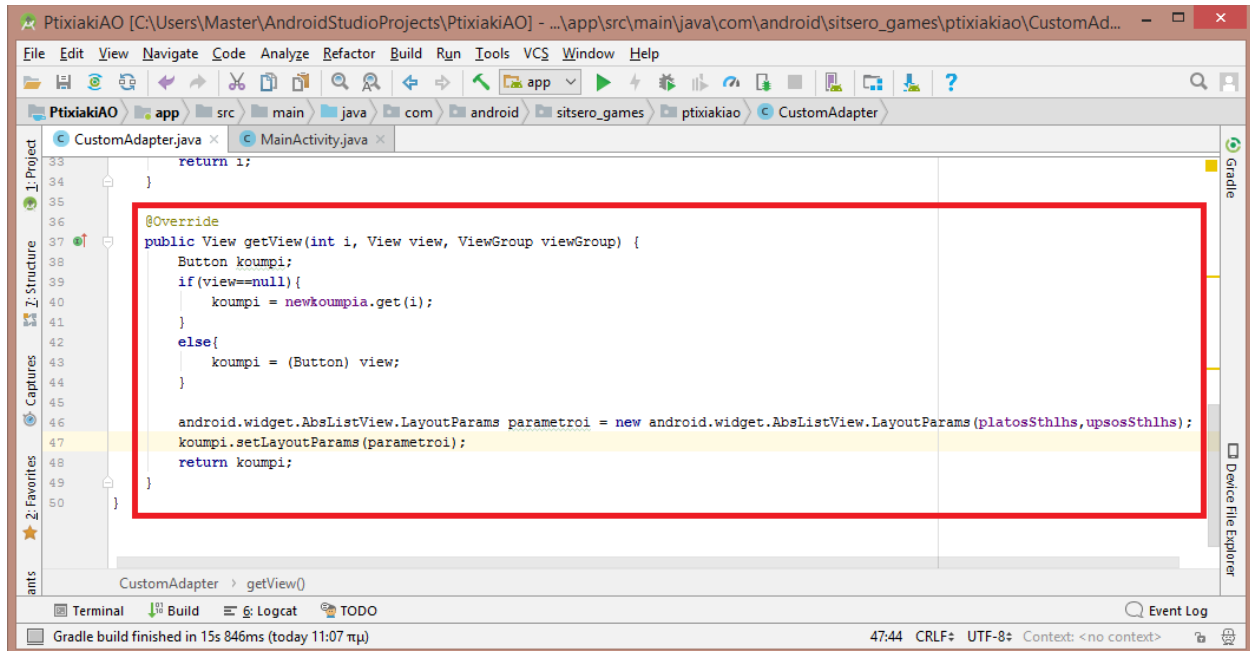
Συνεχίζοντας, εφόσον έχουμε τις διαστάσεις, περνάμε στην περαιτέρω παραμετροποίηση του αντάπτορά μας. Αρχικά, αρχικοποιούμε το ύψος και το πλάτος των στηλών στον constructor.



```
1  age com.android.sitsero_games.ptixiakiao;
2
3  rt android.view.View;
4  rt android.view.ViewGroup;
5  rt android.widget.BaseAdapter;
6  rt android.widget.Button;
7
8  rt java.util.ArrayList;
9
10 ic class CustomAdapter extends BaseAdapter {
11
12     private ArrayList<Button> newkoupia;
13     private int platosSthlhs, upsosSthlhs;
14
15     public CustomAdapter(ArrayList<Button> koumpia, int platos, int upsos) {
16         newkoupia = koumpia;
17         platosSthlhs = platos;
18         upsosSthlhs = upsos;
19     }
20
21 }
```

Παραμετροποίηση αντάπτορα 1/2

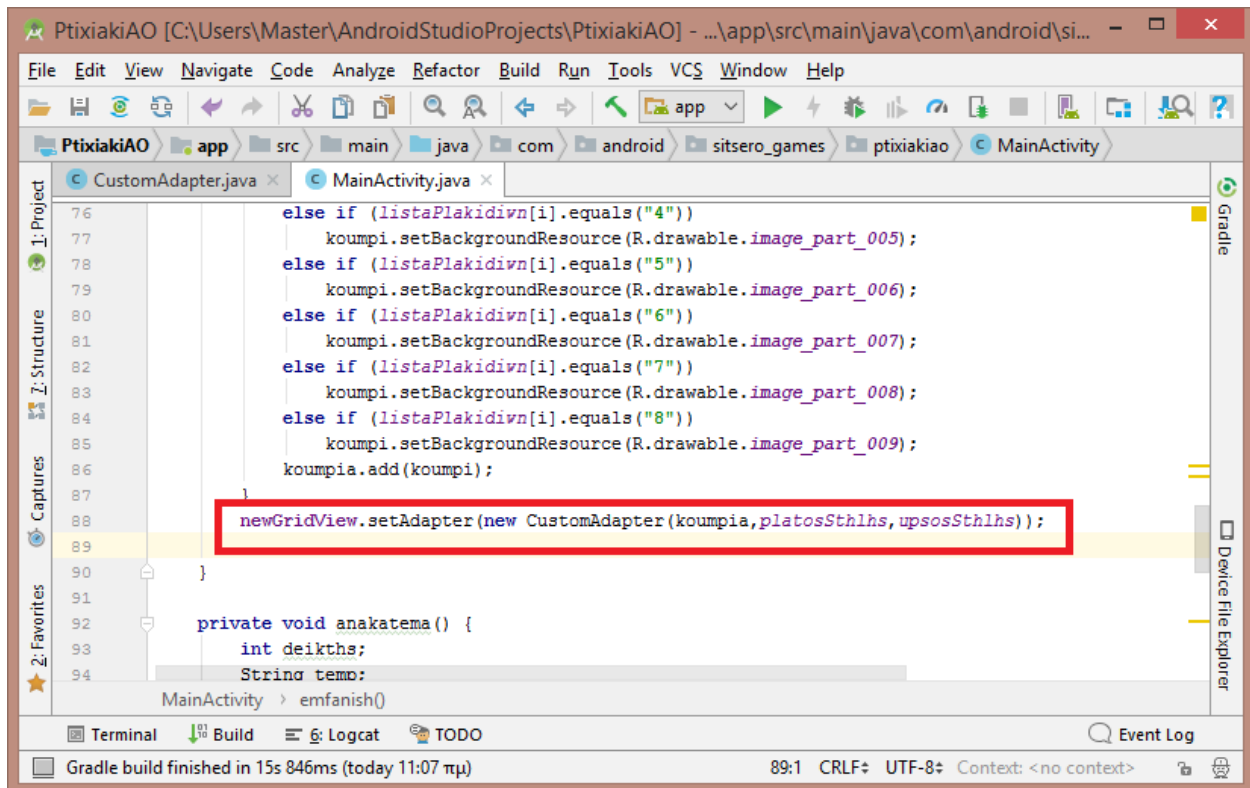
Στη συνέχεια τροποποιούμε την getView την οποία είχαμε αφήσει ανέγγιχτη προηγουμένως. Μέσα στην getView θα γίνεται όλο το rendering, οπότε τραβάμε τις παραμέτρους κάθε κουμπιού με τη βοήθεια των σταθερών πλάτους και ύψους στηλών, τις ορίζουμε στο κουμπί και επιστρέφουμε το κουμπί.



```
33     return i;
34 }
35
36
37 @Override
38 public View getView(int i, View view, ViewGroup viewGroup) {
39     Button koumpi;
40     if (view == null) {
41         koumpi = newKoumpia.get(i);
42     }
43     else {
44         koumpi = (Button) view;
45     }
46
47     android.widget.AbsListView.LayoutParams parametroi = new android.widget.AbsListView.LayoutParams(platosSthlhs, upsosSthlhs);
48     koumpi.setLayoutParams(parametroi);
49     return koumpi;
50 }
```

Παραμετροποίηση αντίκτυπα 2/2

Επιστρέφοντας στο MainActivity, θα προσθέσουμε τις 2 επιπλέον παραμέτρους στο set του αντάπτορα μέσα στη μέθοδο Εμφάνισης.

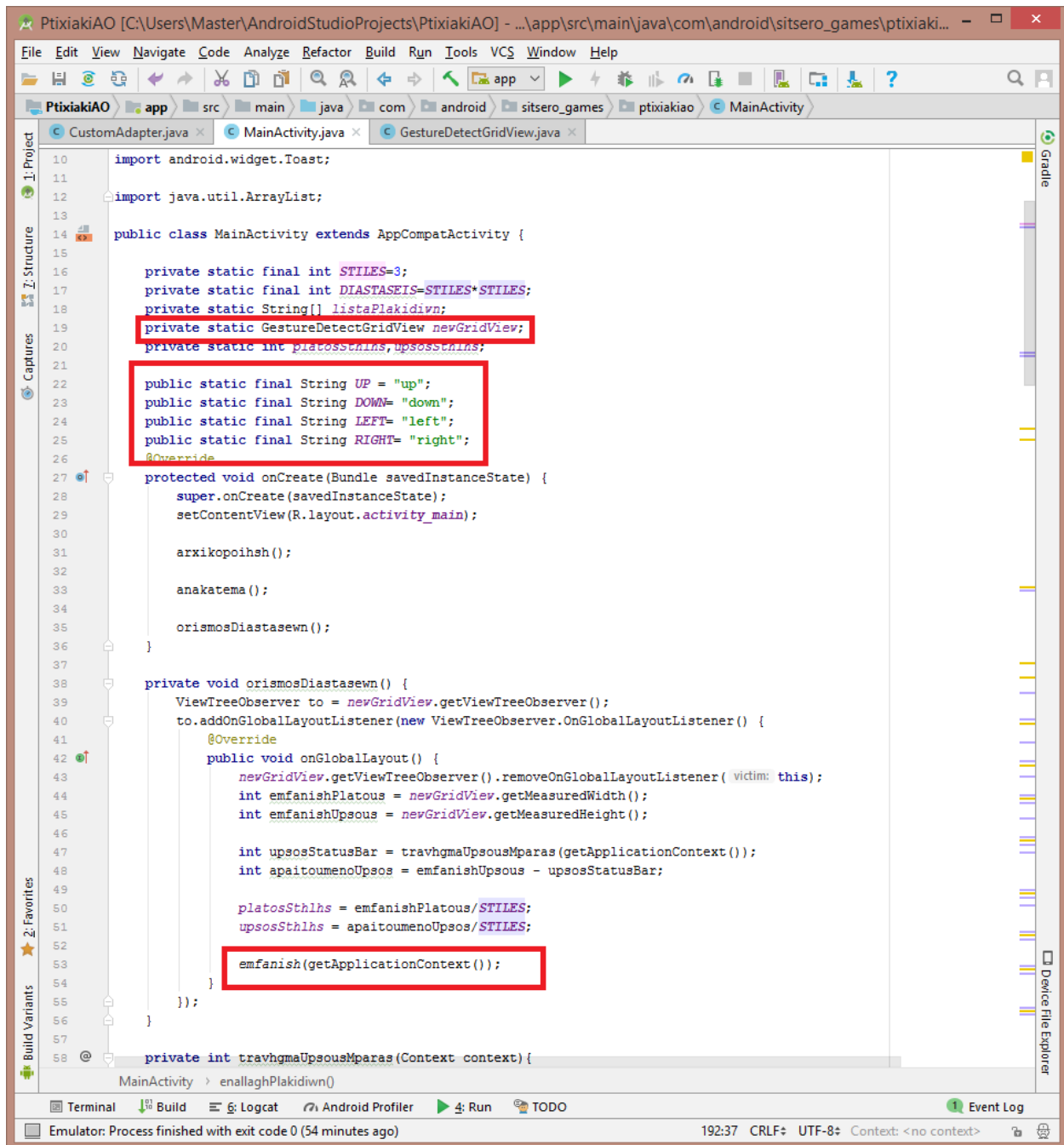


```
File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help
Ptixiakiao [C:\Users\Master\AndroidStudioProjects\Ptixiakiao] - ...\app\src\main\java\com\android\si...
Ptixiakiao app src main java com android sitsero_games ptixiakiao MainActivity
CustomAdapter.java x MainActivity.java x
76 else if (listaPlakidiwn[i].equals("4"))
77     koumpi.setBackgroundResource(R.drawable.image_part_005);
78 else if (listaPlakidiwn[i].equals("5"))
79     koumpi.setBackgroundResource(R.drawable.image_part_006);
80 else if (listaPlakidiwn[i].equals("6"))
81     koumpi.setBackgroundResource(R.drawable.image_part_007);
82 else if (listaPlakidiwn[i].equals("7"))
83     koumpi.setBackgroundResource(R.drawable.image_part_008);
84 else if (listaPlakidiwn[i].equals("8"))
85     koumpi.setBackgroundResource(R.drawable.image_part_009);
86 koumpia.add(koumpi);
87
88 newGridView.setAdapter(new CustomAdapter(koumpia, platosSthlhs, upsosSthlhs));
89
90 }
91
92 private void anakatema() {
93     int deikths;
94     String temo;
```

Terminal Build Logcat TODO Event Log
Gradle build finished in 15s 846ms (today 11:07 πμ) 89:1 CRLF UTF-8 Context: <no context>

Προσθήκη επιπλέον παραμέτρων

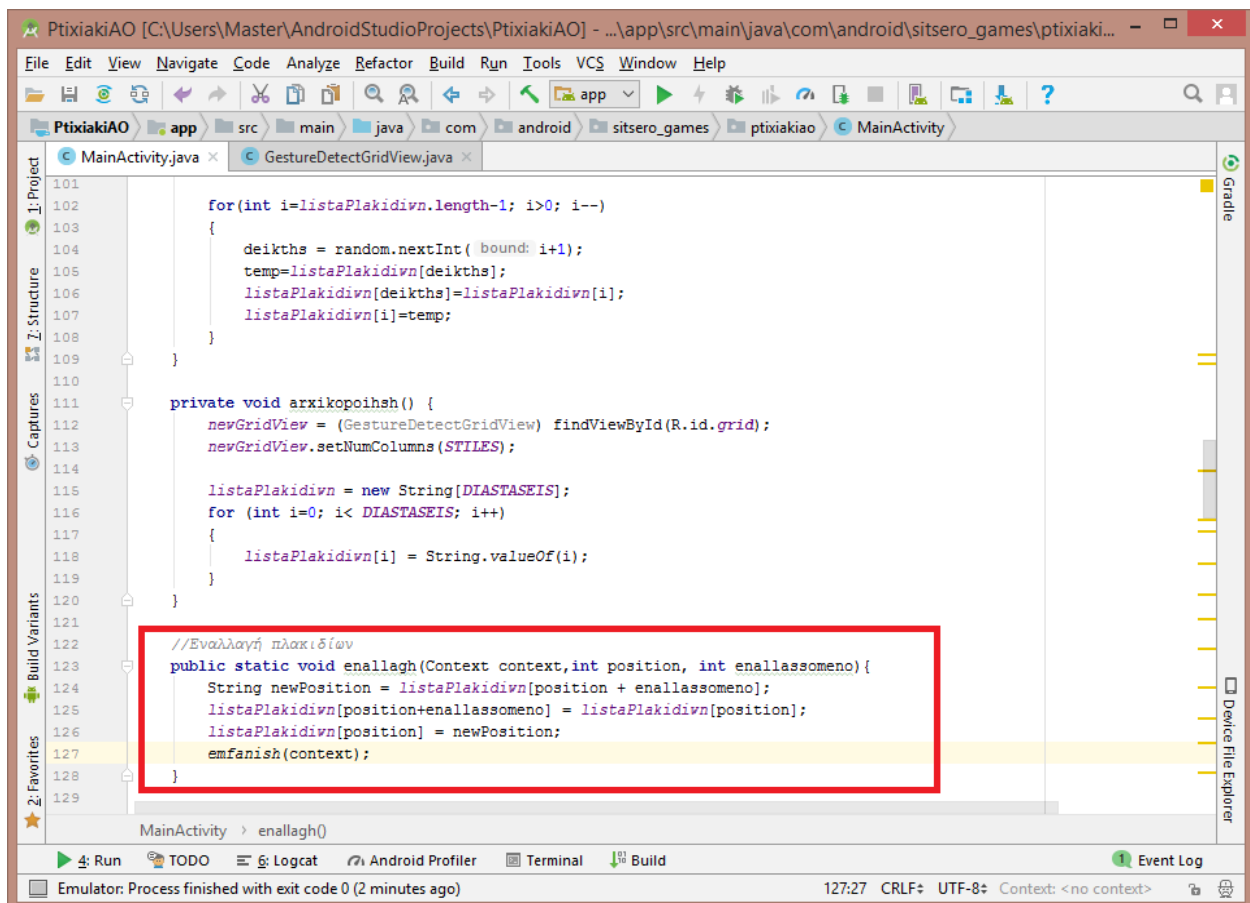
Έφτασε η ώρα να δημιουργήσουμε τη μέθοδο μέσω της οποίας θα μετακινούμε τα πλακίδια(μέσα στη MainActivity). Αρχικά δηλώνουμε 4 String σταθερές, μια για κάθε κατεύθυνση, κάνουμε στατική την GridView και σβήνουμε τη μέθοδο εμφάνισης από την OnCreate, εφόσον βάλαμε κλήση της μέσα στη μέθοδο ορισμού διαστάσεων .



```
10 import android.widget.Toast;
11
12 import java.util.ArrayList;
13
14 public class MainActivity extends AppCompatActivity {
15
16     private static final int STILES=3;
17     private static final int DIASTASEIS=STILES*STILES;
18     private static String[] listaPlakidiwn;
19     private static GestureDetectorGridView newGridView;
20     private static int platosSthlhs,upsosSthlhs;
21
22     public static final String UP = "up";
23     public static final String DOWN= "down";
24     public static final String LEFT= "left";
25     public static final String RIGHT= "right";
26     @Override
27     protected void onCreate(Bundle savedInstanceState) {
28         super.onCreate(savedInstanceState);
29         setContentView(R.layout.activity_main);
30
31         arxikopoihsh();
32
33         anakatema();
34
35         orismosDiastasewn();
36     }
37
38     private void orismosDiastasewn() {
39         ViewTreeObserver to = newGridView.getViewTreeObserver();
40         to.addOnGlobalLayoutListener(new ViewTreeObserver.OnGlobalLayoutListener() {
41             @Override
42             public void onGlobalLayout() {
43                 newGridView.getViewTreeObserver().removeOnGlobalLayoutListener( victim: this);
44                 int emfanishPlatous = newGridView.getMeasuredWidth();
45                 int emfanishUpsous = newGridView.getMeasuredHeight();
46
47                 int upsosStatusBar = travhgmaUpsousMparas(getApplicationContext());
48                 int apaitoumenoUpsos = emfanishUpsous - upsosStatusBar;
49
50                 platosSthlhs = emfanishPlatous/STILES;
51                 upsosSthlhs = apaitoumenoUpsos/STILES;
52
53                 emfanish(getApplicationContext());
54             }
55         });
56     }
57
58     private int travhgmaUpsousMparas(Context context){
```

Δηλώσεις GridView και Strings, καθώς και αφαίρεση κλήσης εμφάνισης από OnCreate

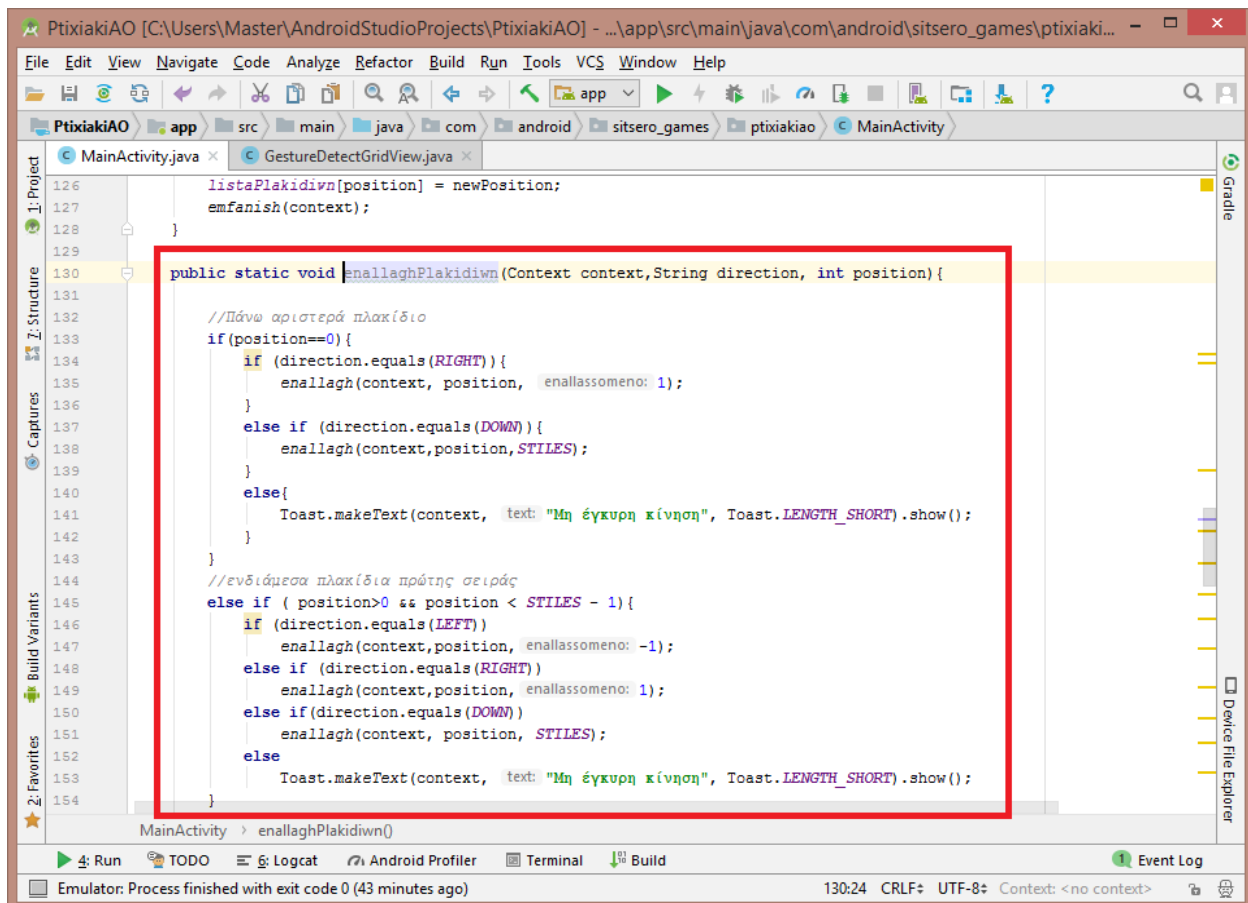
Φτιάχνουμε μια μέθοδο Εναλλαγής, παρόμοια με αυτή που υπάρχει μέσα στο ανακάτεμα, η οποία θα καλείται από την μέθοδο Εναλλαγής Πλακιδίων για να εναλλάσσει τα πλακίδια στα οποία προσπαθεί να κάνει swiρε ο χρήστης.



```
101
102     for(int i=listaPlakidiwn.length-1; i>0; i--)
103     {
104         deikths = random.nextInt( bound: i+1);
105         temp=listaPlakidiwn[deikths];
106         listaPlakidiwn[deikths]=listaPlakidiwn[i];
107         listaPlakidiwn[i]=temp;
108     }
109
110
111     private void arxikopoihsh() {
112         newGridView = (GestureDetectGridView) findViewById(R.id.grid);
113         newGridView.setNumColumns( STILES );
114
115         listaPlakidiwn = new String[DIASTASEIS];
116         for (int i=0; i< DIASTASEIS; i++)
117         {
118             listaPlakidiwn[i] = String.valueOf(i);
119         }
120     }
121
122     //Εναλλαγή πλακιδίων
123     public static void enallagh(Context context,int position, int enallassomeno){
124         String newPosition = listaPlakidiwn[position + enallassomeno];
125         listaPlakidiwn[position+enallassomeno] = listaPlakidiwn[position];
126         listaPlakidiwn[position] = newPosition;
127         emfanish(context);
128     }
129
```

Μέθοδος εναλλαγής

Εν συνεχεία, δημιουργούμε τη μέθοδο Εναλλαγής Πλακιδίων. Σε αυτή ξεχωρίζουμε κάθε θέση πλακιδίου, και της φερόμαστε ανάλογα. Παραδείγματος χάρη, για το πάνω αριστερό πλακίδιο οι θέσεις με τις οποίες μπορεί να εναλλαχθεί είναι προς τα δεξιά και προς τα κάτω. Άρα, αναλόγως της θέσης του πλακιδίου(0-8), ορίζουμε τις κατευθύνσεις που είναι έγκυρες. Σε όσες κινήσεις είναι άκυρες, δημιουργούμε splash(Toast) text μικρής χρονικής διάρκειας που να ενημερώνει τον χρήστη ότι έκανε λάθος κίνηση. Ομοίως, στο δεύτερο πλακίδιο οι επιτρεπτές κινήσεις είναι αυτές προς τα αριστερά, δεξιά και κάτω.

