ΚΕΦΑΛΑΙΟ 7

Υλοποίηση εφαρμογών σε προγραμματιστικά περιβάλλοντα

Διδακτικές ενότητες

7.1 Προγραμματισμός εφαρμογών για φορητές συσκευές
7.2 Αντικειμενοστρεφής προγραμματισμός σε 3D περιβάλλον

Διδακτικοί στόχοι

Σκοπός του κεφαλαίου είναι οι μαθητές να υλοποιήσουν στην πράξη ολοκληρωμένες εφαρμογές σε ένα σύγχρονο περιβάλλον προγραμματισμού, ακολουθώντας βήμα - βήμα όλες τις φάσεις του κύκλου ζωής εφαρμογών.

Οι μαθητές πρέπει να είναι σε θέση:

- √ να δημιουργούν μια εφαρμογή με το οπτικό περιβάλλον προγραμματισμού App Inventor για φορητές συσκευές (κινητά, ταμπλέτες-tablets) με λειτουργικό σύστημα Android.
- √ να αναγνωρίζουν τις έννοιες κλάση, αντικείμενο, ιδιότητα, μέθοδος και κληρονομικότητα σε ένα αντικειμενοστρεφές περιβάλλον προγραμματισμού.
- √ να δημιουργούν έναν εικονικό κόσμο στο τρισδιάστατο (3D) περιβάλλον Alice με δυναμικές κινήσεις χαρακτήρων και αλληλεπίδραση με τον χρήστη.

Ερωτήματα

- √ Πώς πιστεύετε ότι μπορείτε να δημιουργήσετε εφαρμογή που θα τη χρησιμοποιείτε στο κινητό σας τηλέφωνο;
- √ Γνωρίζετε τις έννοιες *κλάση* και *αντικείμενο* στον αντικειμενοστρεφή προγραμματισμό;
- √ Τι είναι η κληρονομικότητα στον αντικειμενοστρεφή προγραμματισμό;
- √ Πιστεύετε ότι μπορείτε να δημιουργήσετε ένα παιχνίδι παρόμοιο με το αγαπημένο σας ηλεκτρονικό παιχνίδι;

Βασική ορολογία

Οπτικός προγραμματισμός, αντικειμενοστρεφής προγραμματισμός, κλάση, αντικείμενο, ιδιότητα, κληρονομικότητα, μέθοδος, 3D περιβάλλον, App Inventor, Alice

Εισαγωγή

Το παρόν κεφάλαιο χωρίζεται σε δυο ενότητες. Σε κάθε ενότητα μας δίνεται η ευκαιρία να δημιουργήσουμε μια ολοκληρωμένη εφαρμογή - παιχνίδι εφαρμόζοντας στην πράξη όλες τις φάσεις του κύκλου ζωής μιας εφαρμογής. Τα προγραμματιστικά περιβάλλοντα που αναλύονται ανήκουν στην κατηγορία των ΕΛ/ΛΑΚ (Ελεύθερο Λογισμικό / Λογισμικό Ανοικτού Κώδικα).

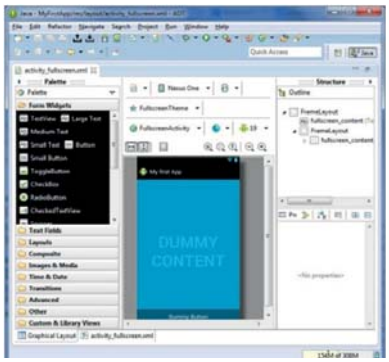
7.1 Προγραμματισμός εφαρμογών για φορητές συσκευές

Ανάπτυξη εφαρμογών για φορητές συσκευές

Οι φορητές συσκευές, κυρίως τα έξυπνα κινητά (smartphones) και οι ταμπλέτες (tablets), έχουν διεισδύσει σε πολλούς τομείς της ανθρώπινης δραστηριότητας, όπως είναι η ενημέρωση, η ψυχαγωγία και η εργασία. Οι συσκευές αυτές γίνονται ολοένα και πιο δημοφιλείς και χρηστικές χάρη στο πλήθος εφαρμογών και δυνατοτήτων που διαθέτουν. Επίσης, τείνουν σε αρκετές περιπτώσεις να αντικαταστήσουν τους υπολογιστές και μια πληθώρα άλλων συσκευών, όπως είναι οι φωτογραφικές μηχανές και οι MP3 players.