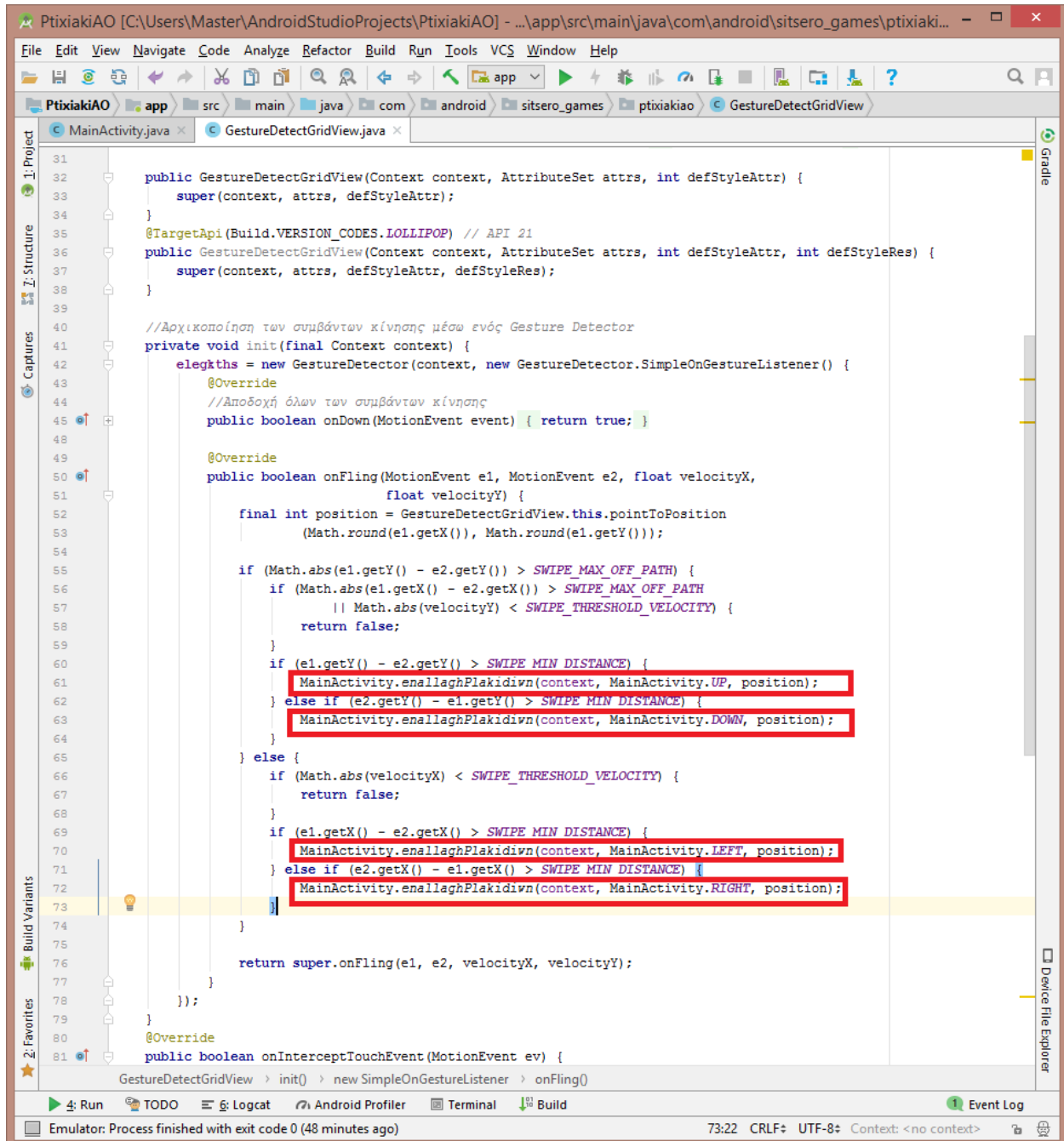


```
126     listaPlakidiwn[position] = newPosition;
127     emfanish(context);
128 }
129
130 public static void enallaghPlakidiwn(Context context, String direction, int position){
131
132     //Πάνω αριστερά πλακίδιο
133     if(position==0){
134         if (direction.equals(RIGHT)){
135             enallagh(context, position, enallassomeno: 1);
136         }
137         else if (direction.equals(DOWN)){
138             enallagh(context, position, STILES);
139         }
140         else{
141             Toast.makeText(context, text: "Μη έγκυρη κίνηση", Toast.LENGTH_SHORT).show();
142         }
143     }
144     //ενδιάμεσα πλακίδια πρώτης σειράς
145     else if ( position>0 && position < STILES - 1){
146         if (direction.equals(LEFT))
147             enallagh(context, position, enallassomeno: -1);
148         else if (direction.equals(RIGHT))
149             enallagh(context, position, enallassomeno: 1);
150         else if(direction.equals(DOWN))
151             enallagh(context, position, STILES);
152         else
153             Toast.makeText(context, text: "Μη έγκυρη κίνηση", Toast.LENGTH_SHORT).show();
154     }
155 }
```

Μέθοδος εναλλαγής πλακιδίων

Συνεχίζουμε με τον ίδιο τρόπο για όλες τις θέσεις πλακιδίων, όπως θα φανεί και στο ΠΑΡΑΡΤΗΜΑ Α της βιβλιογραφίας.

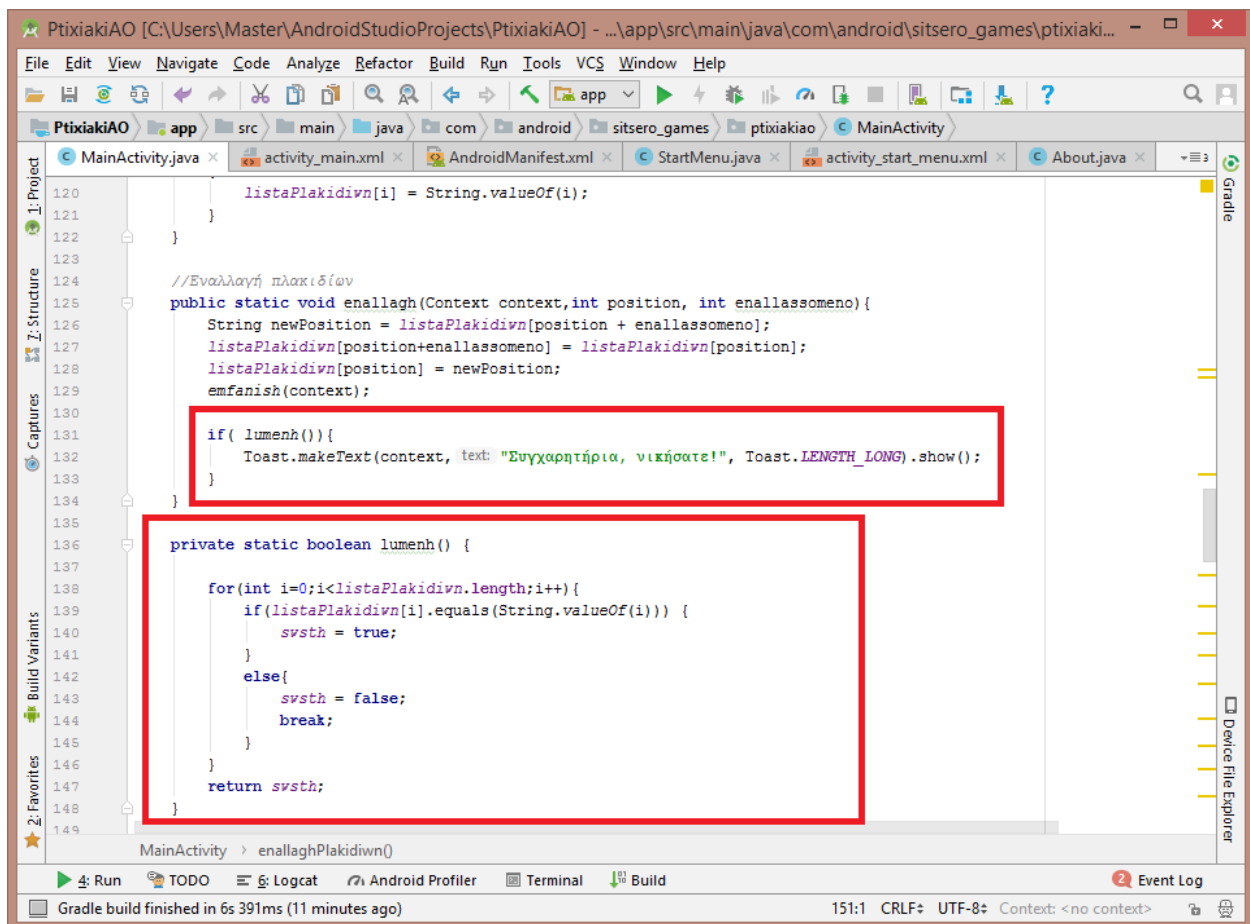
Για να δουλέψει ο κώδικας με τις κατευθύνσεις, επιστρέφουμε στη GestureDetectGridView και προσθέτουμε τις κλήσεις της Εναλλαγής Πλακιδίων:



```
31
32 public GestureDetectGridView(Context context, AttributeSet attrs, int defStyleAttr) {
33     super(context, attrs, defStyleAttr);
34 }
35 @TargetApi(Build.VERSION_CODES.LOLLIPOP) // API 21
36 public GestureDetectGridView(Context context, AttributeSet attrs, int defStyleAttr, int defStyleRes) {
37     super(context, attrs, defStyleAttr, defStyleRes);
38 }
39
40 //Αρχικοποίηση των συμβάντων κίνησης μέσω ενός Gesture Detector
41 private void init(final Context context) {
42     elegkths = new GestureDetector(context, new GestureDetector.SimpleOnGestureListener() {
43         @Override
44         //Αποδοχή όλων των συμβάντων κίνησης
45         public boolean onDown(MotionEvent event) { return true; }
46
47         @Override
48         public boolean onFling(MotionEvent e1, MotionEvent e2, float velocityX,
49             float velocityY) {
50             final int position = GestureDetectGridView.this.pointToPosition
51                 (Math.round(e1.getX()), Math.round(e1.getY()));
52
53             if (Math.abs(e1.getY() - e2.getY()) > SWIPE_MAX_OFF_PATH) {
54                 if (Math.abs(e1.getX() - e2.getX()) > SWIPE_MAX_OFF_PATH
55                     || Math.abs(velocityY) < SWIPE_THRESHOLD_VELOCITY) {
56                     return false;
57                 }
58                 if (e1.getY() - e2.getY() > SWIPE_MIN_DISTANCE) {
59                     MainActivity.enallaghPlakidiwn(context, MainActivity.UP, position);
60                 } else if (e2.getY() - e1.getY() > SWIPE_MIN_DISTANCE) {
61                     MainActivity.enallaghPlakidiwn(context, MainActivity.DOWN, position);
62                 }
63             } else {
64                 if (Math.abs(velocityX) < SWIPE_THRESHOLD_VELOCITY) {
65                     return false;
66                 }
67                 if (e1.getX() - e2.getX() > SWIPE_MIN_DISTANCE) {
68                     MainActivity.enallaghPlakidiwn(context, MainActivity.LEFT, position);
69                 } else if (e2.getX() - e1.getX() > SWIPE_MIN_DISTANCE) {
70                     MainActivity.enallaghPlakidiwn(context, MainActivity.RIGHT, position);
71                 }
72             }
73         }
74     });
75
76     return super.onFling(e1, e2, velocityX, velocityY);
77 }
78
79 }
80 @Override
81 public boolean onInterceptTouchEvent(MotionEvent ev) {
```

Προσθήκη εναλλαγής πλακιδίων στην αρχικοποίηση του ελεγκτή

Τέλος, φτιάχνουμε μια boolean μέθοδο μέσω της οποίας εάν λύθηκε το παζλ, εμφανίζουμε ένα splash (Toast) text, το οποίο ενημερώνει τον χρήστη ότι νίκησε. Αυτή η μέθοδος θα καλείται μέσα στην Εναλλαγή και θα ελέγχει αν όλα τα κουμπιά είναι στη σωστή θέση. Επίσης προσθέτουμε την σημαία “Σωστή” στις δηλώσεις της MainActivity ως `public static boolean swsth = false;`

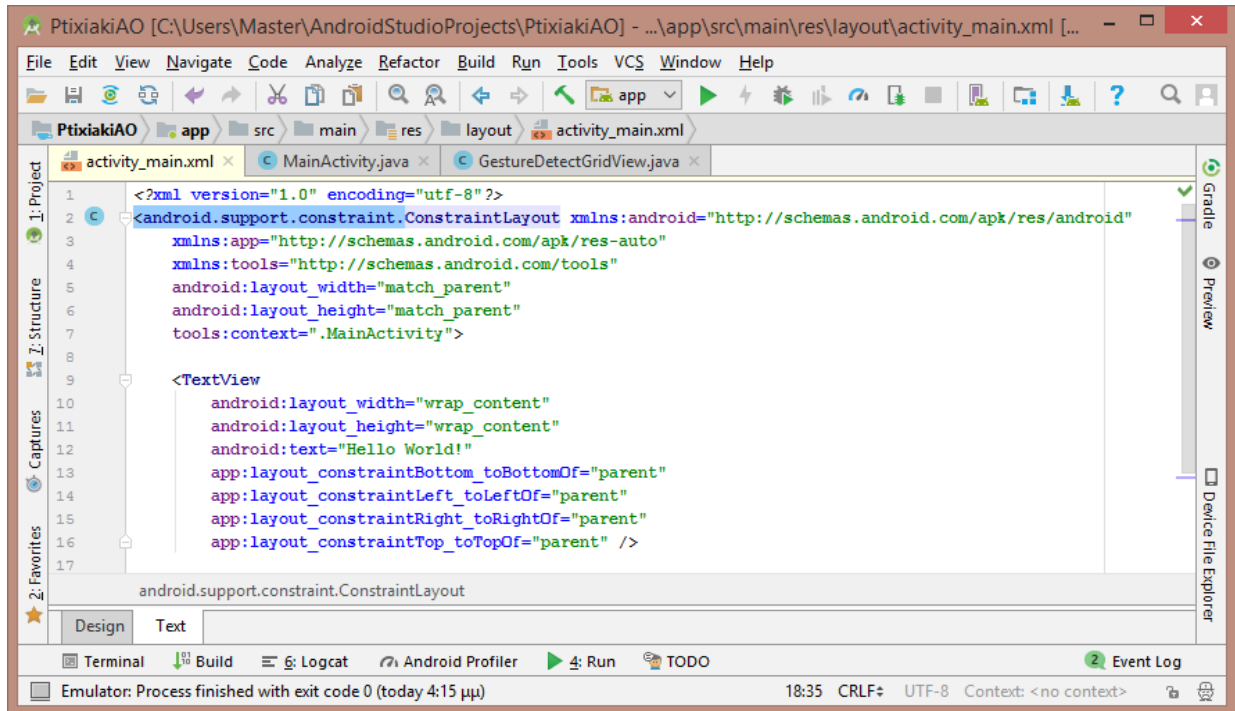


```
120         listaPlakidiwn[i] = String.valueOf(i);
121     }
122 }
123
124 //Εναλλαγή πλακιδίων
125 public static void enallagh(Context context,int position, int enallassomeno){
126     String newPosition = listaPlakidiwn[position + enallassomeno];
127     listaPlakidiwn[position+enallassomeno] = listaPlakidiwn[position];
128     listaPlakidiwn[position] = newPosition;
129     emfanish(context);
130
131     if( lumenh()){
132         Toast.makeText(context, text: "Ευχαρητήρια, νικήσατε!", Toast.LENGTH_LONG).show();
133     }
134 }
135
136 private static boolean lumenh() {
137
138     for(int i=0;i<listaPlakidiwn.length;i++){
139         if(listaPlakidiwn[i].equals(String.valueOf(i))) {
140             swsth = true;
141         }
142         else{
143             swsth = false;
144             break;
145         }
146     }
147     return swsth;
148 }
149 }
```

Δημιουργία μεθόδου λύσης

Προχωρώντας, ανοίγουμε το activity_main.xml, με σκοπό να το τροποποιήσουμε ώστε να φαίνεται το GridView που φτιάξαμε σε πλήρη οθόνη, χωρίς περιορισμούς(μπάρα τίτλου εφαρμογής κλπ). Για να το κάνουμε αυτό, αντικαθιστούμε το **“android.support.constraint.ConstraintLayout”** με το GridView που φτιάξαμε όπως φαίνεται στις εικόνες. Επιπλέον, αφαιρούμε το αρχικό TextView “Hello World” που έχει ως default xml. Προσθέτουμε και ένα android id για να μπορούμε να κάνουμε refer στο Grid.