Εικόνα 7.1. Δημοφιλή Λειτουργικά Συστήματα για φορητές συσκευές

Οι φορητές συσκευές υποστηρίζονται από Λειτουργικά Συστήματα τα οποία διακρίνονται από συγκεκριμένα χαρακτηριστικά. Τα δημοφιλέστερα Λειτουργικά Συστήματα είναι το **iOS**, το **Android**, το **Windows Phone**, το **Symbian** και το **BlackBerry**. Οι επαγγελματίες προγραμματιστές εφαρμογών για φορητές συσκευές χρησιμοποιούν ολοκληρωμένα περιβάλλοντα ανάπτυξης εφαρμογών, επαγγελματικές γλώσσες προγραμματισμού (π.χ. Java) και αντιμετωπίζουν προβλήματα που σχετίζονται με τους περιορισμένους πόρους των συσκευών (π.χ. επεξεργαστής, μνήμη), με το μικρό μέγεθος της διεπαφής χρήστη, με θέματα ασφάλειας, με τεχνολογίες αυτόματου προσδιορισμού της θέσης του χρήστη κ.ά. Ένας αρχάριος προγραμματιστής που φιλοδοξεί να αναπτύξει τις πρώτες του εφαρμογές για Android μπορεί να χρησιμοποιήσει το εκπαιδευτικό περιβάλλον App Inventor.



Εικόνα 7.2. Περιβάλλον ανάπτυξης εφαρμογών για φορητές συσκευές με Android

Οι εφαρμογές που αναπτύσσονται για φορητές συσκευές είναι πολλών κατηγοριών: παιχνίδια, ψυχαγωγίας, κοινωνικής δικτύωσης, επικοινωνίας, εκπαιδευτικές, ενημέρωσης, ηλεκτρονικού εμπορίου κ.ά. Οι χρήστες μπορούν να κατεβάσουν τις εφαρμογές της προτίμησής τους, κάποιες δωρεάν και κάποιες άλλες επί πληρωμή, από ηλεκτρονικά καταστήματα, για παράδειγμα το Google Play για το Android, το App Store για το iOS και το Windows Phone Store για το Windows Phone. Οι επαγγελματίες ή ερασιτέχνες προγραμματιστές ανεβάζουν και διαθέτουν τις εφαρμογές τους στα παραπάνω ηλεκτρονικά καταστήματα.

Εικόνα 7.3. Το ηλεκτρονικό κατάστημα Google play

Το εκπαιδευτικό περιβάλλον ανάπτυξης εφαρμογών App Inventor

Η ανάγκη για εύκολη ανάπτυξη εφαρμογών για φορητές συσκευές με Android οδήγησε αρχικά τα εργαστήρια της Google στη δημιουργία του App Inventor, ενός ελεύθερου, διαδικτυακού και

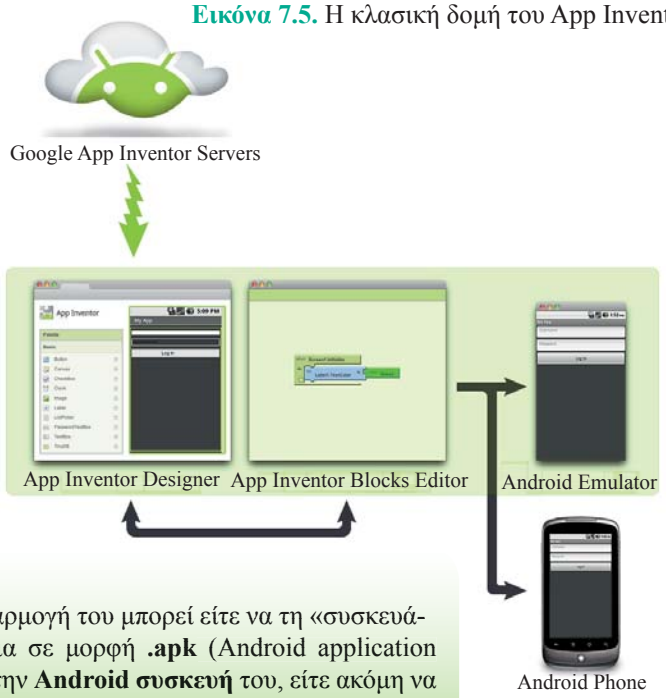
Το **Android** είναι ένα δημοφιλές, ελεύθερο και ανοικτού κώδικα (open source) Λειτουργικό Σύστημα για φορητές συσκευές. Βασίζεται στον πυρήνα του Linux. Το πρώτο κινητό που κυκλοφόρησε με Android έφτασε στα ράφια των καταστημάτων στις 22 Οκτωβρίου 2008.

οπτικού προγραμματιστικού περιβάλλοντος με πλακίδια (blocks), όπως και το Scratch. Στη συνέχεια, το γνωστό κορυφαίο αμερικάνικο πανεπιστήμιο MIT (Massachusetts Institute of Technology) ανέλαβε την ανάπτυξη και συντήρηση αυτού. Ακόμα και ένας αρχάριος χρήστης μπορεί να συνδεθεί στο App Inventor και με διαδικασία «σύρε και άφησε» να συνδυάσει πλακίδια και να αναπτύξει εφαρμογές για φορητές συσκευές με Android, το οποίο επίσης κατασκεύασε η Google βασισμένη στο ελεύθερο κι ανοικτό λειτουργικό σύστημα για υπολογιστές Linux. Τα πλακίδια ενώνονται μόνο όταν προκύπτει συντακτικά σωστό πρόγραμμα, και η τελική εφαρμογή μπορεί να εκτελεστεί και να δοκιμαστεί είτε απευθείας σε συσκευή που είναι συνδεδεμένη με τον υπολογιστή του χρήστη (ενσύρματα με USB ή ασύρματα με WiFi) είτε σε ενσωματωμένο emulator (προσομοιωτή κινητού τηλεφώνου).

Η κλασική δομή του περιβάλλοντος του App Inventor (εικόνα 7.5) αποτελείται από: (α) τον **Designer** (Σχεδιαστή), όπου ο χρήστης επιλέγει τα συστατικά μέρη για την εφαρμογή που αναπτύσσει, και (β) τον **Blocks Editor** (Συντάκτη πλακιδίων), όπου ο χρήστης συνδυάζει οπτικά τα πλακίδια του προγράμματος, για να ορίσει τη συμπεριφορά των μερών της εφαρμογής (μοιάζει με τη συναρμολόγηση ενός πάζλ). Τα πλακίδια είναι ταξινομημένα σε διαφορετικά χρώματα ανάλογα με τη λειτουργία που επιτελούν.

Εικόνα 7.4. Κινητό τηλέφωνο με Android

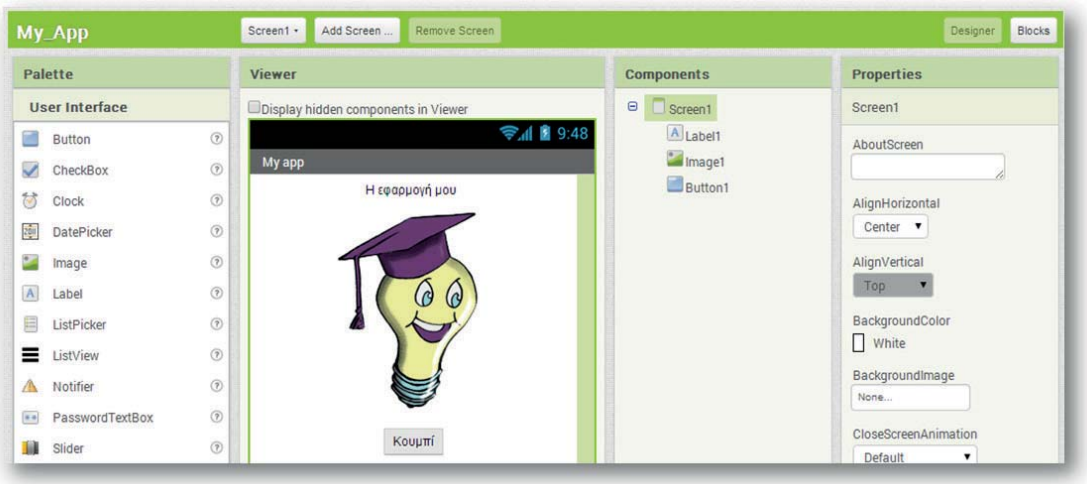
Εικόνα 7.5. Η κλασική δομή του App Inventor