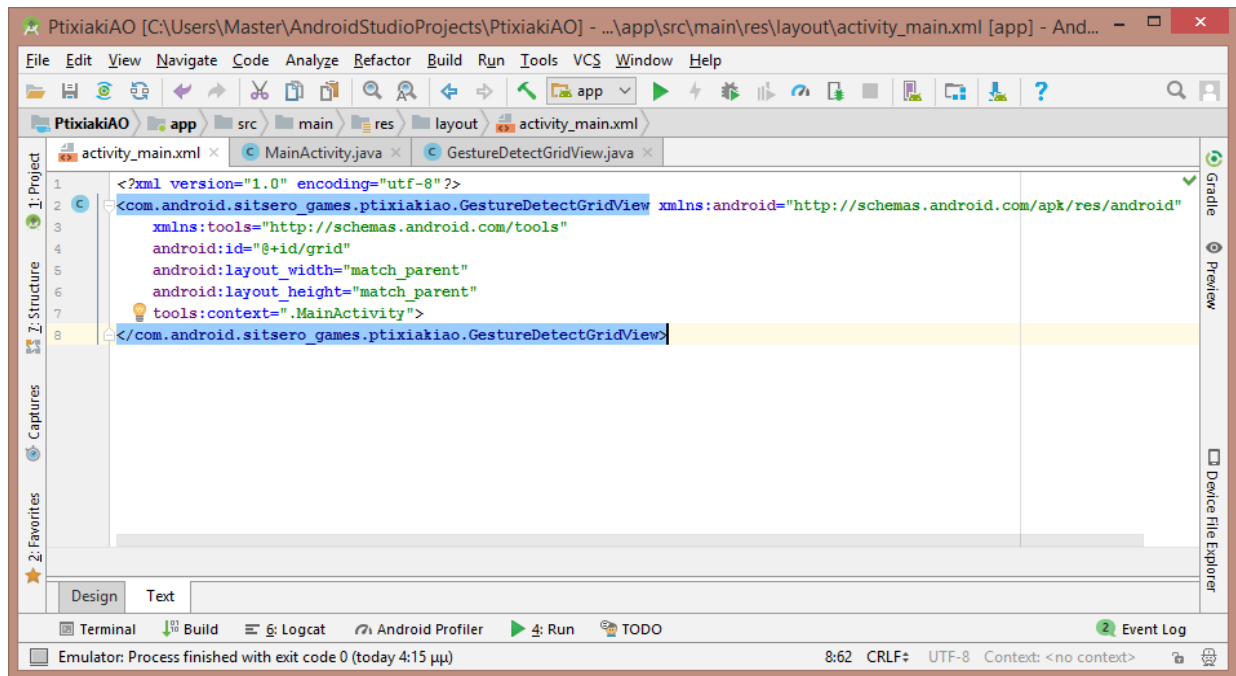
Πριν:



```
<?xml version="1.0" encoding="utf-8" ?>
<android.support.constraint.ConstraintLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".MainActivity">
    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Hello World!"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintLeft_toLeftOf="parent"
        app:layout_constraintRight_toRightOf="parent"
        app:layout_constraintTop_toTopOf="parent" />
</android.support.constraint.ConstraintLayout>
```

Πριν την ενημέρωση του activity_main.xml

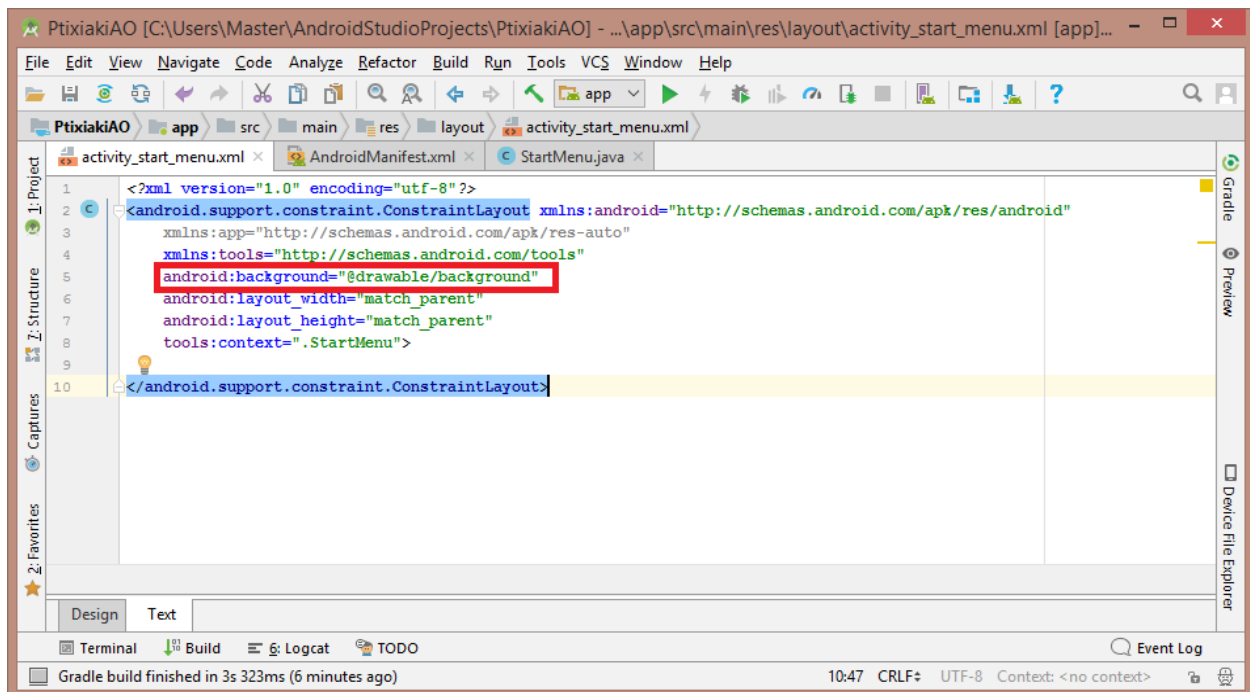
Μετά:



Μετά την ενημέρωση του activity_main.xml

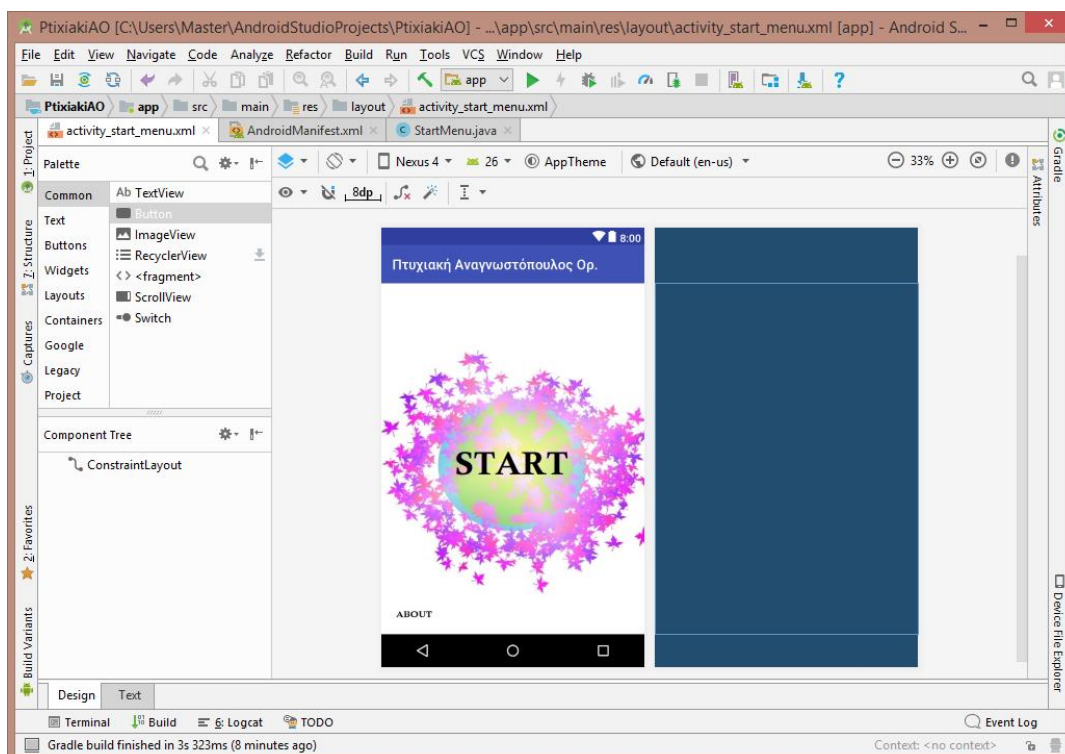
3.2.6. ΔΗΜΙΟΥΡΓΙΑ Start Screen

Για να δημιουργήσουμε την Αρχική οθόνη της εφαρμογής, κάνουμε δεξί κλικ στην εταιρία, New->Activity->Empty Activity, την μετονομάζουμε σε StartMenu και πατάμε Finish. Πάμε στο activity_start_menu.xml και αρχίζουμε να το παραμετροποιούμε. Αρχικά προσθέτουμε στις παραμέτρους του Text tab την εικόνα του background, μέσα από το φάκελο του drawable.



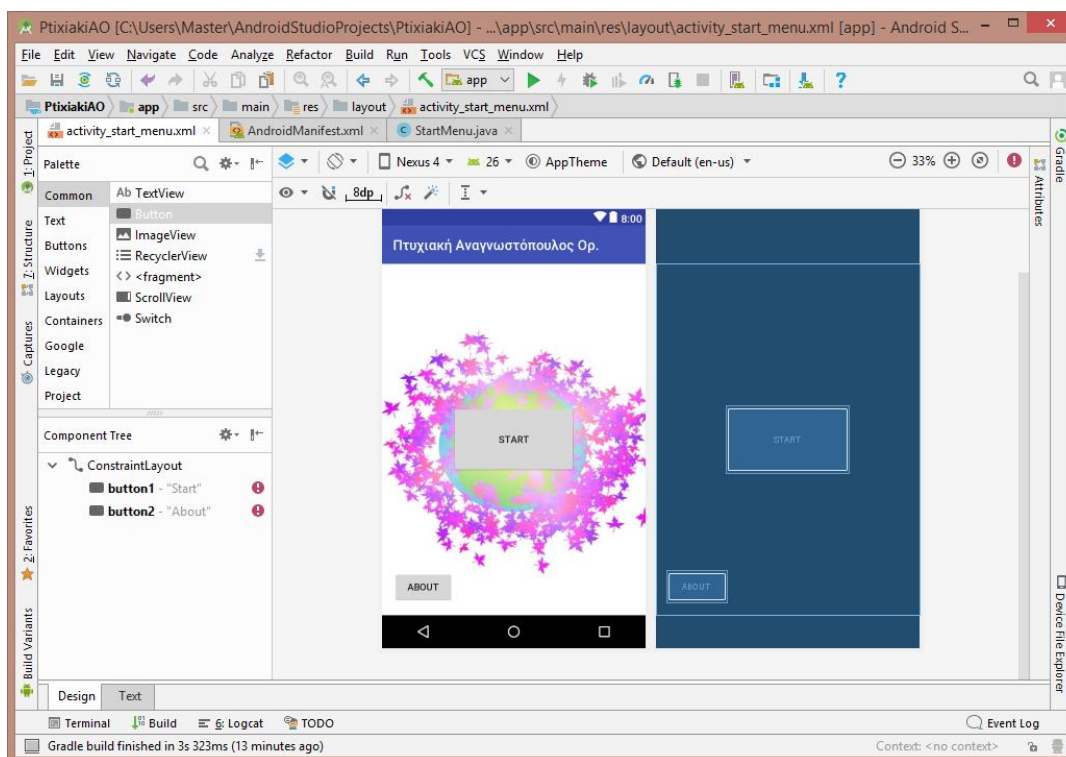
Προσθήκη εικόνας ως background

Με αποτέλεσμα η κενή σελίδα στο design να φαίνεται έτσι:



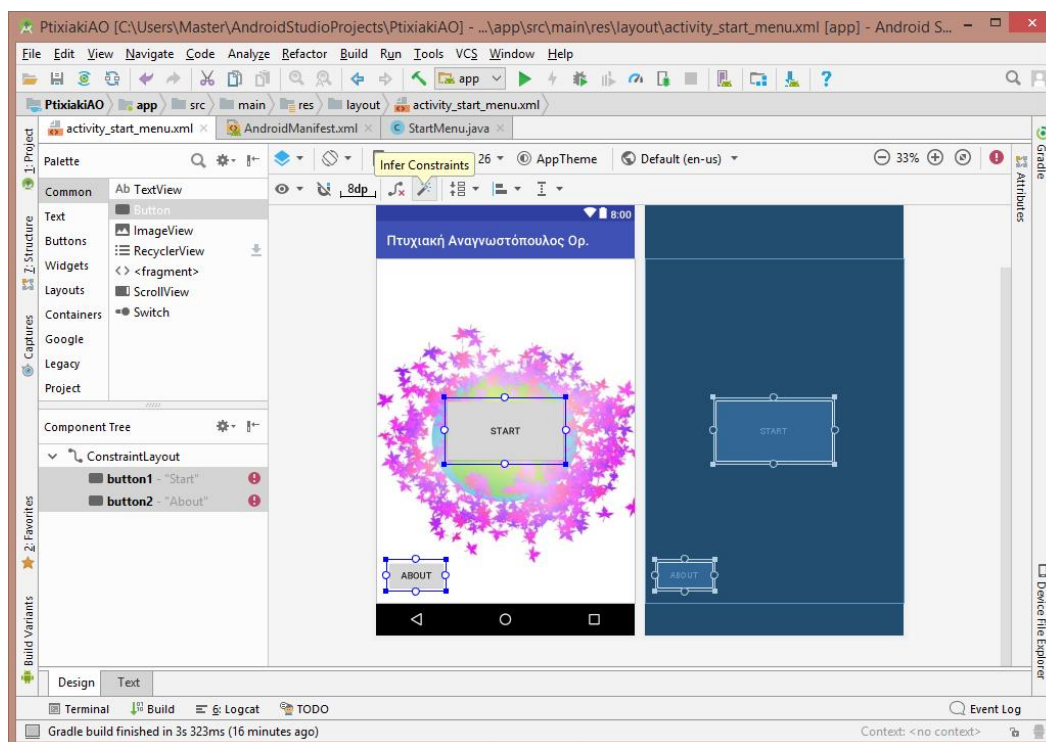
Μετά την προσθήκη background

Συνεχίζουμε κάνοντας Drag & Drop 2 κουμπιά στις θέσεις Start και About, και ορίζουμε το μέγεθός τους και αλλάζοντας το κείμενό τους σε Start και About.



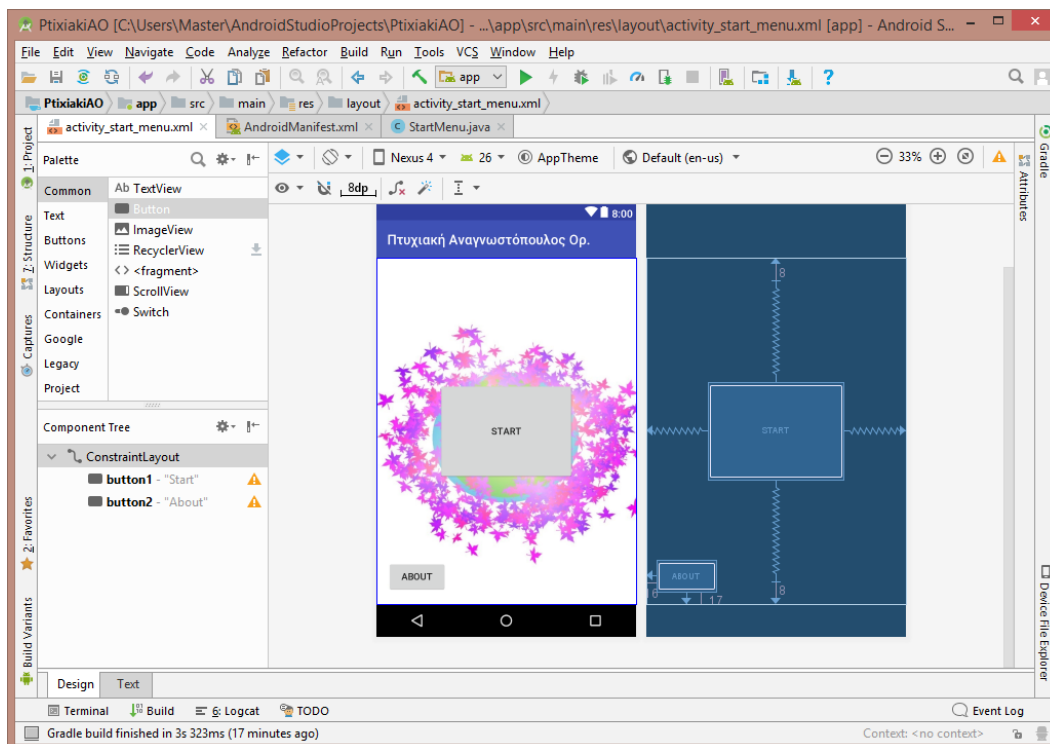
Προσθήκη κουμπιών

Επιλέγουμε και τα 2 κουμπιά και χρησιμοποιούμε το εργαλείο αυτόματης σύνδεσης, έτσι ώστε οτιδήποτε μέγεθος και να έχει η συσκευή, να είναι λειτουργικά και να βρίσκονται στο ίδιο σημείο.



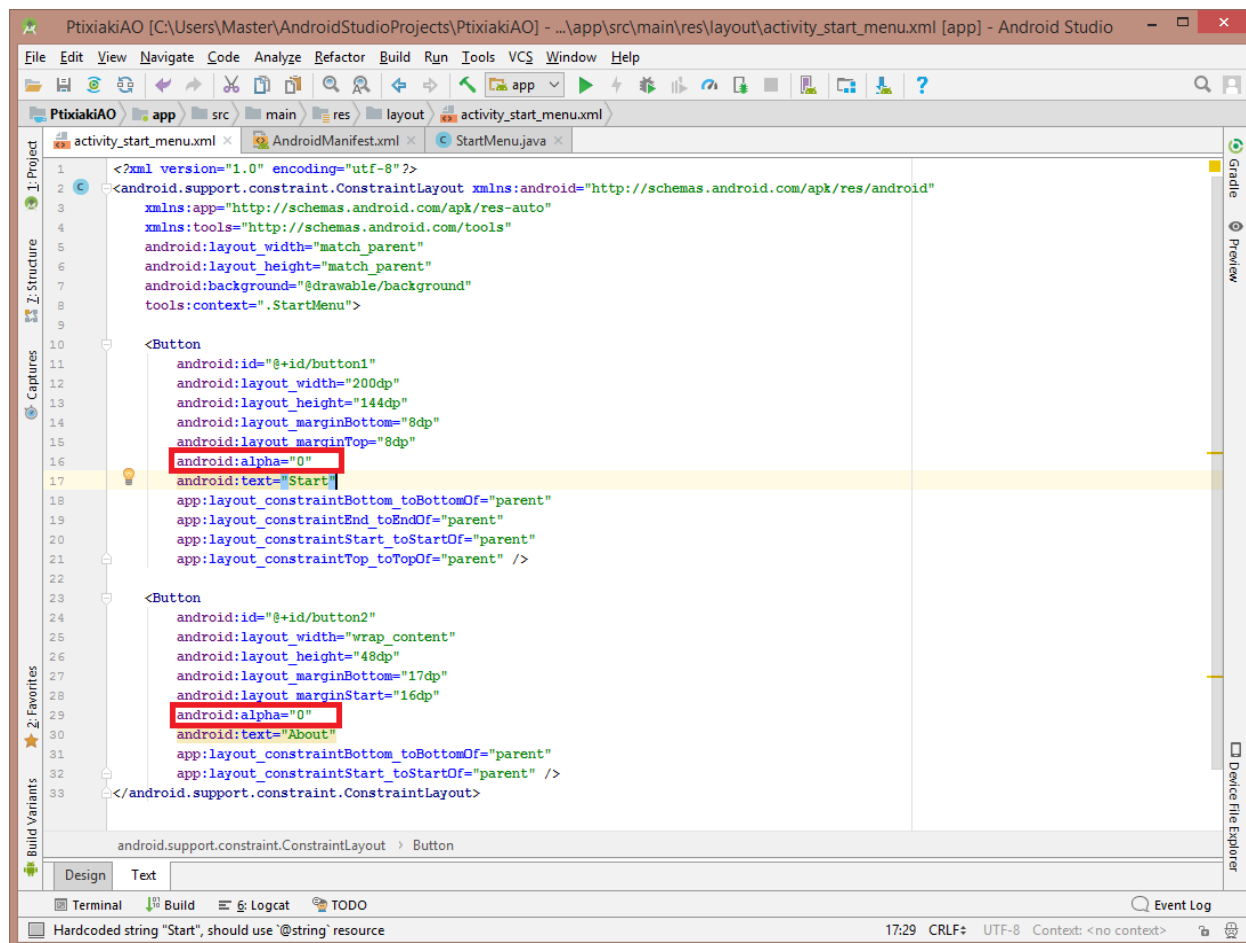
Επιλογή κουμπιών και δημιουργία συνδέσεων/περιορισμών

Θα πρέπει να έχουμε ένα τέτοιο αποτέλεσμα:



Μετά τις συνδέσεις

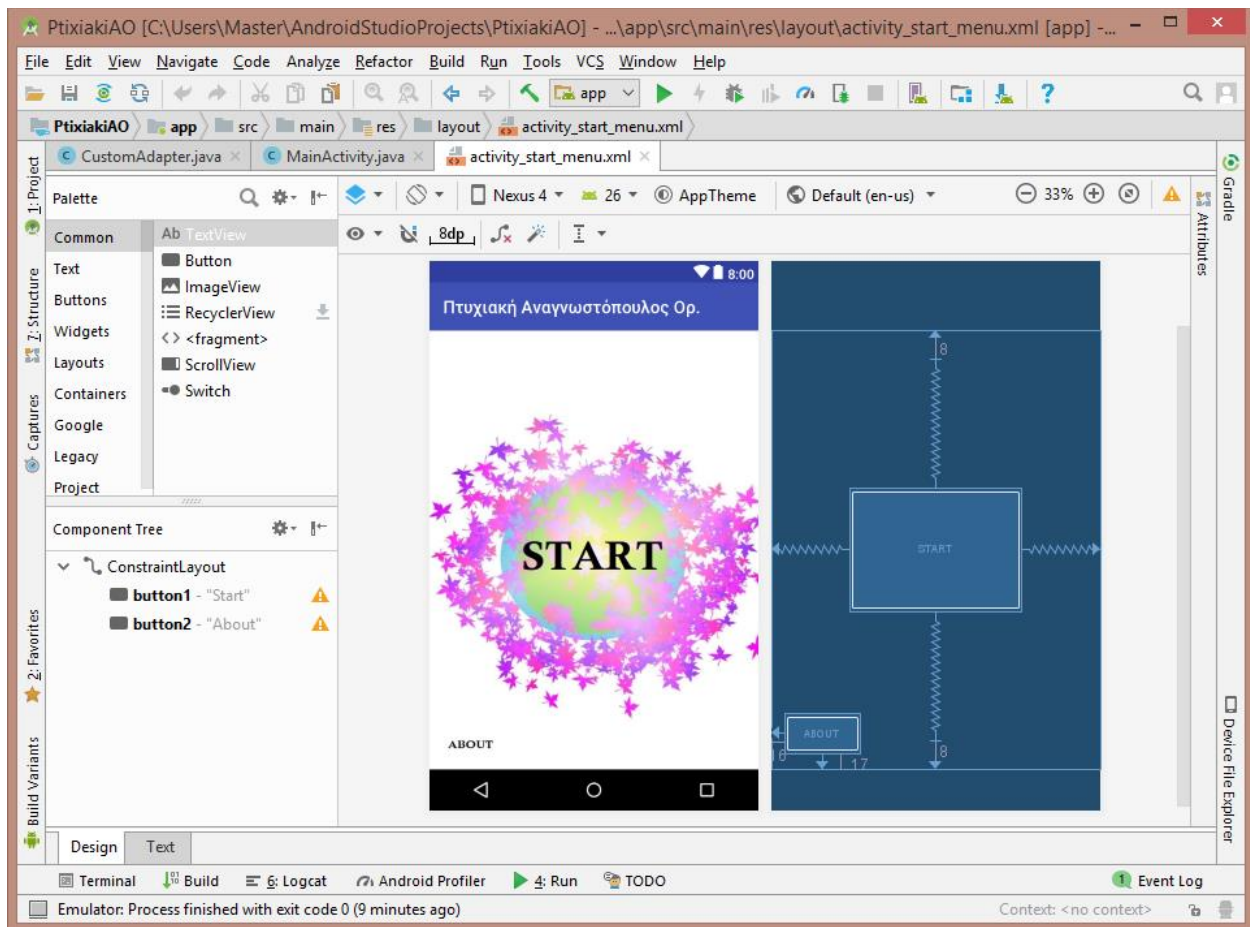
Επειδή όμως θέλουμε τα κουμπιά να είναι αόρατα, αλλά λειτουργικά, προσθέτουμε στο text tab από μια εντολή σε κάθε κουμπί.



```
<?xml version="1.0" encoding="utf-8" ?>
<android.support.constraint.ConstraintLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:background="@drawable/background"
    tools:context=". StartMenu">
    <Button
        android:id="@+id/button1"
        android:layout_width="200dp"
        android:layout_height="144dp"
        android:layout_marginBottom="8dp"
        android:layout_marginTop="8dp"
        android:alpha="0"
        android:text="Start"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent" />
    <Button
        android:id="@+id/button2"
        android:layout_width="wrap_content"
        android:layout_height="48dp"
        android:layout_marginBottom="17dp"
        android:layout_marginStart="16dp"
        android:alpha="0"
        android:text="About"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintStart_toStartOf="parent" />
</android.support.constraint.ConstraintLayout>
```