Όταν ο χρήστης ολοκληρώσει την εφαρμογή του μπορεί είτε να τη «συσκευάσει», για να παραγάγει το τελικό πρόγραμμα σε μορφή **.apk** (Android application package), προκειμένου να το εγκαταστήσει στην **Android συσκευή** του, είτε ακόμη να το διανείμει δωρεάν ή εμπορικά στο Google Play. Εναλλακτικά, αν δεν υπάρχει διαθέσιμη κάποια συσκευή Android, ο χρήστης έχει τη δυνατότητα να δημιουργήσει και να ελέγξει τη λειτουργία της εφαρμογής του, χρησιμοποιώντας τον προσομοιωτή **Android Emulator**, ο οποίος είναι λογισμικό που εκτελείται τοπικά στον υπολογιστή του και συμπεριφέρεται σαν ένα κινητό τηλέφωνο.

Διαδικασία δημιουργίας μιας εφαρμογής στο App Inventor

1. Αρχικά, επισκεπτόμαστε τον επίσημο ιστότοπο του App Inventor, ο οποίος περιέχει υλικό στην αγγλική γλώσσα με υποστηρικτικές οδηγίες, οδηγούς εκμάθησης, βιβλιοθήκες, ομάδες συζητήσεων κ.ά.
2. Για να έχουμε δικαίωμα πρόσβασης στο προγραμματιστικό περιβάλλον, θα πρέπει να διαθέτουμε λογαριασμό στην Google. Για όσους δεν έχουν λογαριασμό, η εγγραφή είναι εύκολη και δωρεάν. Επιλέγουμε τον σύνδεσμο **Create** και στο παράθυρο που μας ανοίγει κάνουμε είσοδο με τα στοιχεία του λογαριασμού μας.
3. Στην αρχική οθόνη που εμφανίζεται επιλέγουμε **New Project** (νέο έργο), οπότε και μας ζητείται να δώσουμε ένα όνομα για την εφαρμογή που πρόκειται να δημιουργήσουμε.
4. Ανοίγει η καρτέλα **Designer** (εικόνα 7.6), για να σχεδιάσουμε την εμφάνιση της εφαρμογής μας επιλέγοντας τα απαραίτητα συστατικά στοιχεία και ορίζοντας ιδιότητες γι' αυτά.

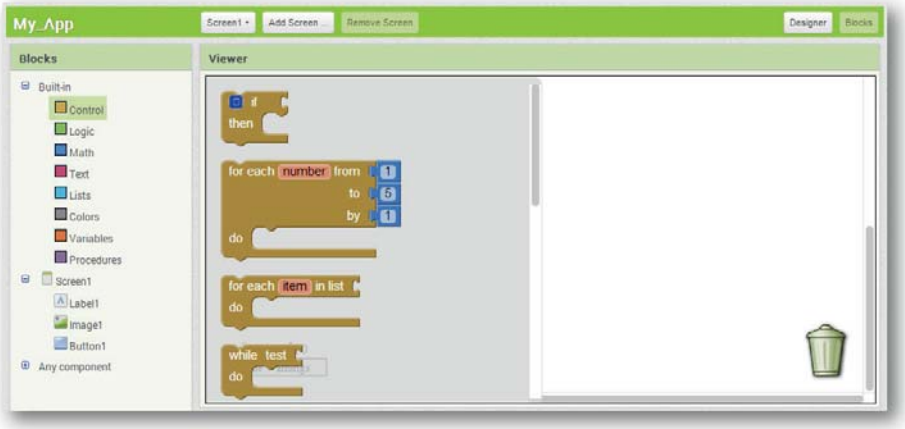


Εικόνα 7.6. App Inventor Designer

Ο Designer αποτελείται από τα παρακάτω κύρια πλαίσια:

- **Palette (συλλογή συστατικών)**: περιέχει όλα τα στοιχεία, χωρισμένα σε κατηγορίες (User Interface, Layout, Media κ.ά.) που μπορούμε να εισάγουμε στην εφαρμογή μας με απλό σύρσιμο.
- **Viewer (οθόνη συσκευής)**: εδώ τοποθετούμε στη θέση που θέλουμε τα συστατικά στοιχεία της εφαρμογής με τη διαδικασία «σύρε και άφησε» από το πλαίσιο Palette.
- **Components (επιλεγμένα συστατικά)**: μια δενδροειδής δομή των στοιχείων που έχουμε επιλέξει.
- **Properties (ιδιότητες)**: το πλαίσιο παραμετροποίησης του κάθε συστατικού (π.χ. χρώμα, μέγεθος, συμπεριφορά).

5. Μόλις ολοκληρώσουμε τη σχεδίαση της εφαρμογής μας και την παραμετροποίηση των συστατικών της μέσω των ιδιοτήτων τους, ανοίγουμε την καρτέλα **Blocks** (εικόνα 7.7). Ο προγραμματισμός γίνεται στο πλαίσιο **Viewer**, όπου σύρουμε από το πλαίσιο **Blocks** τα κατάλληλα πλακίδια και τα συνδυάζουμε, για να ορίσουμε τις συμπεριφορές και τις συσχετίσεις της εφαρμογής μας. Τα πλακίδια είναι χρωματιστά και χωρίζονται σε δύο κατηγορίες: τα ενσωματωμένα (Built-in), που ορίζουν γενικές συμπεριφορές στην εφαρμογή μας, και τα σχετικά με συγκεκριμένα συστατικά της εφαρμογής που ορίζουν συμπεριφορές γι' αυτά.



Εικόνα 7.7. App Inventor Blocks Editor

6. Στο τελευταίο βήμα εγκαθιστούμε και ελέγχουμε την εφαρμογή μας με σύνδεση σε κάποια φορητή συσκευή (εικόνα 7.8). Επιλέγουμε από το μενού **Connect: AI Companion** για σύνδεση μέσω WiFi, με την απαραίτητη προϋπόθεση να το έχουμε πρώτα εγκαταστήσει στη συσκευή μας ή **USB** για ενσύρματη σύνδεση ή **Emulator** για προσομοίωση φορητής συσκευής στον υπολογιστή μας (εικόνα 7.9).

Εικόνα 7.8. Σύνδεση φορητής συσκευής

Εικόνα 7.9. Ο Emulator

Project με την εφαρμογή App Inventor

Στη συνέχεια θα δημιουργήσουμε μια ολοκληρωμένη εφαρμογή ακολουθώντας τις φάσεις του κύκλου ζωής εφαρμογών, όπως τις μελετήσαμε στο κεφάλαιο 5. Υποθέτουμε ότι εργαζόμαστε στην εταιρεία «ΛΑΜΔΑ Software Production» που παράγει προγράμματα και εφαρμογές σε διάφορες γλώσσες προγραμματισμού.

Φάση 1η: Ανάλυση

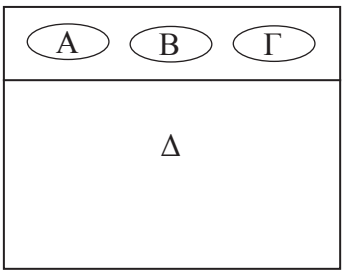
Ένας πελάτης της εταιρείας μάς ζητάει να φτιάξουμε μια εφαρμογή για φορητές συσκευές που λειτουργούν με λειτουργικό σύστημα Android. Η εφαρμογή απαιτείται να είναι πρωτότυπη, για να προκαλέσει το ενδιαφέρον των εφήβων–μαθητών στους οποίους απευθύνεται. Προτιμάμε να γίνει η υλοποίησή της με το περιβάλλον App Inventor.

Ζητείται η εφαρμογή να έχει ένα κεντρικό μενού με 3 επιλογές. Η πρώτη επιλογή να ξεκινάει την εκτέλεση προστασίας της οθόνης, η δεύτερη επιλογή να περιέχει την εκτέλεση ενός παιχνιδιού και η τελευταία επιλογή να υπολογίζει τον Μέσο Όρο ενός μαθήματος.