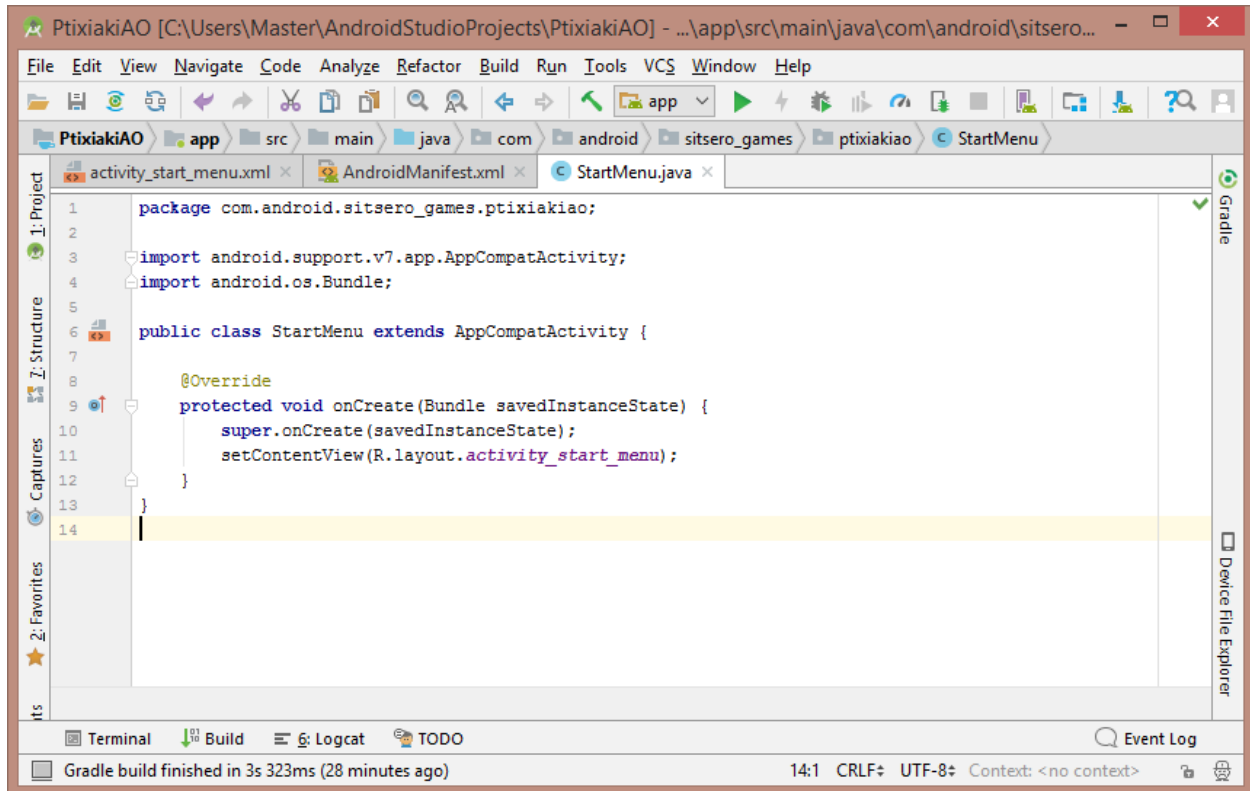
Προσθήκη εντολής για να κάνουμε τα κουμπιά αόρατα

Το αποτέλεσμα μετά την μετατροπή των κουμπιών σε διαφανή είναι το εξής:



Μετά τη διαφάνεια των κουμπιών

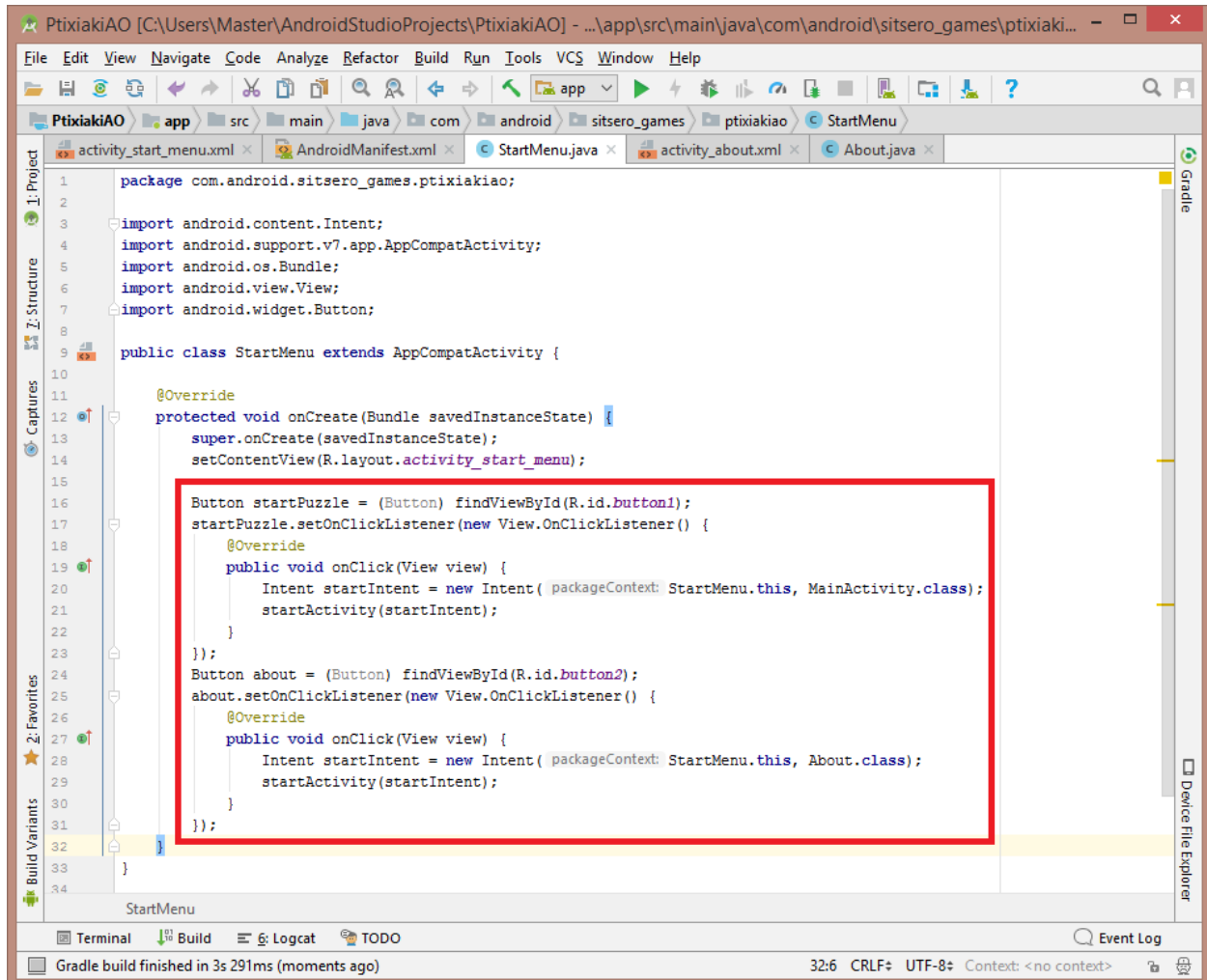
Περνώντας στο StartMenu.java, έχουμε μια άδεια κλάση. Σε αυτή θέλουμε να προσθέσουμε το άνοιγμα δύο νέων Activities, το οποίο γίνεται με τη χρήση της κλάσης Intent. Άρα, πριν έχουμε:



```
1 package com.android.sitsero_games.ptixiakiao;
2
3 import android.support.v7.app.AppCompatActivity;
4 import android.os.Bundle;
5
6 public class StartMenu extends AppCompatActivity {
7
8     @Override
9     protected void onCreate(Bundle savedInstanceState) {
10         super.onCreate(savedInstanceState);
11         setContentView(R.layout.activity_start_menu);
12     }
13 }
14
```

Αρχική κλάση StartMenu.java

Χρησιμοποιώντας τις id των κουμπιών που μόλις δημιουργήσαμε, συνδέουμε τα δύο κουμπιά με τα ανάλογα ids τους και δημιουργούμε από μια τυπική onClickListener σε κάθε κουμπί, μέσω της οποίας με τα ανάλογα Intents, θα ανοίγεται και η αντίστοιχη Activity. Ο κώδικας μετά τις συγκεκριμένες προσθήκες και τα αυτόματα imports γίνεται ως εξής:

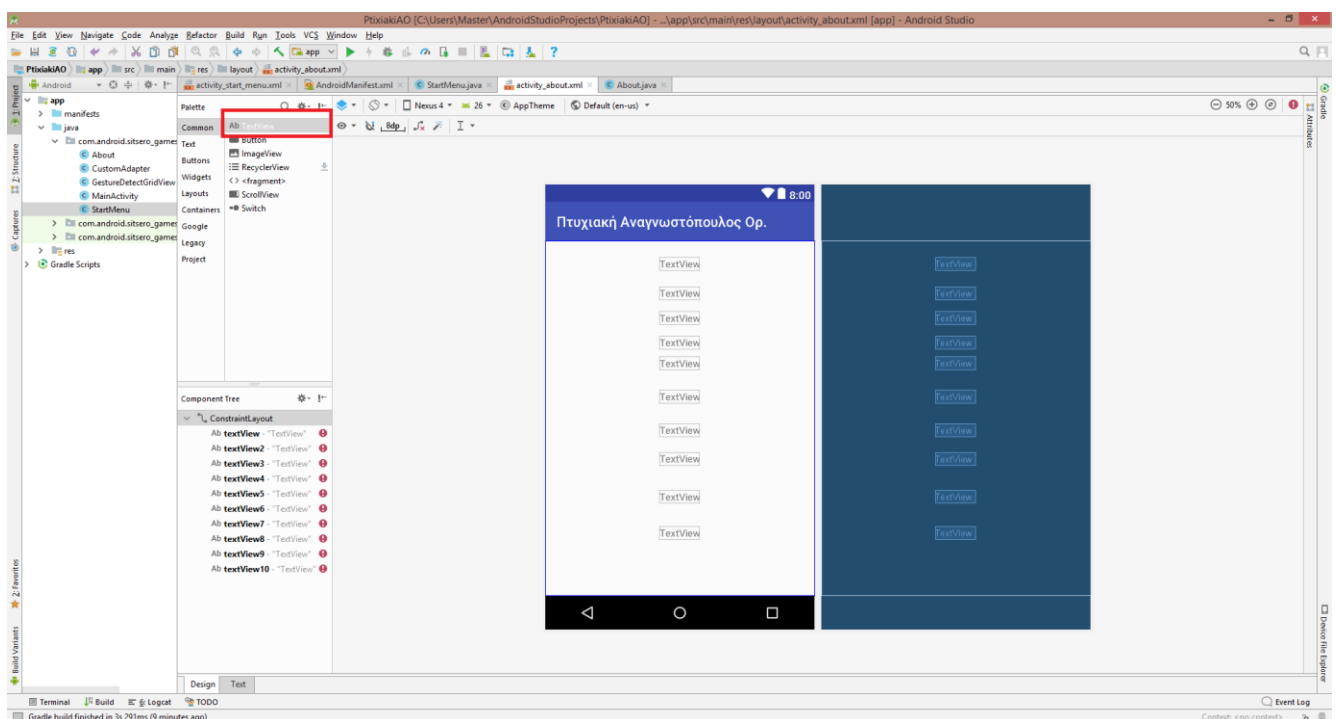


```
1 package com.android.sitsero_games.ptixiakiao;
2
3 import android.content.Intent;
4 import android.support.v7.app.AppCompatActivity;
5 import android.os.Bundle;
6 import android.view.View;
7 import android.widget.Button;
8
9 public class StartMenu extends AppCompatActivity {
10
11     @Override
12     protected void onCreate(Bundle savedInstanceState) {
13         super.onCreate(savedInstanceState);
14         setContentView(R.layout.activity_start_menu);
15
16         Button startPuzzle = (Button) findViewById(R.id.button1);
17         startPuzzle.setOnClickListener(new View.OnClickListener() {
18             @Override
19             public void onClick(View view) {
20                 Intent startIntent = new Intent( packageContext: StartMenu.this, MainActivity.class);
21                 startActivity(startIntent);
22             }
23         });
24         Button about = (Button) findViewById(R.id.button2);
25         about.setOnClickListener(new View.OnClickListener() {
26             @Override
27             public void onClick(View view) {
28                 Intent startIntent = new Intent( packageContext: StartMenu.this, About.class);
29                 startActivity(startIntent);
30             }
31         });
32     }
33 }
34
```

Προσθήκη προθέσεων ανοίγματος άλλων activities με τα πατήματα των δυο κουμπιών

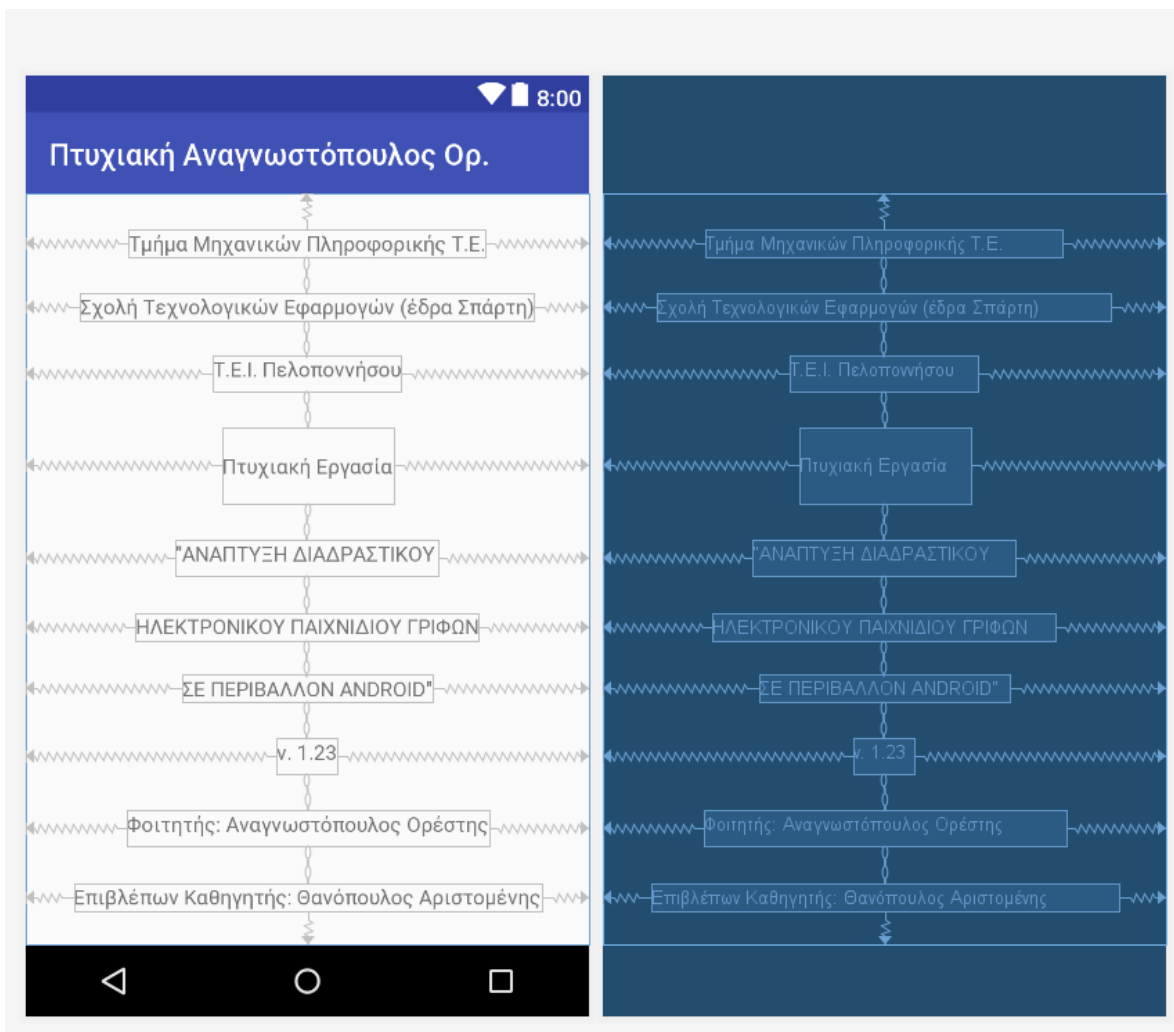
3.2.7. ΔΗΜΙΟΥΡΓΙΑ About

Για να δημιουργήσουμε την Αρχική οθόνη της εφαρμογής, κάνουμε δεξί κλικ στην εταιρία, New->Activity->Empty Activity, την μετονομάζουμε σε About και πατάμε Finish. Η συγκεκριμένη Activity περιέχει πληροφορίες σχετικά με την εφαρμογή. Δε θα χρειαστεί να γραφεί κώδικας στο Java αρχείο, παρά μόνο να φτιάξουμε ορισμένα TextViews στο xml αρχείο. Ανοίγουμε λοιπόν το activity_about.xml αρχείο στο design tab και αρχίζουμε να προσθέτουμε αρκετά TextViews, στην περίπτωση μας χρειαστήκαμε 10.



Αρχικό drag & drop TextViews

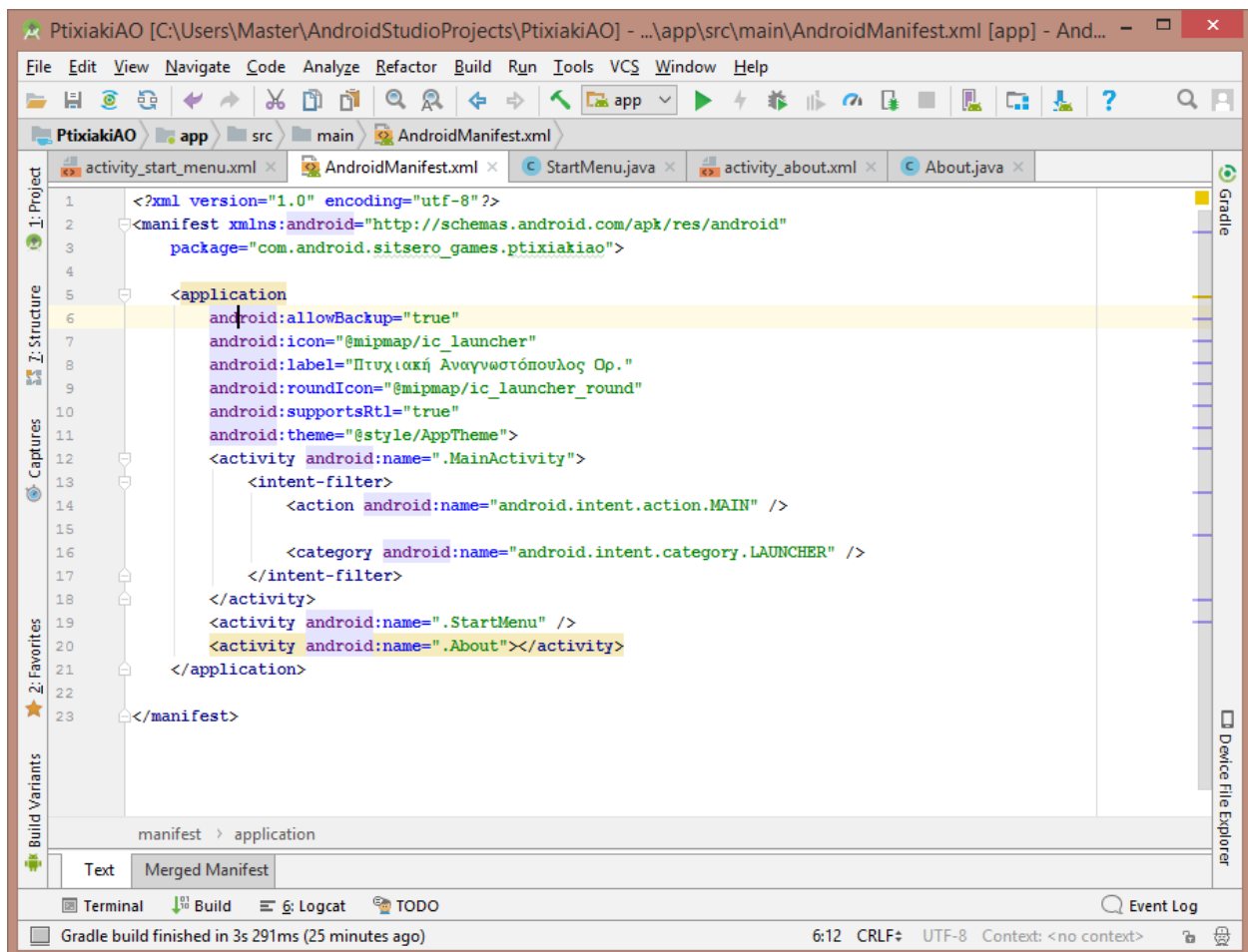
Στη συνέχεια κάνουμε constraint και edit κάθε view. Μόλις τελειώσουμε, πρέπει να φαίνεται κάπως έτσι:



Παραμετροποίηση κειμένων και ορισμός συνδέσεων μεταξύ των Views

3.2.8. ΕΠΕΞΕΡΓΑΣΙΑ AndroidManifest.xml

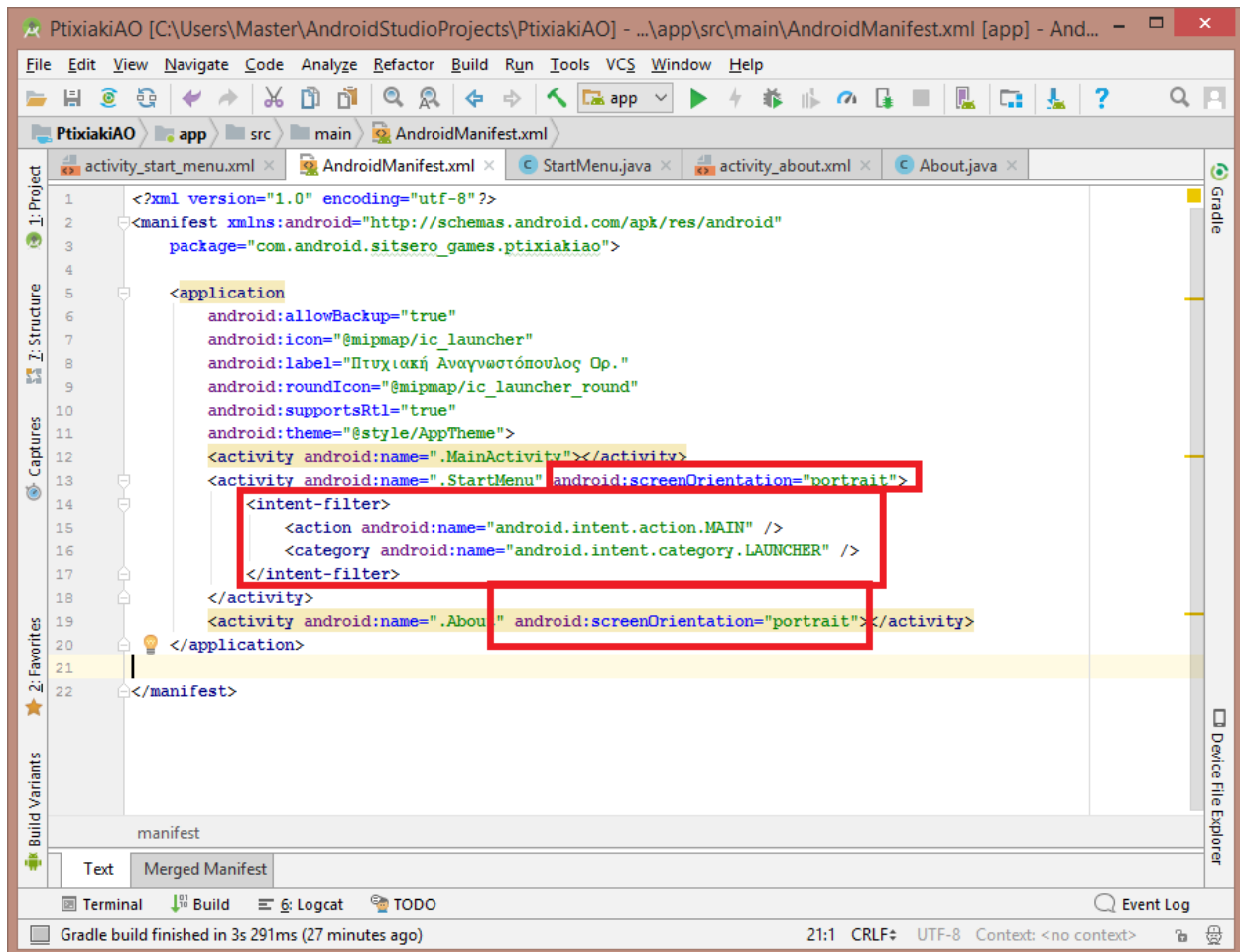
Εφόσον τελειώσαμε με τα Activities που θα χρειαστούμε, ήρθε η ώρα να συμμαζέψουμε και το AndroidManifest.xml. Θα χρειαστεί να ορίσουμε ως αρχική Activity την StartMenu, διότι αυτή τη στιγμή είναι η MainActivity, καθώς με αυτή ήταν η πρώτη Activity που φτιάξαμε, αλλά και να ορίσουμε περιορισμούς orientation σε portrait στην Αρχική Οθόνη και στην σελίδα About, ακόμη και στην ίδια την MainActivity εάν δε θέλουμε να γυρίζει το παζλ εάν γυρίσουμε τη συσκευή μας. Το AndroidManifest.xml προ περιορισμών και ρυθμίσεων:



```
1 <?xml version="1.0" encoding="utf-8"?>
2 <manifest xmlns:android="http://schemas.android.com/apk/res/android"
3     package="com.android.sitsero_games.ptixiakiao">
4
5     <application
6         android:allowBackup="true"
7         android:icon="@mipmap/ic_launcher"
8         android:label="Πτυχιακή Αναγνωστικόπουλος Op."
9         android:roundIcon="@mipmap/ic_launcher_round"
10        android:supportsRtl="true"
11        android:theme="@style/AppTheme">
12        <activity android:name=".MainActivity">
13            <intent-filter>
14                <action android:name="android.intent.action.MAIN" />
15
16                <category android:name="android.intent.category.LAUNCHER" />
17            </intent-filter>
18        </activity>
19        <activity android:name=".StartMenu" />
20        <activity android:name=".About"></activity>
21    </application>
22
23 </manifest>
```

AndroidManifest.xml προ επεξεργασίας

Για να αλλάξουμε την αρχική Activity, μεταφέραμε τα φίλτρα της MainActivity στην StartMenu. Επιπλέον, μπήκαν οι περιορισμοί πορτρέτου στις StartMenu και About. Το AndroidManifest.xml μετά τις αλλαγές:



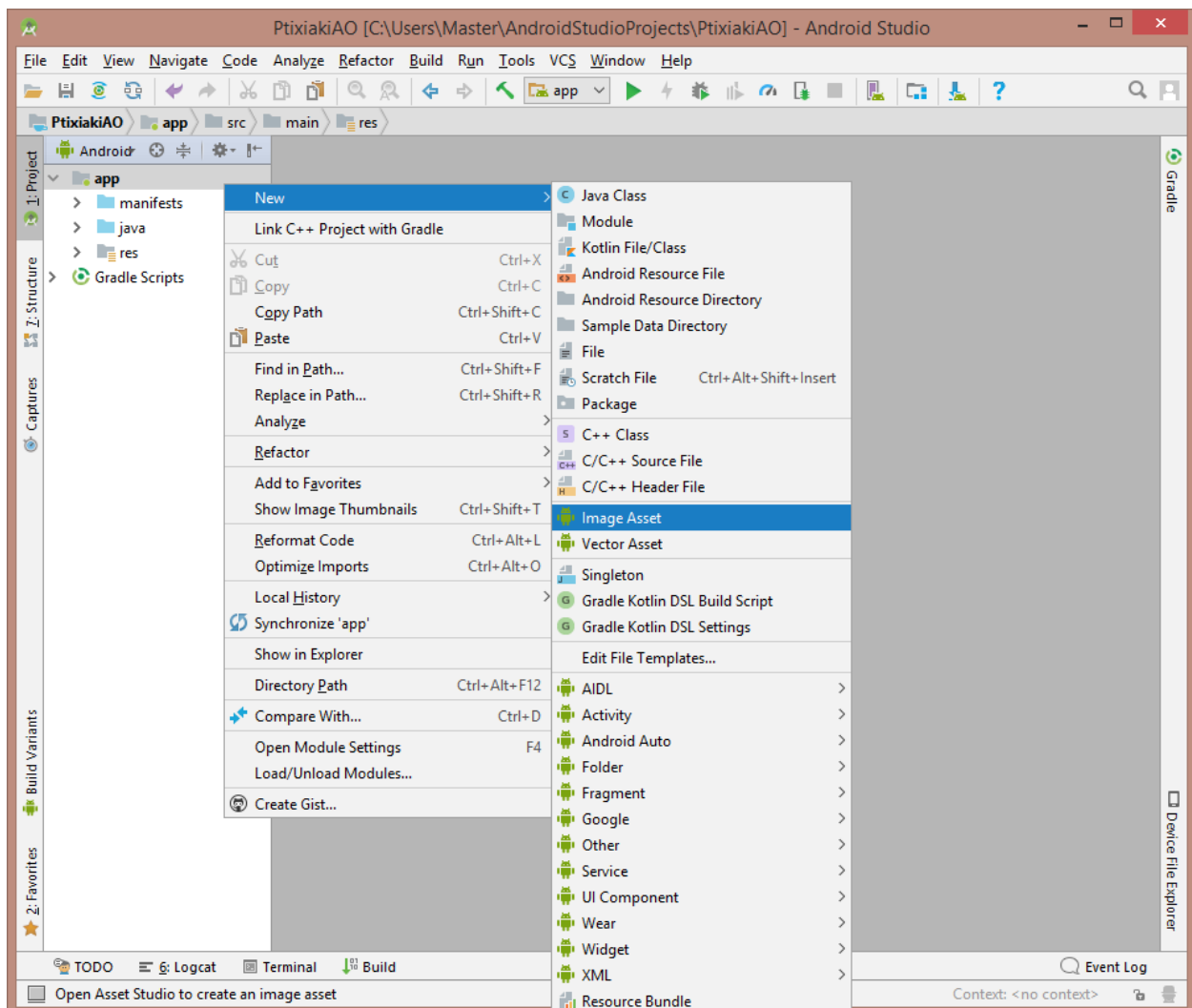
```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.android.sitsero_games.ptixiakiao">
    <application
        android:allowBackup="true"
        android:icon="@mipmap/ic_launcher"
        android:label="Πτυχιακή Αναγνωστικόπουλος Op."
        android:roundIcon="@mipmap/ic_launcher_round"
        android:supportsRtl="true"
        android:theme="@style/AppTheme">
        <activity android:name=".MainActivity"></activity>
        <activity android:name=".StartMenu" android:screenOrientation="portrait">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />
                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
        <activity android:name=".About" android:screenOrientation="portrait"></activity>
    </application>
</manifest>
```

AndroidManifest.xml μετά επεξεργασία

3.2.9. ΕΙΣΑΓΩΓΗ Icon ΕΦΑΡΜΟΓΗΣ

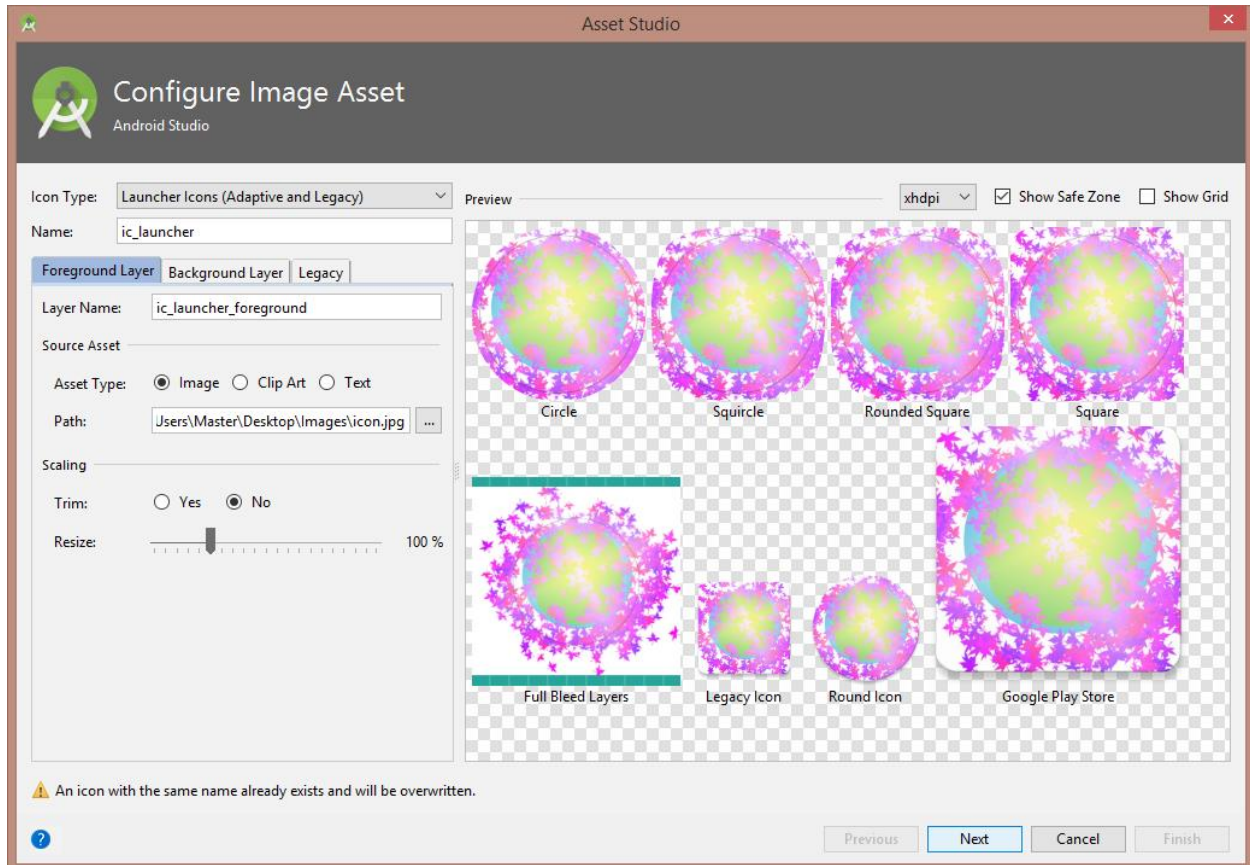
Για να βάλουμε δικό μας εικονίδιο στην εφαρμογή, κάνουμε τις εξής ενέργειες:

Δεξί κλικ σε app->New->Image Asset.



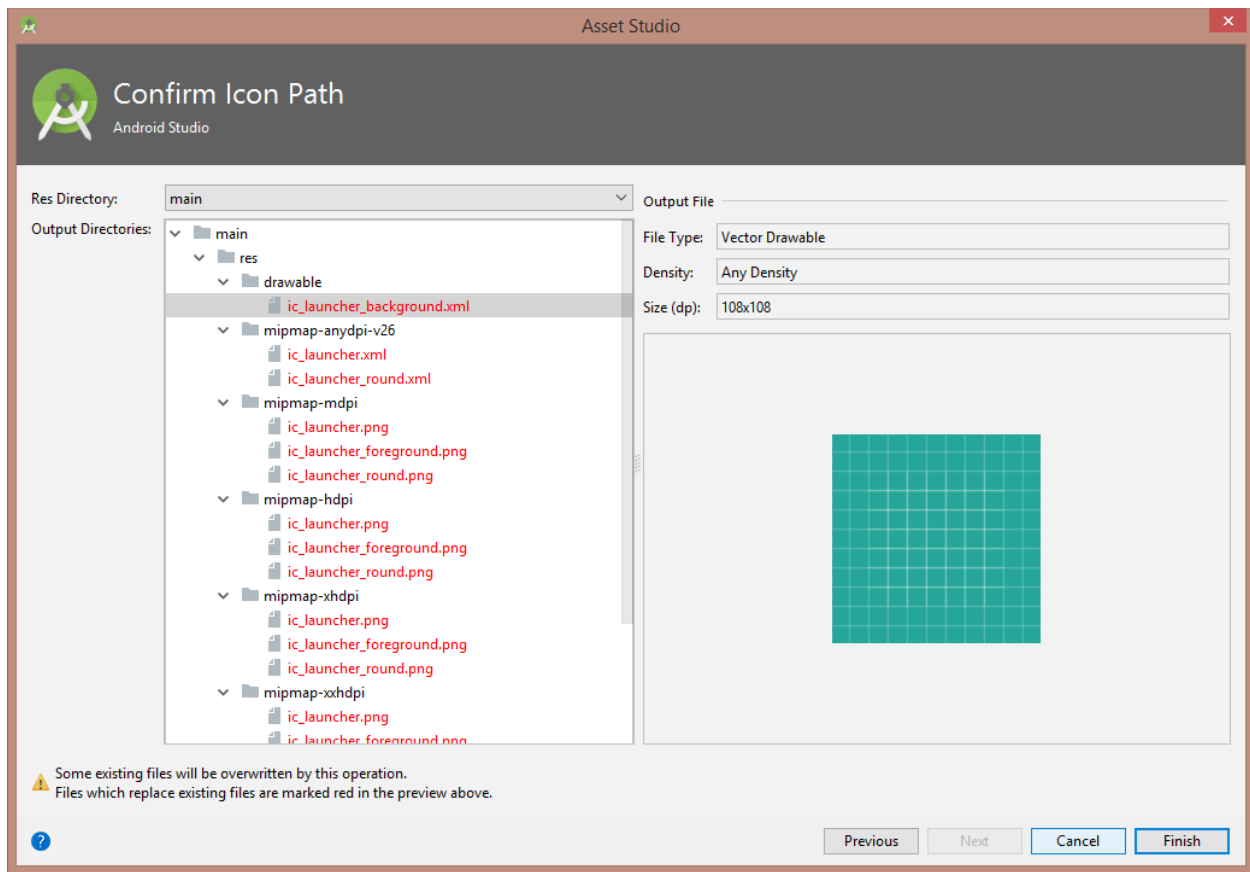
Εισαγωγή νέου Image Asset

Επιλέγουμε ως Asset type το Image, καθορίζουμε το path της εικόνας που θέλουμε να χρησιμοποιήσουμε ως εικονίδιο, και κάνουμε τις ανάλογες μικρορυθμίσεις, εάν θέλουμε. Πατάμε Next μόλις τελειώσουμε.



Επιλογή Icon

Στην επόμενη οθόνη πατάμε Finish και είμαστε έτοιμοι.

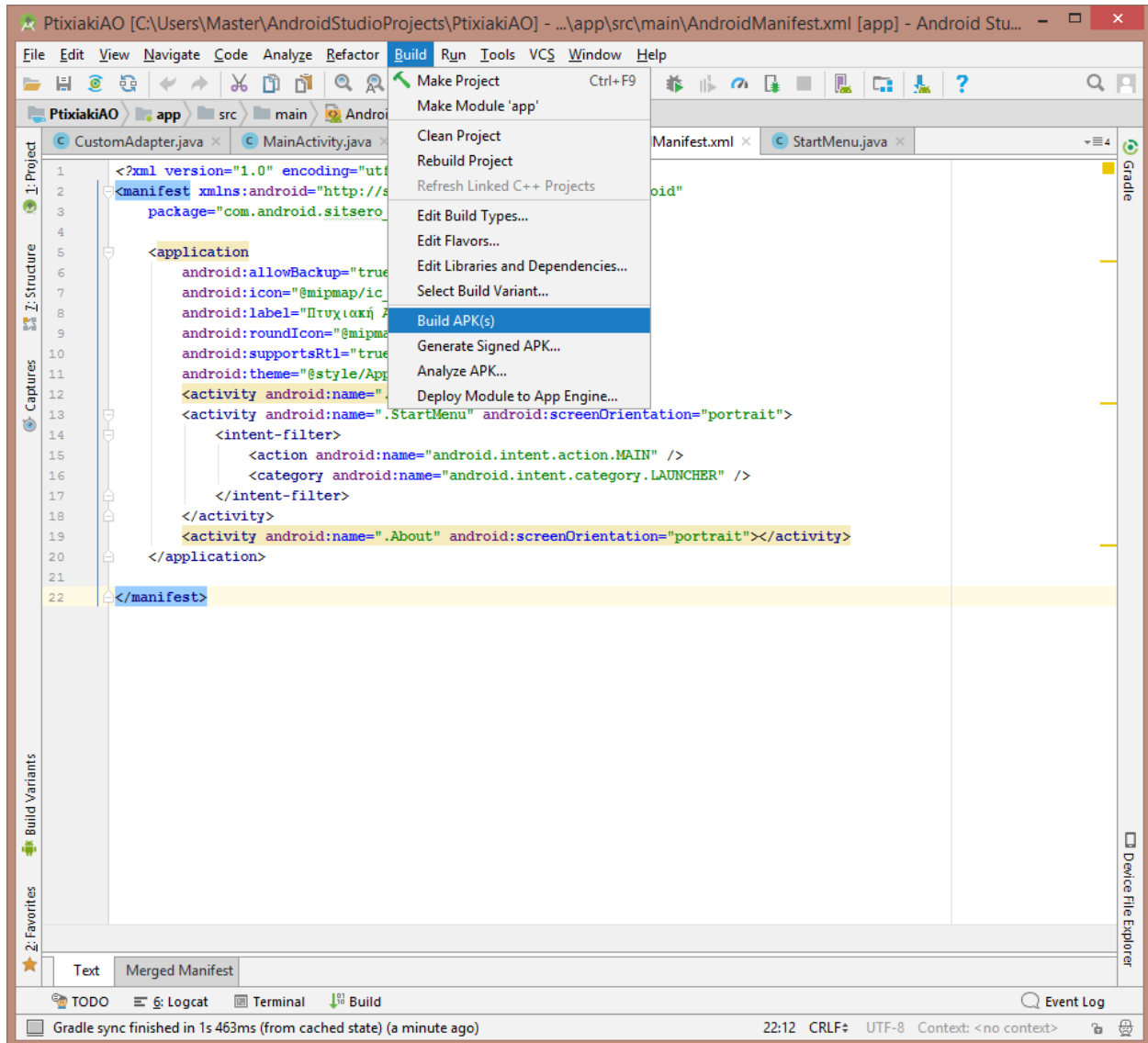


Τελική οθόνη ενημέρωσης των αλλαγών

3.3. ΕΚΤΕΛΕΣΗ-ΔΟΚΙΜΕΣ-ΑΛΛΑΓΕΣ

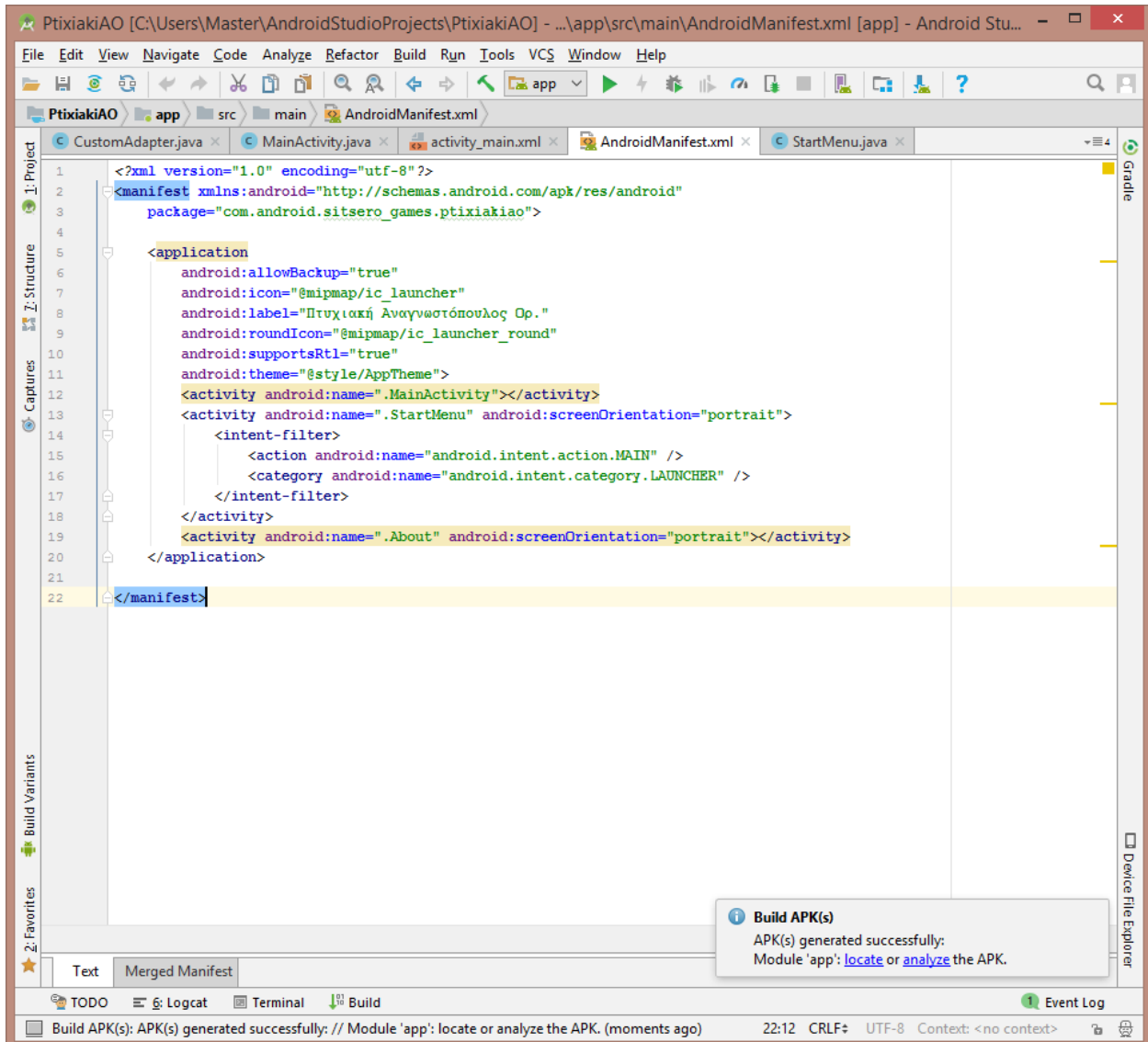
Εφόσον τελειώσαμε με την υλοποίηση του κώδικα της εργασίας, θα χρειαστεί να κάνουμε build ένα apk, δηλαδή ένα πακέτο εγκατάστασης της εφαρμογής μας.

Για να το καταφέρουμε αυτό, πατάμε Build->Build APK(s)



APK Build

Το Android Studio θα μας ενημερώσει για τυχόν σφάλματα κατά το compiling, τα οποία και θα εμφανίζονται στο tab Logcat. Εάν πάνε όλα καλά, όπως στην περίπτωση μας, μετά από λίγα δευτερόλεπτα από το ξεκίνημα της διαδικασίας χτισίματος της εφαρμογής, θα ενημερωθούμε ότι το χτίσιμο ήταν επιτυχές μέσω ενός prompt, το οποίο θα έχει τις επιλογές ανάλυσης ή τοποθεσίας του αρκ μας. Πατάμε στο locate, για να βρούμε γρήγορα την τοποθεσία του, το μετονομάζουμε προαιρετικά (έχει ως default name το "app-debug.apk") και το μεταφέρουμε στη συσκευή μας για εγκατάσταση.



Prompt εύρεσης τοποθεσίας ή ανάλυσης

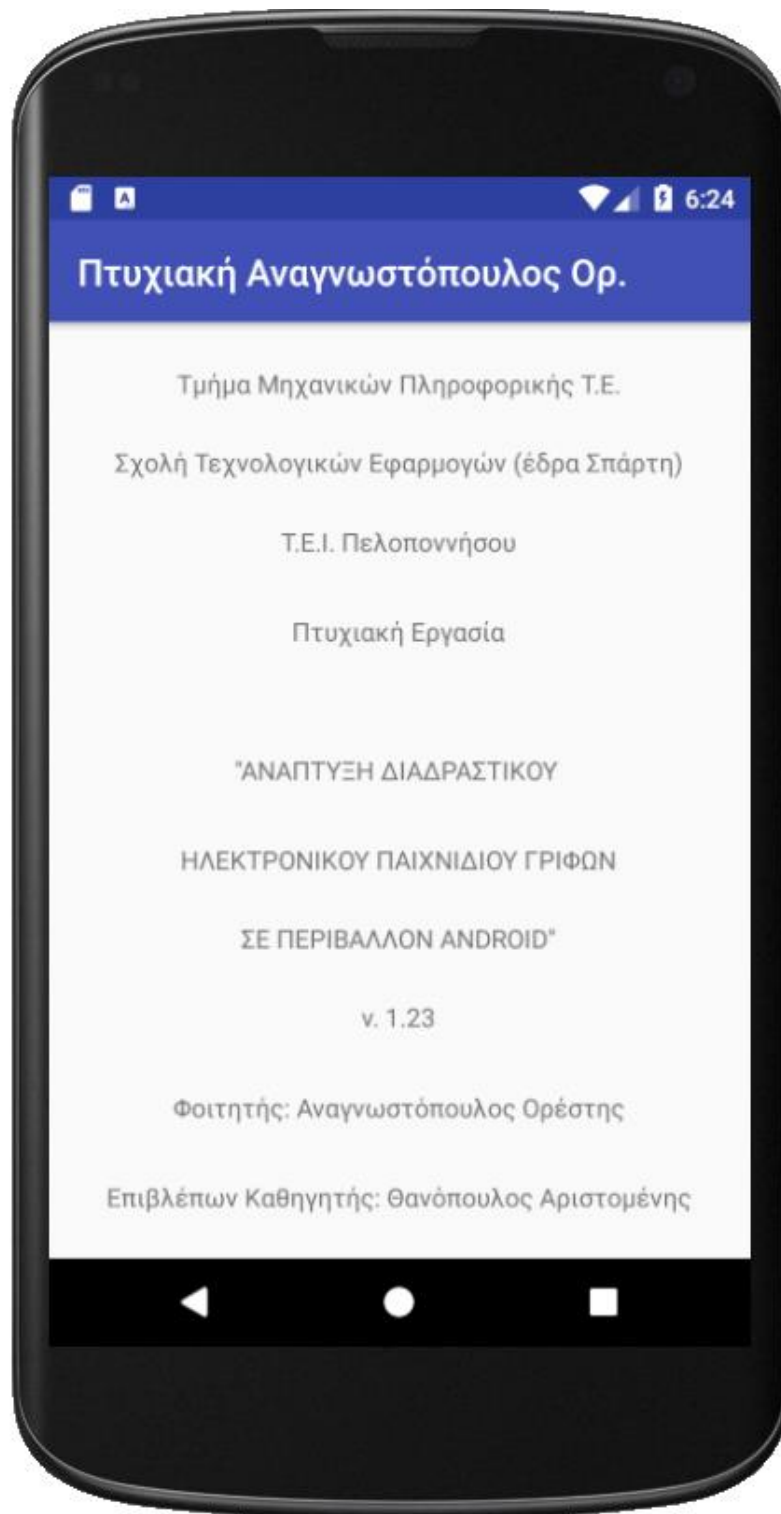
Για την παρουσίαση του demo, για την ευκολία της συγγραφής της Πτυχιακής Εργασίας, τα παρακάτω παραδείγματα είναι από τον ενσωματωμένο προσομοιωτή του Android Studio.

Ανοίγοντας την εφαρμογή, εισερχόμαστε στην StartMenu, η οποία έχει ως τα δύο κουμπιά, Start & About.



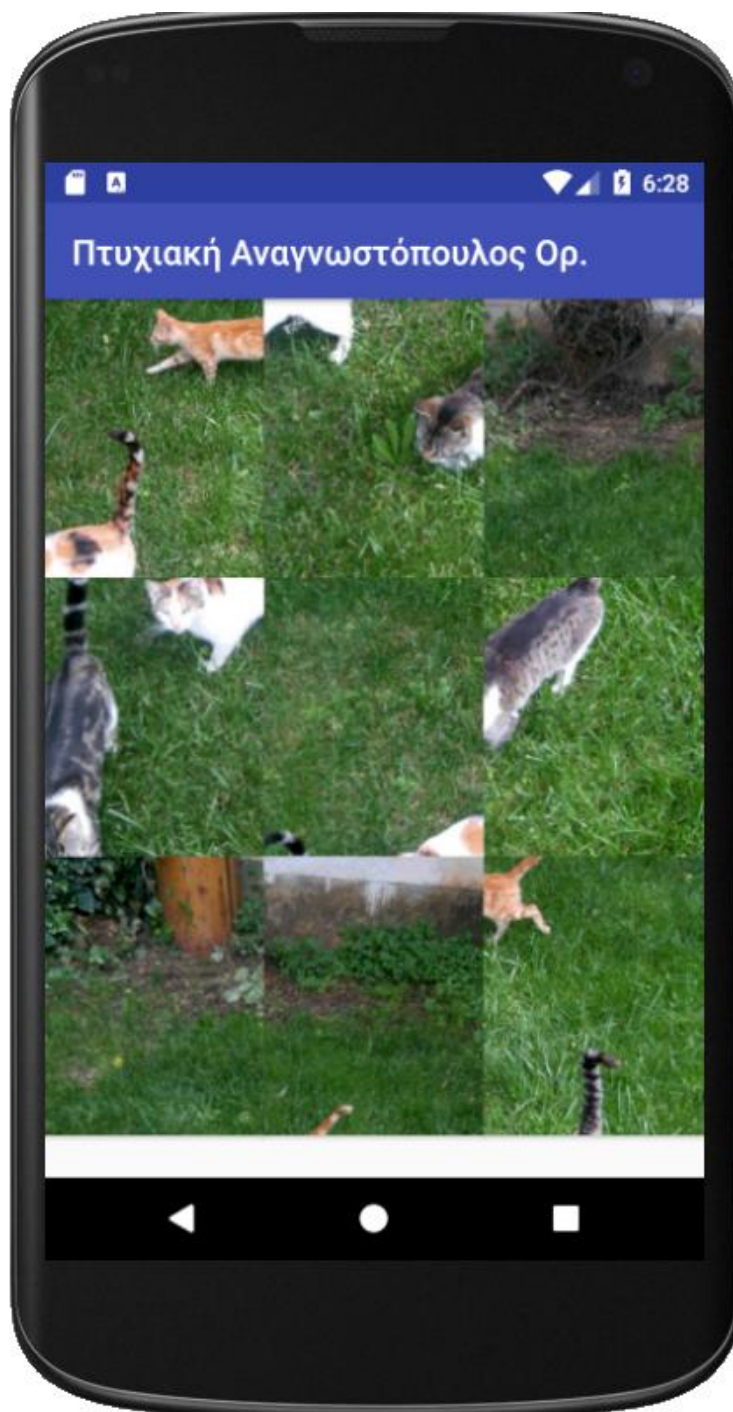
Αρχική οθόνη

Κάνοντας κλικ στο About, εμφανίζεται η οθόνη με τις πληροφορίες της εφαρμογής, τις οποίες είχαμε εμείς ορίσει στα TextViews.



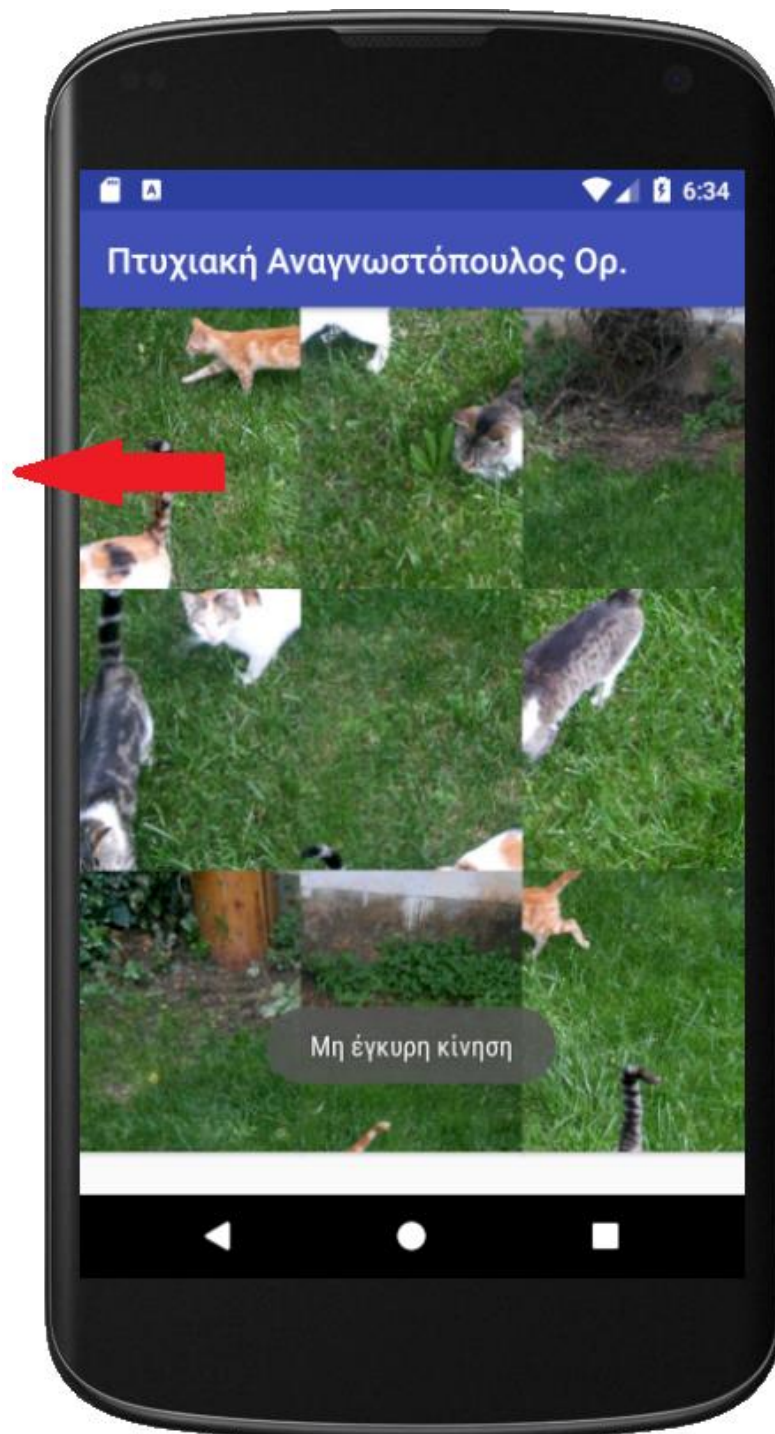
Σελίδα About

Πατάμε το Πίσω της συσκευής μας για να εξέλθουμε από αυτό το μενού. Κάνοντάς το αυτό, επιστρέφουμε στο αρχικό μενού. Στη συνέχεια πατάμε Start. Βλέπουμε ότι εισερχόμαστε σε ένα άλλο μενού, αυτό μιας ανακατεμένης εικόνας.



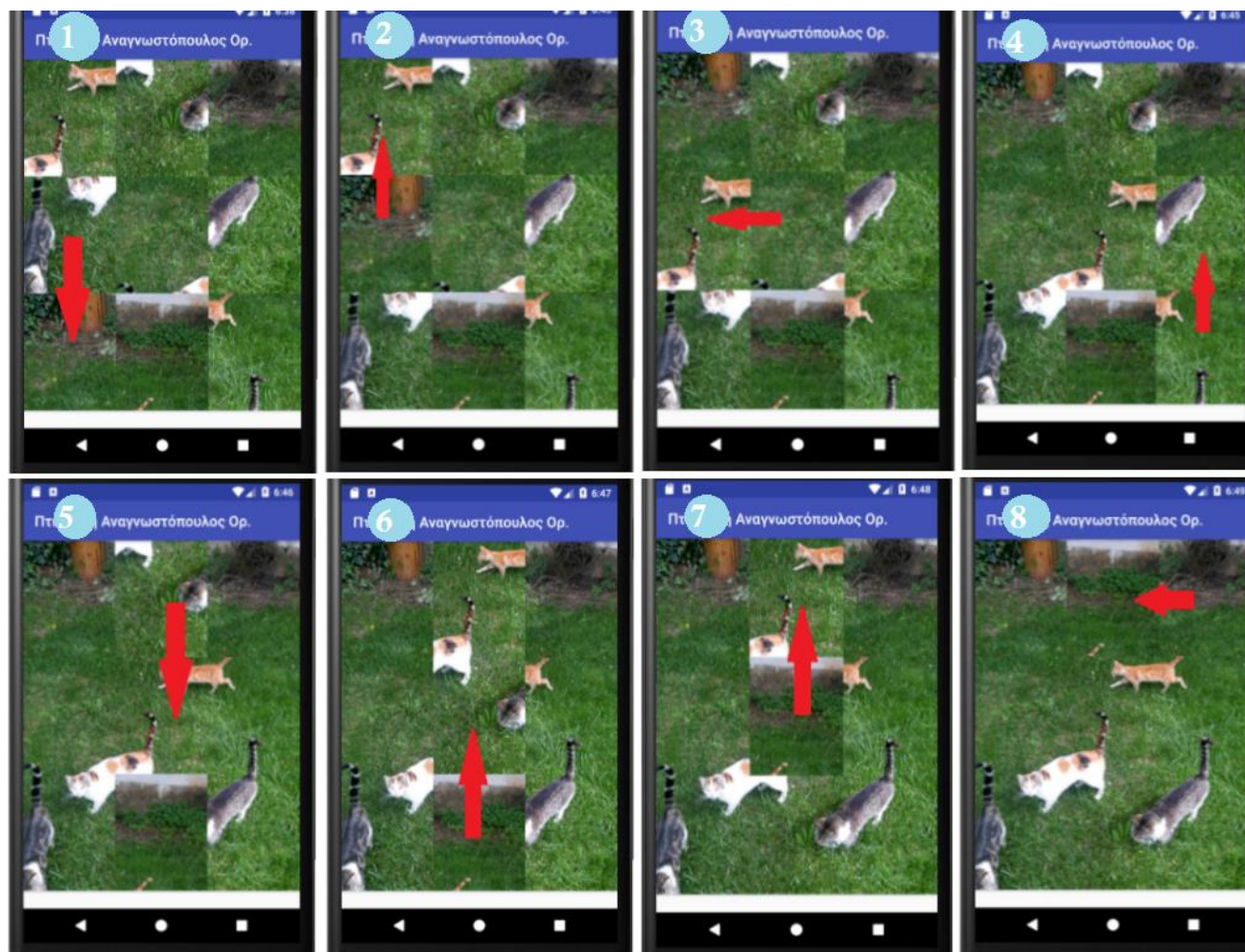
Εισαγωγή σε σελίδα Puzzle

Παρακάτω θα παρουσιαστεί μια επιτηδευμένη λανθασμένη κίνηση, για να φανεί το μήνυμα λάθους κίνησης. Θα προσπαθήσουμε να κουνήσουμε το πάνω αριστερά πλακίδιο εκτός κάδρου.



Μήνυμα λάθους

Πλέον, μπορούμε να δοκιμάσουμε να το λύσουμε. Παρακάτω παρουσιάζεται βήμα-βήμα η επίλυσή του. Ας ξεκινήσουμε!