Φάση 2η: Σχεδίαση

Η ομάδα των προγραμματιστών της εταιρείας συνεδριάζει και καταλήγει στην παρακάτω πρόταση προς τον πελάτη.

Στο σχήμα 7.1 παρουσιάζεται το σχέδιο της διεπαφής χρήσης του κινητού. Τα Α, Β, Γ είναι τα κουμπιά για τις 3 επιλογές και το Δ είναι ο χώρος, όπου θα εμφανίζονται τα αποτελέσματα από την εκτέλεση του προγράμματος που αντιστοιχεί σε κάθε επιλογή. Συγκεκριμένα:



Σχήμα 7.1. Σχεδίαση οθόνης κινητού

- √ Όταν πατηθεί το κουμπί Α, εκτελείται η προστασία της οθόνης, όπου εμφανίζεται μια εικόνα ενός ήρεμου σκύλου, και, όταν ο χρήστης αγγίξει την περιοχή (Δ), τότε αλλάζει η εικόνα του σκύλου σε αγριεμένο και ακούγεται ο ανάλογος ήχος.
- √ Όταν πατηθεί το κουμπί Β, εκτελείται το παιχνίδι. Ο χρήστης θα έχει τη δυνατότητα να ζωγραφίζει στην οθόνη του κινητού του.
- √ Όταν πατηθεί το κουμπί Γ, υπολογίζεται ο Μέσος Όρος του μαθήματος και εμφανίζεται η προαγωγή ή απόρριψη του μαθητή στο συγκεκριμένο μάθημα.

Αναλυτικότερα, η ομάδα σχεδίασε τα παρακάτω πλαίσια (σενάρια εντολών), ώστε στη συνέχεια να τα υλοποιήσει στην επόμενη φάση.

(Α) Για την πρώτη επιλογή προστασίας της οθόνης έχουμε τα παρακάτω σενάρια-ψευδοκώδικα:

Όταν το κουμπί Α πατηθεί, τότε

απόκρυψε ό,τι δεν αφορά στον σκύλο και εμφάνισε τον ήρεμο σκύλο.

Όταν ο χρήστης αγγίξει την περιοχή Δ, τότε

άλλαξε την εικόνα του σκύλου σε αγριεμένο και παίξε ήχο γαβγίσματος

Όταν ο χρήστης πάψει να αγγίζει την περιοχή Δ, τότε

άλλαξε την εικόνα του σκύλου σε ήρεμο.

(Β) Για τη δεύτερη επιλογή του παιχνιδιού σχεδίασης έχουμε τα παρακάτω σενάρια-ψευδοκώδικα:

Όταν το κουμπί Β πατηθεί,τότε

απόκρυψε ό,τι δεν αφορά στη ζωγραφική και καθάρισε την περιοχή Δ

Όταν ο χρήστης αγγίξει την οθόνη, τότε

ζωγράφισε σε εκείνη τη θέση μια τελεία

Όταν ο χρήστης κινήσει το δάκτυλο του επάνω στην οθόνη, τότε

ζωγράφισε μια γραμμή μεταξύ της τρέχουσας θέσης του δακτύλου και της προηγούμενης.

Για να μπορέσουμε όμως να σχεδιάσουμε κάτι άλλο από την αρχή, θα πρέπει να καθαρίσουμε την οθόνη.

Όταν κουνηθεί η φορητή συσκευή, τότε
καθάρισε την οθόνη.

(Γ) Για την τρίτη επιλογή όπου υπολογίζουμε τον μέσο όρο ενός μαθήματος και εμφανίζεται στην οθόνη το αποτέλεσμα της προαγωγής του μαθητή, έχουμε τα παρακάτω σενάρια:

Ορίζουμε και αρχικοποιούμε τις μεταβλητές:

- Α (ο προφορικός βαθμός του Α τετραμήνου σε ένα μάθημα) ← 0 (μηδέν)
- Β (ο προφορικός βαθμός του Β τετραμήνου στο ίδιο μάθημα) ← 0 (μηδέν)
- Γ (ο γραπτός βαθμός στο ίδιο μάθημα) ← 0 (μηδέν)
- ΜΟ (ο μέσος όρος βαθμολογίας του μαθήματος) ← 0 (μηδέν)

Επιλέγουμε να γίνουν οι υπολογισμοί και η εμφάνιση των αποτελεσμάτων με τη χρήση διαδικασιών.