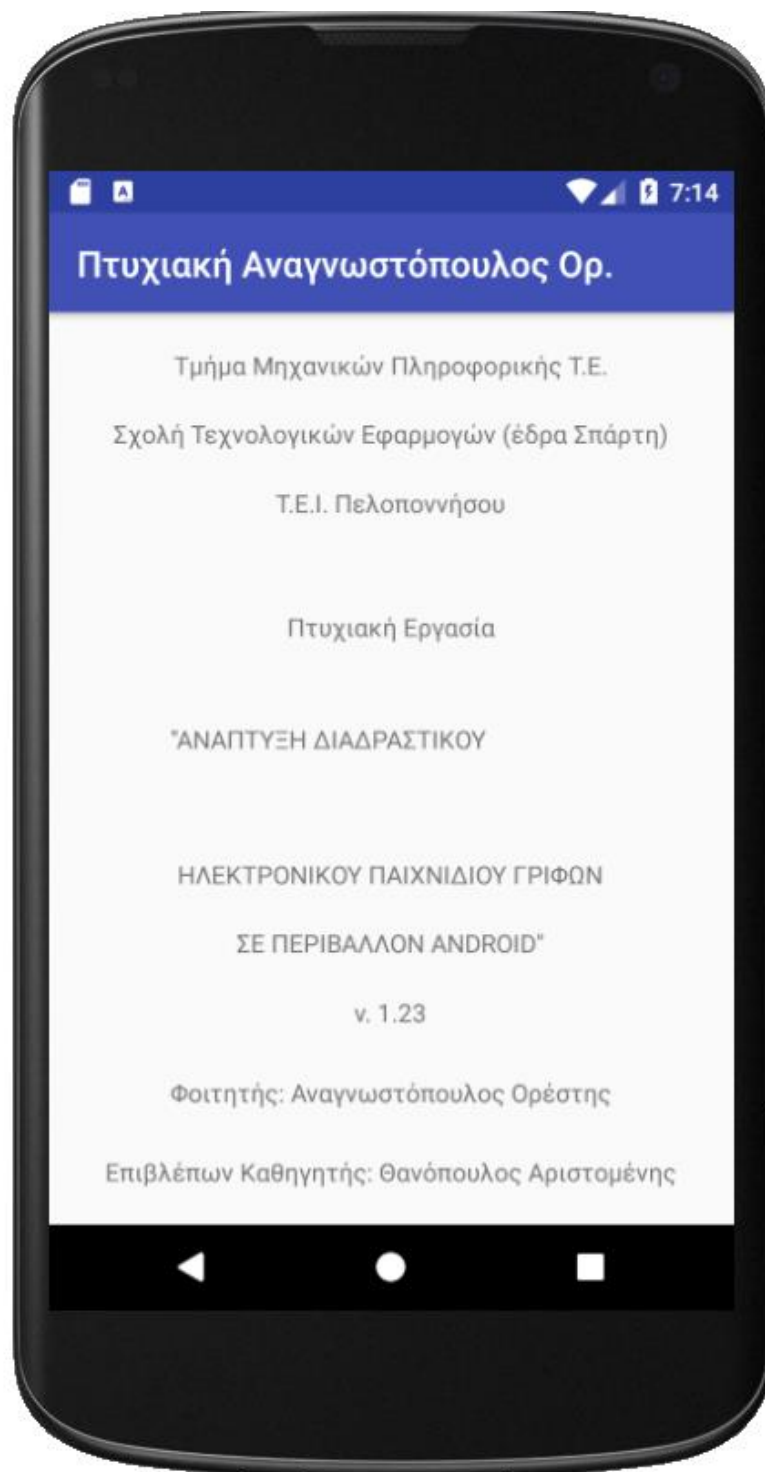
Επίλυση puzzle

Μετά και την τελευταία μας κίνηση, θα εμφανιστεί το μήνυμα συγχαρητηρίων για την επίλυση του παζλ.



Μήνυμα συγχαρητηρίων

Οι κύριες διορθώσεις που έγιναν, μετά από πολλές αλλαγές trial & error, είναι κυρίως στα TextViews της σελίδας About. Τα κείμενα πολλές φορές εμφανίζονταν στραβά ή ημιτελή, όπως παρακάτω.



Λανθασμένη στοίχιση TextView

Ένα άλλο λάθος που υπήρχε ήταν ότι οι σελίδες Αρχική και About μπορούσαν να αλλάξουν orientation, κάτι το οποίο φαινόταν άσχημο στο μάτι, οπότε και κλειδώσαμε αυτές τις 2 σελίδες σε portrait orientation, όπως φαίνεται και από το AndroidManifest.xml. Παράδειγμα βλέπουμε εδώ:



Πλαγιασμένη Αρχική Οθόνη

4. ΣΥΜΠΕΡΑΣΜΑΤΑ

Η ανάπτυξη της συγκεκριμένης εφαρμογής χρειάστηκε ευρεία γκάμα γνώσεων Java, οπότε και ακόνισε τις γνώσεις μου, αλλά και προσέθεσε σε αυτές. Η συγκεκριμένη εφαρμογή θα μπορούσε να είχε φτιαχτεί με κάποιο άλλο IDE(Integrated development Environment). Προτιμήθηκε το Android Studio λόγω της ευκολίας χρήσης, των ποικίλων tutorials στο development site του, καθώς και προηγούμενης προσωπικής εξοικείωσης με αυτό.

Ξεκινώντας, εγκαταστήσαμε το Android Studio καθώς και διασπάσαμε την εικόνα σε μικρότερα ίσα κομμάτια, έτσι ώστε να τα χρησιμοποιήσουμε σαν κομμάτια παζλ. Στη συνέχεια, με τη βοήθεια από το Android Studio developer's site, αλλά και από μερικά tutorials, υλοποιήσαμε την ιδέα σε εφαρμογή, η οποία ήταν μέσα στις προδιαγραφές που είχαμε αρχικά σκεφτεί, αντιμετωπίζοντας ό,τι πρόβλημα βρέθηκε μπροστά μας με επιτυχία.

Πιθανές επεκτάσεις τις οποίες θα μπορούσε να λάβει η εφαρμογή είναι παραπάνω/δυσκολότερα επίπεδα, αλλά αυτό δεν κρίθηκε αναγκαίο να υλοποιηθεί, καθώς ο κώδικας στο μεγαλύτερο μέρος του κάθε επιπέδου θα έμενε ο ίδιος, θα άλλαζε μόνο ο αριθμός των πλακιδίων καθώς και κάποιες σταθερές ακόμη. Επίσης, θα μπορούσαν να προστεθούν διάφορα υπομενού, με τον ίδιο τρόπο που προστέθηκε το About, όπως ένα υπομενού Hints, το οποίο να δείχνει ολόκληρη την εικόνα πριν το ανακάτεμα, για τη βοήθεια του παίκτη ώστε να κατανοήσει πως ήταν η εικόνα πριν το ανακάτεμα. Το συγκεκριμένο μενού στην ουσία θα ήταν μια εικόνα, οπότε ούτε αυτό κρίθηκε αναγκαίο, καθώς εικόνα χρησιμοποιήσαμε και στο background της Αρχικής Οθόνης, υλοποιημένη με παρόμοιο τρόπο. Επιπλέον, η εφαρμογή θα μπορούσε να έχει τυχαία επιλογή εικόνας, κάνοντάς την όχι και τόσο επαναλαμβανόμενη. Το πρόβλημα που παρουσιάζεται σε ένα τέτοιο σενάριο είναι ότι έχουμε τις εικόνες ήδη τεμαχισμένες με τη συγκεκριμένη υλοποίηση, άρα μια λύση θα ήταν ένας εσωτερικός τεμαχισμός τυχαίας εικόνας. Το ίδιο ισχύει και για την επιλογή επιπέδου δυσκολίας από τον χρήστη, όπου θα μπορούσε ο ίδιος να επιλέξει διαστάσεις και να υλοποιείται αυτόματα από την εφαρμογή.

Τελικά, η ανάπτυξη της εφαρμογής, και η υλοποίησή της, πέτυχαν ακριβώς τον σκοπό της Πτυχιακής Εργασίας. Να δειχθεί βήμα-βήμα το χτίσιμο μιας εφαρμογής γρίφων/ruzzle σε περιβάλλον Android από το μηδέν, από μία ιδέα, και να φτάσει να εκτελείται στα χέρια μας.

ΒΙΒΛΙΟΓΡΑΦΙΑ/ΙΣΤΟΓΡΑΦΙΑ

ΑΠΟΛΥΤΗ JAVA, Walter Savitch, 2006(Επιμέλεια: Δρ. Δημήτρης Ιακωβίδης, ΕΚΔΟΣΕΙΣ ΙΩΝ).

Dave Park. (accessed 24/4/2018). How to Build a Basic 2D-puzzle in Android Studio. Retrieved from:
<https://www.youtube.com/watch?v=YKbFx8PDTIo>

ΑΝΑΦΟΡΕΣ/ΠΑΡΑΠΟΜΠΕΣ

1. <https://developer.android.com/training/gestures/detector> [accessed 14/6/2018]
2. <https://developer.android.com/reference/android/view/GestureDetector>
[accessed 14/6/2018]
3. <https://developer.android.com/reference/android/view/GestureDetector.OnGestureListener> [accessed 14/6/2018]
4. <https://github.com/DaveNOTDavid/sample-puzzle/blob/master/app/src/main/java/com/davenotdavid/samplepuzzle/GestureDetectGridView.java> [accessed 14/6/2018]
5. <https://developer.android.com/reference/android/view/MotionEvent>
[accessed 14/6/2018]
6. <https://developer.android.com/reference/android/view/ViewTreeObserver>
[accessed 14/6/2018]

ΠΑΡΑΡΤΗΜΑ: ΚΩΔΙΚΑΣ ΕΦΑΡΜΟΓΗΣ

Στο παρακάτω παράρτημα περιέχεται εξ' ολοκλήρου ο κώδικας που χρησιμοποιήθηκε στην πτυχιακή εργασία. Σε μερικά σημεία ο κώδικας μπορεί να διαφέρει ελαφρά.

AndroidManifest.xml

```
1. <?xml version="1.0" encoding="utf-8"?>
2. <manifest xmlns:android="http://schemas.android.com/apk/res/android"
3.     package="com.android.sitsero_games.ptixiakiao">
4.
5.     <application
6.         android:allowBackup="true"
7.         android:icon="@mipmap/ic_launcher"
8.         android:label="Πτυχιακή Αναγνωστόπουλος Ορ."
9.         android:roundIcon="@mipmap/ic_launcher_round"
10.        android:supportsRtl="true"
11.        android:theme="@style/AppTheme">
12.        <activity android:name=".MainActivity"></activity>
13.        <activity android:name=".StartMenu" android:screenOrientation="portrait">
14.            <intent-filter>
15.                <action android:name="android.intent.action.MAIN" />
16.                <category android:name="android.intent.category.LAUNCHER" />
17.            </intent-filter>
18.        </activity>
19.        <activity android:name=".About" android:screenOrientation="portrait"></activity>
20.    </application>
21.
22. </manifest>
```

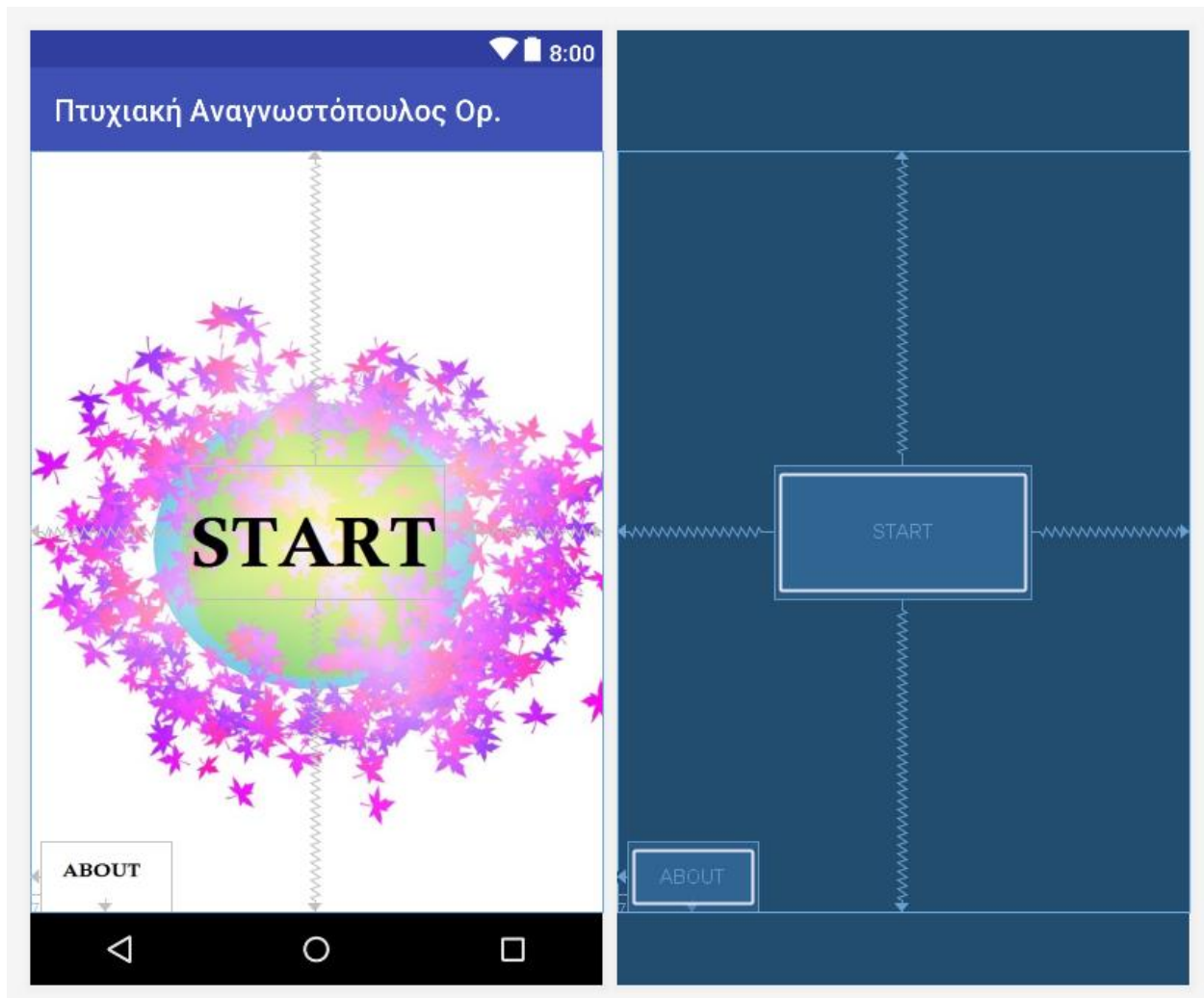
StartMenu.java

```
1. package com.android.sitsero_games.ptixiakiao;
2.
3. import android.content.Intent;
4. import android.support.v7.app.AppCompatActivity;
5. import android.os.Bundle;
6. import android.view.View;
7. import android.widget.Button;
8.
9. public class StartMenu extends AppCompatActivity {
10.
11.     @Override
12.     protected void onCreate(Bundle savedInstanceState) {
13.         super.onCreate(savedInstanceState);
14.         setContentView(R.layout.activity_start_menu);
15.
16.         Button startPuzzle = (Button) findViewById(R.id.button1);
17.         startPuzzle.setOnClickListener(new View.OnClickListener() {
18.             @Override
19.             public void onClick(View view) {
20.                 Intent startIntent = new Intent(StartMenu.this, MainActivity.class);
21.                 startActivity(startIntent);
22.             }
23.         });
24.         Button about = (Button) findViewById(R.id.button2);
25.         about.setOnClickListener(new View.OnClickListener() {
26.             @Override
27.             public void onClick(View view) {
28.                 Intent startIntent = new Intent(StartMenu.this, About.class);
29.                 startActivity(startIntent);
30.             }
31.         });
32.     }
33. }
```

activity_start_menu.xml - Text

```
1. <?xml version="1.0" encoding="utf-8"?>
2. <android.support.constraint.ConstraintLayout xmlns:android="http://schemas.android.com/
   apk/res/android"
3.     xmlns:app="http://schemas.android.com/apk/res-auto"
4.     xmlns:tools="http://schemas.android.com/tools"
5.     android:layout_width="match_parent"
6.     android:layout_height="match_parent"
7.     android:background="@drawable/background"
8.     tools:context=".StartMenu">
9.
10.    <Button
11.        android:id="@+id/button1"
12.        android:layout_width="200dp"
13.        android:layout_height="144dp"
14.        android:layout_marginBottom="8dp"
15.        android:layout_marginTop="8dp"
16.        android:alpha="0"
17.        android:text="Start"
18.        app:layout_constraintBottom_toBottomOf="parent"
19.        app:layout_constraintEnd_toEndOf="parent"
20.        app:layout_constraintStart_toStartOf="parent"
21.        app:layout_constraintTop_toTopOf="parent" />
22.
23.    <Button
24.        android:id="@+id/button2"
25.        android:layout_width="wrap_content"
26.        android:layout_height="48dp"
27.        android:layout_marginBottom="17dp"
28.        android:layout_marginStart="16dp"
29.        android:alpha="0"
30.        android:text="About"
31.        app:layout_constraintBottom_toBottomOf="parent"
32.        app:layout_constraintStart_toStartOf="parent" />
33. </android.support.constraint.ConstraintLayout>
```

activity_start_menu.xml - Design



About.java

```
1. package com.android.sitsero_games.ptixiakiao;
2.
3. import android.support.v7.app.AppCompatActivity;
4. import android.os.Bundle;
5.
6. public class About extends AppCompatActivity {
7.
8.     @Override
9.     protected void onCreate(Bundle savedInstanceState) {
10.         super.onCreate(savedInstanceState);
11.         setContentView(R.layout.activity_about);
12.     }
13. }
```

activity_about.xml - Text

```
1. <?xml version="1.0" encoding="utf-8"?>
2. <android.support.constraint.ConstraintLayout xmlns:android="http://schemas.android.com/
   apk/res/android"
3.     xmlns:app="http://schemas.android.com/apk/res-auto"
4.     xmlns:tools="http://schemas.android.com/tools"
5.     android:layout_width="match_parent"
6.     android:layout_height="match_parent"
7.     tools:context=".About"
8.     tools:layout_editor_absoluteY="81dp">
9.
10.    <TextView
11.        android:id="@+id/textView4"
12.        android:layout_width="wrap_content"
13.        android:layout_height="wrap_content"
14.        android:text="Τμήμα Μηχανικών Πληροφορικής Τ.Ε."
15.        app:layout_constraintBottom_toTopOf="@+id/textView5"
16.        app:layout_constraintEnd_toEndOf="parent"
17.        app:layout_constraintHorizontal_bias="0.5"
18.        app:layout_constraintStart_toStartOf="parent"
19.        app:layout_constraintTop_toTopOf="parent" />
20.
21.    <TextView
22.        android:id="@+id/textView5"
23.        android:layout_width="wrap_content"
24.        android:layout_height="wrap_content"
25.        android:text="Σχολή Τεχνολογικών Εφαρμογών (έδρα Σπάρτη)"
26.        app:layout_constraintBottom_toTopOf="@+id/textView3"
27.        app:layout_constraintEnd_toEndOf="parent"
28.        app:layout_constraintHorizontal_bias="0.5"
29.        app:layout_constraintStart_toStartOf="parent"
30.        app:layout_constraintTop_toBottomOf="@+id/textView4" />
31.
32.    <TextView
33.        android:id="@+id/textView3"
34.        android:layout_width="wrap_content"
35.        android:layout_height="25dp"
36.        android:text="Τ.Ε.Ι. Πελοποννήσου"
37.        app:layout_constraintBottom_toTopOf="@+id/textView"
38.        app:layout_constraintEnd_toEndOf="parent"
39.        app:layout_constraintHorizontal_bias="0.5"
40.        app:layout_constraintStart_toStartOf="parent"
41.        app:layout_constraintTop_toBottomOf="@+id/textView5" />
42.
43.    <TextView
44.        android:id="@+id/textView"
45.        android:layout_width="117dp"
46.        android:layout_height="52dp"
47.        android:layout_gravity="center"
48.        android:gravity="center"
49.        android:text="Πτυχιακή Εργασία"
50.        app:layout_constraintBottom_toTopOf="@+id/textView8"
51.        app:layout_constraintEnd_toEndOf="parent"
52.        app:layout_constraintHorizontal_bias="0.5"
53.        app:layout_constraintStart_toStartOf="parent"
54.        app:layout_constraintTop_toBottomOf="@+id/textView3" />
```

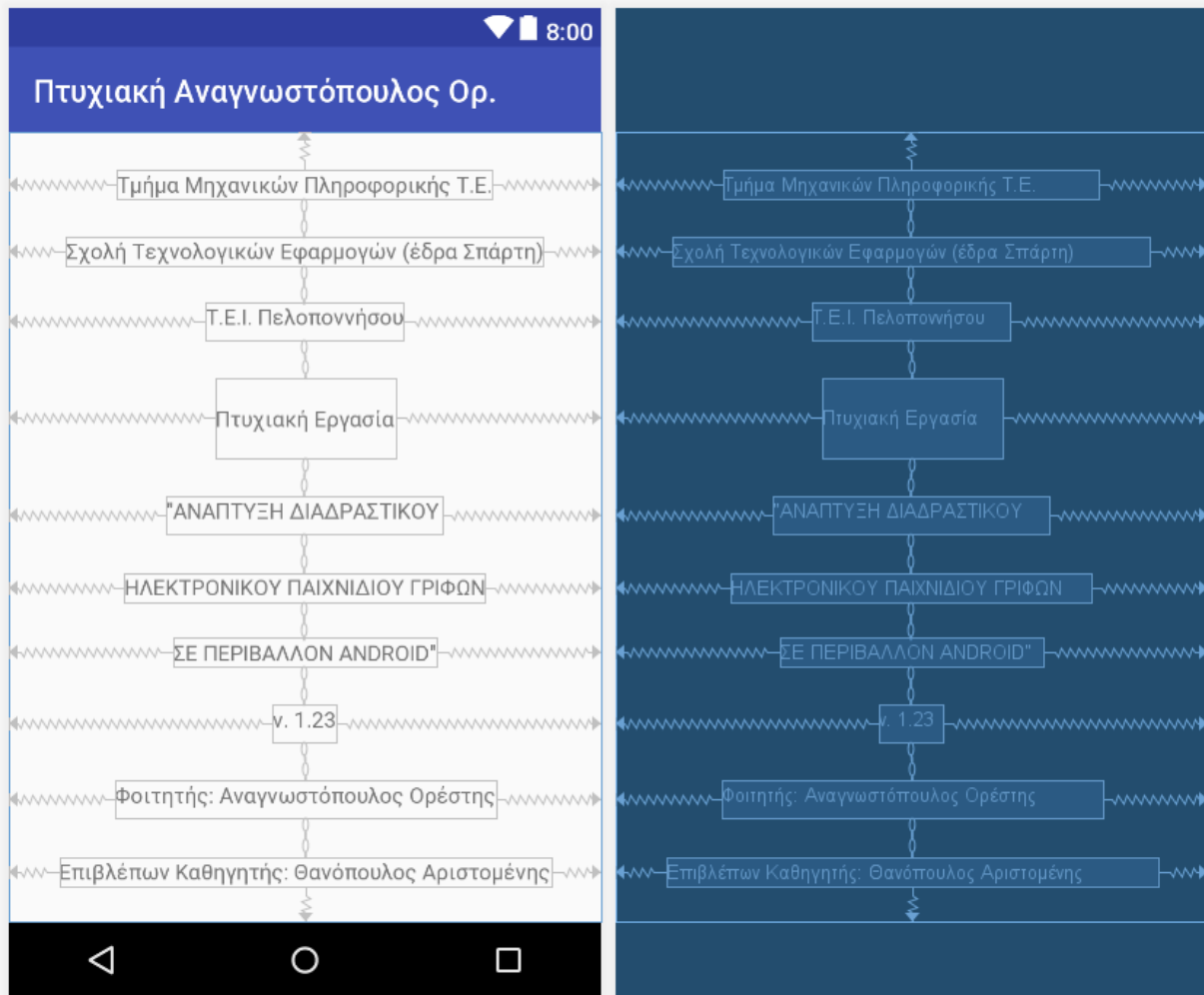
```

55.
56. <TextView
57.     android:id="@+id/textView8"
58.     android:layout_width="180dp"
59.     android:layout_height="25dp"
60.     android:text="'ΑΝΑΠΤΥΞΗ ΔΙΑΔΡΑΣΤΙΚΟΥ'
61.     app:layout_constraintBottom_toTopOf="@+id/textView9"
62.     app:layout_constraintEnd_toEndOf="parent"
63.     app:layout_constraintHorizontal_bias="0.5"
64.     app:layout_constraintStart_toStartOf="parent"
65.     app:layout_constraintTop_toBottomOf="@+id/textView" />
66.
67. <TextView
68.     android:id="@+id/textView9"
69.     android:layout_width="wrap_content"
70.     android:layout_height="wrap_content"
71.     android:text="ΗΛΕΚΤΡΟΝΙΚΟΥ ΠΑΙΧΝΙΔΙΟΥ ΓΡΙΦΩΝ"
72.     app:layout_constraintBottom_toTopOf="@+id/textView10"
73.     app:layout_constraintEnd_toEndOf="parent"
74.     app:layout_constraintHorizontal_bias="0.5"
75.     app:layout_constraintStart_toStartOf="parent"
76.     app:layout_constraintTop_toBottomOf="@+id/textView8" />
77.
78. <TextView
79.     android:id="@+id/textView10"
80.     android:layout_width="wrap_content"
81.     android:layout_height="wrap_content"
82.     android:text="'ΣΕ ΠΕΡΙΒΑΛΛΟΝ ANDROID'"
83.     app:layout_constraintBottom_toTopOf="@+id/textView7"
84.     app:layout_constraintEnd_toEndOf="parent"
85.     app:layout_constraintHorizontal_bias="0.5"
86.     app:layout_constraintStart_toStartOf="parent"
87.     app:layout_constraintTop_toBottomOf="@+id/textView9" />
88.
89. <TextView
90.     android:id="@+id/textView7"
91.     android:layout_width="wrap_content"
92.     android:layout_height="25dp"
93.     android:text="v. 1.23"
94.     app:layout_constraintBottom_toTopOf="@+id/textView2"
95.     app:layout_constraintEnd_toEndOf="parent"
96.     app:layout_constraintHorizontal_bias="0.5"
97.     app:layout_constraintStart_toStartOf="parent"
98.     app:layout_constraintTop_toBottomOf="@+id/textView10" />
99.
100. <TextView
101.     android:id="@+id/textView2"
102.     android:layout_width="wrap_content"
103.     android:layout_height="25dp"
104.     android:text="Φοιτητής: Αναγνωστόπουλος Ορέστης"
105.     app:layout_constraintBottom_toTopOf="@+id/textView6"
106.     app:layout_constraintEnd_toEndOf="parent"
107.     app:layout_constraintHorizontal_bias="0.5"
108.     app:layout_constraintStart_toStartOf="parent"
109.     app:layout_constraintTop_toBottomOf="@+id/textView7" />
110.
111. <TextView
112.     android:id="@+id/textView6"
113.     android:layout_width="wrap_content"
114.     android:layout_height="wrap_content"
115.     android:text="Επιβλέπων Καθηγητής: Θανόπουλος Αριστομένης"

```

```
116.         app:layout_constraintBottom_toBottomOf="parent"
117.         app:layout_constraintEnd_toEndOf="parent"
118.         app:layout_constraintHorizontal_bias="0.5"
119.         app:layout_constraintStart_toStartOf="parent"
120.         app:layout_constraintTop_toBottomOf="@+id/textView2" />
121.
122.     </android.support.constraint.ConstraintLayout>
```

activity_about.xml - Design



MainActivity.java

```
1. package com.android.sitsero_games.ptixiakiao;
2.
3. import android.content.Context;
4. import android.support.v7.app.AppCompatActivity;
5. import android.os.Bundle;
6. import java.util.Random;
7.
8. import android.view.ViewTreeObserver;
9. import android.widget.Button;
10. import android.widget.Toast;
11.
12. import java.util.ArrayList;
13.
14. public class MainActivity extends AppCompatActivity {
15.
16.     private static final int STILES=3;
17.     private static final int DIASTASEIS=STILES*STILES;
18.     private static String[] listaPlakidiwn;
19.     private static GestureDetectorGridView newGridView;
20.     private static int platosSthlhs,upsosSthlhs;
21.     public static boolean swsth=false;
22.
23.
24.     public static final String UP = "up";
25.     public static final String DOWN= "down";
26.     public static final String LEFT= "left";
27.     public static final String RIGHT= "right";
28.     @Override
29.     protected void onCreate(Bundle savedInstanceState) {
30.         super.onCreate(savedInstanceState);
31.         setContentView(R.layout.activity_main);
32.
33.         arxikopoihsh();
34.
35.         anakatema();
36.
37.         orismosDiastasewn();
38.     }
39.
40.     private void orismosDiastasewn() {
41.         ViewTreeObserver to = newGridView.getViewTreeObserver();
42.         to.addOnGlobalLayoutListener(new ViewTreeObserver.OnGlobalLayoutListener() {
43.             @Override
44.             public void onGlobalLayout() {
45.                 newGridView.getViewTreeObserver().removeOnGlobalLayoutListener(this);
46.                 int emfanishPlatous = newGridView.getMeasuredWidth();
47.                 int emfanishUpsous = newGridView.getMeasuredHeight();
48.
49.                 int upsosStatusBar = travhgmaUpsousMparas(getApplicationContext());
50.                 int apaitoumenoUpsos = emfanishUpsous - upsosStatusBar;
51.
52.                 platosSthlhs = emfanishPlatous/STILES;
53.                 upsosSthlhs = apaitoumenoUpsos/STILES;
54.
55.                 emfanish(getApplicationContext());
56.             }
57.         });
58.     }
```

```

59.
60.     private int travhgmaUpsousMparas(Context context){
61.         int i=0;
62.         int resourceId = context.getResources().getIdentifier("status_bar_height" , "di
men" , "android");
63.         if (resourceId >0 ){
64.             i = context.getResources().getDimensionPixelSize(resourceId);
65.         }
66.         return i;
67.     }
68.
69.     private static void emfanish(Context context) {
70.         ArrayList<Button> koumpia= new ArrayList<>();
71.         Button koumpi;
72.         for (int i=0;i<listaPlakidiwn.length; i++)
73.         {
74.             koumpi = new Button(context);
75.             if( listaPlakidiwn[i].equals("0"))
76.                 koumpi.setBackgroundResource(R.drawable.image_part_001);
77.             else if (listaPlakidiwn[i].equals("1"))
78.                 koumpi.setBackgroundResource(R.drawable.image_part_002);
79.             else if (listaPlakidiwn[i].equals("2"))
80.                 koumpi.setBackgroundResource(R.drawable.image_part_003);
81.             else if (listaPlakidiwn[i].equals("3"))
82.                 koumpi.setBackgroundResource(R.drawable.image_part_004);
83.             else if (listaPlakidiwn[i].equals("4"))
84.                 koumpi.setBackgroundResource(R.drawable.image_part_005);
85.             else if (listaPlakidiwn[i].equals("5"))
86.                 koumpi.setBackgroundResource(R.drawable.image_part_006);
87.             else if (listaPlakidiwn[i].equals("6"))
88.                 koumpi.setBackgroundResource(R.drawable.image_part_007);
89.             else if (listaPlakidiwn[i].equals("7"))
90.                 koumpi.setBackgroundResource(R.drawable.image_part_008);
91.             else if (listaPlakidiwn[i].equals("8"))
92.                 koumpi.setBackgroundResource(R.drawable.image_part_009);
93.             koumpia.add(koumpi);
94.         }
95.         newGridView.setAdapter(new CustomAdapter(koumpia, platosSthlhs, upsosSthlhs));
96.
97.     }
98.
99.     private void anakatema() {
100.         int deikths;
101.         String temp;
102.         Random random = new Random();
103.
104.         for(int i=listaPlakidiwn.length-1; i>0; i--)
105.         {
106.             deikths = random.nextInt(i+1);
107.             temp=listaPlakidiwn[deikths];
108.             listaPlakidiwn[deikths]=listaPlakidiwn[i];
109.             listaPlakidiwn[i]=temp;
110.         }
111.     }
112.
113.     private void arxikopoihsh() {
114.         newGridView = (GestureDetectGridView) findViewById(R.id.grid);
115.         newGridView.setNumColumns(STILES);
116.
117.         listaPlakidiwn = new String[DIASTASEIS];
118.         for (int i=0; i< DIASTASEIS; i++)

```

```

119.         {
120.             listaPlakidiwn[i] = String.valueOf(i);
121.         }
122.     }
123.
124.     //Εναλλαγή πλακιδίων
125.     public static void enallagh(Context context,int position, int enallassomeno)
126.     {
127.         String newPosition = listaPlakidiwn[position + enallassomeno];
128.         listaPlakidiwn[position+enallassomeno] = listaPlakidiwn[position];
129.         listaPlakidiwn[position] = newPosition;
130.         emfanish(context);
131.         if( lumenh()){
132.             Toast.makeText(context,"Συγχαρητήρια, νικήσατε!", Toast.LENGTH_LONG)
133.             .show();
134.         }
135.
136.         private static boolean lumenh() {
137.
138.             for(int i=0;i<listaPlakidiwn.length;i++){
139.                 if(listaPlakidiwn[i].equals(String.valueOf(i))) {
140.                     swsth = true;
141.                 }
142.                 else{
143.                     swsth = false;
144.                     break;
145.                 }
146.             }
147.             return swsth;
148.         }
149.
150.         public static void enallaghPlakidiwn(Context context,String direction, int p
151.         osition){
152.             //Πάνω αριστερά πλακίδιο
153.             if(position==0){
154.                 if (direction.equals(RIGHT)){
155.                     enallagh(context, position, 1);
156.                 }
157.                 else if (direction.equals(DOWN)){
158.                     enallagh(context,position,STILES);
159.                 }
160.                 else{
161.                     Toast.makeText(context, "Μη έγκυρη κίνηση", Toast.LENGTH_SHORT).
162.                     show();
163.                 }
164.             }
165.             //ενδιάμεσα πλακίδια πρώτης σειράς
166.             else if ( position>0 && position < STILES - 1){
167.                 if (direction.equals(LEFT))
168.                     enallagh(context,position,-1);
169.                 else if (direction.equals(RIGHT))
170.                     enallagh(context,position,1);
171.                 else if(direction.equals(DOWN))
172.                     enallagh(context, position, STILES);
173.                 else
174.                     Toast.makeText(context, "Μη έγκυρη κίνηση", Toast.LENGTH_SHORT).
175.                     show();
176.             }

```



```

175.         //πάνω δεξί πλακίδιο
176.         else if (position == STILES-1){
177.             if(direction.equals(LEFT))
178.                 enallagh(context, position,-1);
179.             else if(direction.equals(DOWN))
180.                 enallagh(context, position, STILES);
181.             else
182.                 Toast.makeText(context, "Μη έγκυρη κίνηση", Toast.LENGTH_SHORT).
show();
183.         }
184.         //πρώτο πλακίδιο δεύτερης σειράς
185.         else if (position == STILES){
186.             if(direction.equals(UP))
187.                 enallagh(context, position,-STILES);
188.             else if (direction.equals(RIGHT))
189.                 enallagh(context,position,1);
190.             else if(direction.equals(DOWN))
191.                 enallagh(context,position,STILES);
192.             else
193.                 Toast.makeText(context, "Μη έγκυρη κίνηση", Toast.LENGTH_SHORT).
show();
194.         }
195.         //Ενδιάμεσα πλακίδια δεύτερης σειράς
196.         else if(position > STILES && position < 2*STILES-1) {
197.             if (direction.equals(UP))
198.                 enallagh(context, position, -STILES);
199.             else if (direction.equals(DOWN))
200.                 enallagh(context, position, STILES);
201.             else if (direction.equals(LEFT))
202.                 enallagh(context, position, -1);
203.             else if (direction.equals(RIGHT))
204.                 enallagh(context, position, 1);
205.             //Δεν χρειάζεται
206.             else
207.                 Toast.makeText(context, "Μη έγκυρη κίνηση", Toast.LENGTH_SHORT).
show();
208.         }
209.         //τελευταίο πλακίδιο μεσαίας σειράς
210.         else if(position == 2*STILES-1){
211.             if(direction.equals(UP))
212.                 enallagh(context,position,-STILES);
213.             else if (direction.equals(DOWN))
214.                 enallagh(context, position, STILES);
215.             else if (direction.equals(LEFT))
216.                 enallagh(context, position, -1);
217.             else
218.                 Toast.makeText(context, "Μη έγκυρη κίνηση", Toast.LENGTH_SHORT).
show();
219.         }
220.         //πρώτο πλακίδιο τρίτης σειράς
221.         else if(position == 2*STILES){
222.             if(direction.equals(UP))
223.                 enallagh(context,position,-STILES);
224.             else if (direction.equals(RIGHT))
225.                 enallagh(context, position, 1);
226.             else
227.                 Toast.makeText(context, "Μη έγκυρη κίνηση", Toast.LENGTH_SHORT).
show();
228.         }
229.         //ενδιάμεσα πλακίδια τρίτης σειράς
230.         else if(position>2*STILES && position < 3*STILES -1){

```

```

231.         if (direction.equals(UP))
232.             enallagh(context, position, -STILES);
233.         else if (direction.equals(LEFT))
234.             enallagh(context, position, -1);
235.         else if (direction.equals(RIGHT))
236.             enallagh(context, position, 1);
237.         else
238.             Toast.makeText(context, "Μη έγκυρη κίνηση", Toast.LENGTH_SHORT).
show();
239.     }
240.     //τελευταίο πλακίδιο τρίτης σειράς
241.     else if(position==3*STILES-1){
242.         if (direction.equals(UP))
243.             enallagh(context, position, -STILES);
244.         else if (direction.equals(LEFT))
245.             enallagh(context, position, -1);
246.         else
247.             Toast.makeText(context, "Μη έγκυρη κίνηση", Toast.LENGTH_SHORT).
show();
248.     }
249. }
250. }

```

activity_main.xml

```
1. <?xml version="1.0" encoding="utf-8"?>
2. <com.android.sitsero_games.ptixiakiao.GestureDetectGridView xmlns:android="http://schemas.android.com/apk/res/android"
3.     xmlns:tools="http://schemas.android.com/tools"
4.     android:id="@+id/grid"
5.     android:layout_width="match_parent"
6.     android:layout_height="match_parent"
7.     tools:context=".MainActivity">
8. </com.android.sitsero_games.ptixiakiao.GestureDetectGridView>
```

GestureDetectGridView.java

```
1. package com.android.sitsero_games.ptixiakiao;
2.
3. import android.annotation.TargetApi;
4. import android.content.Context;
5. import android.os.Build;
6. import android.util.AttributeSet;
7. import android.view.GestureDetector;
8. import android.view.MotionEvent;
9. import android.widget.GridView;
10.
11. public class GestureDetectGridView extends GridView {
12.     private GestureDetector elegkths;
13.     private boolean mFlingConfirmed = false;
14.     private float mTouchX;
15.     private float mTouchY;
16.
17.     private static final int SWIPE_MIN_DISTANCE = 100;
18.     private static final int SWIPE_MAX_OFF_PATH = 100;
19.     private static final int SWIPE_THRESHOLD_VELOCITY = 100;
20.
21.     public GestureDetectGridView(Context context) {
22.         super(context);
23.         init(context);
24.     }
25.
26.     public GestureDetectGridView(Context context, AttributeSet attrs) {
27.         super(context, attrs);
28.         init(context);
29.     }
30.
31.     public GestureDetectGridView(Context context, AttributeSet attrs, int defStyleAttr)
32.     {
33.         super(context, attrs, defStyleAttr);
34.         init(context);
35.     }
36.     @TargetApi(Build.VERSION_CODES.LOLLIPOP) // API 21
37.     public GestureDetectGridView(Context context, AttributeSet attrs, int defStyleAttr,
38.     int defStyleRes) {
39.         super(context, attrs, defStyleAttr, defStyleRes);
40.         init(context);
41.     }
42.     //Αρχικοποίηση των συμβάντων κίνησης μέσω ενός Gesture Detector
43.     private void init(final Context context) {
44.         elegkths = new GestureDetector(context, new GestureDetector.SimpleOnGestureList
45.         ener() {
46.             @Override
47.             //Αποδοχή όλων των συμβάντων κίνησης
48.             public boolean onDown(MotionEvent event) {
49.                 return true;
50.             }
51.             @Override
52.             public boolean onFling(MotionEvent e1, MotionEvent e2, float velocityX,
```

```

53.         float velocityY) {
54.             final int position = GestureDetectGridView.this.pointToPosition
55.                 (Math.round(e1.getX()), Math.round(e1.getY()));
56.
57.             if (Math.abs(e1.getY() - e2.getY()) > SWIPE_MAX_OFF_PATH) {
58.                 if (Math.abs(e1.getX() - e2.getX()) > SWIPE_MAX_OFF_PATH
59.                     || Math.abs(velocityY) < SWIPE_THRESHOLD_VELOCITY) {
60.                     return false;
61.                 }
62.                 if (e1.getY() - e2.getY() > SWIPE_MIN_DISTANCE) {
63.                     MainActivity.enallaghPlakidiwn(context, MainActivity.UP, posi
64. on);
65.                 } else if (e2.getY() - e1.getY() > SWIPE_MIN_DISTANCE) {
66.                     MainActivity.enallaghPlakidiwn(context, MainActivity.DOWN, posi
67. tion);
68.                 } else {
69.                     if (Math.abs(velocityX) < SWIPE_THRESHOLD_VELOCITY) {
70.                         return false;
71.                     }
72.                     if (e1.getX() - e2.getX() > SWIPE_MIN_DISTANCE) {
73.                         MainActivity.enallaghPlakidiwn(context, MainActivity.LEFT, posi
74. tion);
75.                     } else if (e2.getX() - e1.getX() > SWIPE_MIN_DISTANCE) {
76.                         MainActivity.enallaghPlakidiwn(context, MainActivity.RIGHT, pos
77. ition);
78.                     }
79.                 }
80.             }
81.         }
82.
83.         @Override
84.         public boolean onInterceptTouchEvent(MotionEvent ev) {
85.             int action = ev.getActionMasked();
86.             elegkths.onTouchEvent(ev);
87.             //Αν η χειρονομία ακυρώθηκε ή τελείωσε
88.             if (action == MotionEvent.ACTION_CANCEL || action == MotionEvent.ACTION_UP) {
89.                 mFlingConfirmed = false;
90.                 //Αλλιώς αν έχει ξεκινήσει χειρονομία(ACTION_DOWN παίρνει την μεταβλητή της
91. αρχικής τοποθεσίας της χειρονομίας)
92.                 //παίρνουμε τις μεταβλητές X και Ψ της οθόνης
93.             } else if (action == MotionEvent.ACTION_DOWN) {
94.                 mTouchX = ev.getX();
95.                 mTouchY = ev.getY();
96.             } else {
97.                 //εαν τελειώσει η κίνηση
98.                 if (mFlingConfirmed) {
99.                     return true;
100.                 }
101.                 //παίρνουμε τις απόλυτες αποστάσεις του swipe, και αν είναι μέσα στα
102. επιτρεπτά όρια προχωράμε κανονικά
103.                 float dX = (Math.abs(ev.getX() - mTouchX));
104.                 float dY = (Math.abs(ev.getY() - mTouchY));
105.                 if ((dX > SWIPE_MIN_DISTANCE) || (dY > SWIPE_MIN_DISTANCE)) {
106.                     mFlingConfirmed = true;
107.                     return true;
108.                 }
109.             }
110.         }

```

```
108.         //Επιστρέφουμε τη μεταβλητή συμβάντος
109.         return super.onInterceptTouchEvent(ev);
110.     }
111.
112.     @Override
113.     public boolean onTouchEvent(MotionEvent ev) {
114.         return elegkths.onTouchEvent(ev);
115.     }
116. }
```

CustomAdapter.java

```
1. package com.android.sitsero_games.ptixiakiao;
2.
3. import android.view.View;
4. import android.view.ViewGroup;
5. import android.widget.BaseAdapter;
6. import android.widget.Button;
7.
8. import java.util.ArrayList;
9.
10. public class CustomAdapter extends BaseAdapter {
11.
12.     private ArrayList<Button> newkoumpia = null;
13.     private int platosSthlhs, upsosSthlhs;
14.
15.     public CustomAdapter(ArrayList<Button> koumpia, int platos, int upsos) {
16.         newkoumpia = koumpia;
17.         platosSthlhs = platos;
18.         upsosSthlhs = upsos;
19.     }
20.
21.     @Override
22.     public int getCount() {
23.         return newkoumpia.size();
24.     }
25.
26.     @Override
27.     public Object getItem(int i) {
28.         return (Object) newkoumpia.get(i);
29.     }
30.
31.     @Override
32.     public long getItemId(int i) {
33.         return i;
34.     }
35.
36.     @Override
37.     public View getView(int i, View view, ViewGroup viewGroup) {
38.         Button koumpi;
39.         if(view==null){
40.             koumpi = newkoumpia.get(i);
41.         }
42.         else{
43.             koumpi = (Button) view;
44.         }
45.
46.         android.widget.AbsListView.LayoutParams parametroi = new android.widget.AbsList
View.LayoutParams(platosSthlhs, upsosSthlhs);
47.         koumpi.setLayoutParams(parametroi);
48.         return koumpi;
49.     }
50. }
```