Όταν το κουμπί Γ πατηθεί, τότε

απόκρυψε το σκύλο,
καθάρισε την περιοχή Δ

εμφάνισε την ετικέτα οδηγιών και το πλαίσιο όπου θα εισάγω τους βαθμούς

εμφάνισε το κουμπί «Τελικό αποτέλεσμα»

Όταν πατήσω το κουμπί «Τελικό αποτέλεσμα»,

κάλεσε τη διαδικασία για τον υπολογισμό του μέσου όρου

κάλεσε τη διαδικασία για την εμφάνιση των αποτελεσμάτων

Διαδικασία: Υπολογισμός του μέσου όρου του μαθήματος
Υπολόγισε τον μέσο όρο των 2 προφορικών βαθμών ((Α+Β) / 2) και καταχώρισέ τον στον ΜΟ.

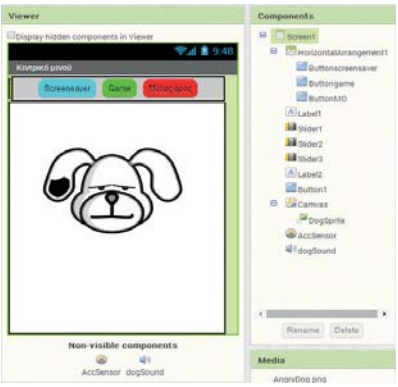
Υπολόγισε τον μέσο όρο ((ΜΟ+Γ) / 2) και καταχώρισέ τον στον ΜΟ.

Διαδικασία: Εμφάνιση των αποτελεσμάτων
Εμφάνισε τον μέσο όρο του μαθήματος.
Αν ο ΜΟ είναι μεγαλύτερος ή ίσος από 10, τότε
    εμφάνισε ότι ο μαθητής πέρασε το μάθημα
αλλιώς
    εμφάνισε ότι ο μαθητής δεν πέρασε το μάθημα.
Τέλος_αν

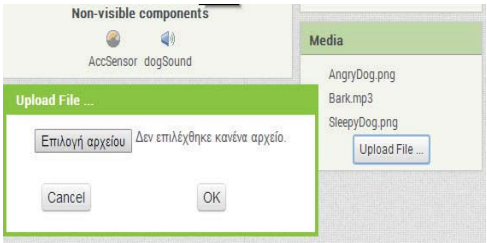
Φάση 3η: Υλοποίηση

Ακολουθούμε τα βήματα δημιουργίας μιας εφαρμογής, όπως περιγράφονται σε προηγούμενη παράγραφο, και δημιουργούμε ένα νέο project με όνομα «Fun & Learn». Βρισκόμαστε στο περιβάλλον εργασίας Designer (σχεδίασης) του App Inventor. Απ' όλη την παραπάνω περιγραφή καταλαβαίνουμε ότι θα χρησιμοποιήσουμε 2 εξωτερικά αρχεία εικόνων του σκύλου και ένα ήχου (γάβγισμα), οπότε, χρησιμοποιώντας το κουμπί Upload File του πλαισίου Media (Εικόνα 7.10), ανεβάζουμε τα σχετικά αρχεία (Προσοχή: το συνολικό μέγεθος των αρχείων δεν πρέπει να υπερβαίνει τα 5 MB, διότι τότε δεν δημιουργείται εκτελέσιμο αρχείο .apk).

Στη συνέχεια εισάγουμε τα παρακάτω στοιχεία στο αντικείμενο Screen1 του πλαισίου Viewer. Αλλάζουμε τις ρυθμίσεις για κάθε αντικείμενο όπως στον πίνακα 7.1. Η τελική μορφή μετά από αυτή την ενέργεια θα πρέπει να είναι αυτή που φαίνεται στην εικόνα 7.11.



Εικόνα 7.11. Τα πλαίσια Viewer και Components μετά την εισαγωγή των αντικειμένων



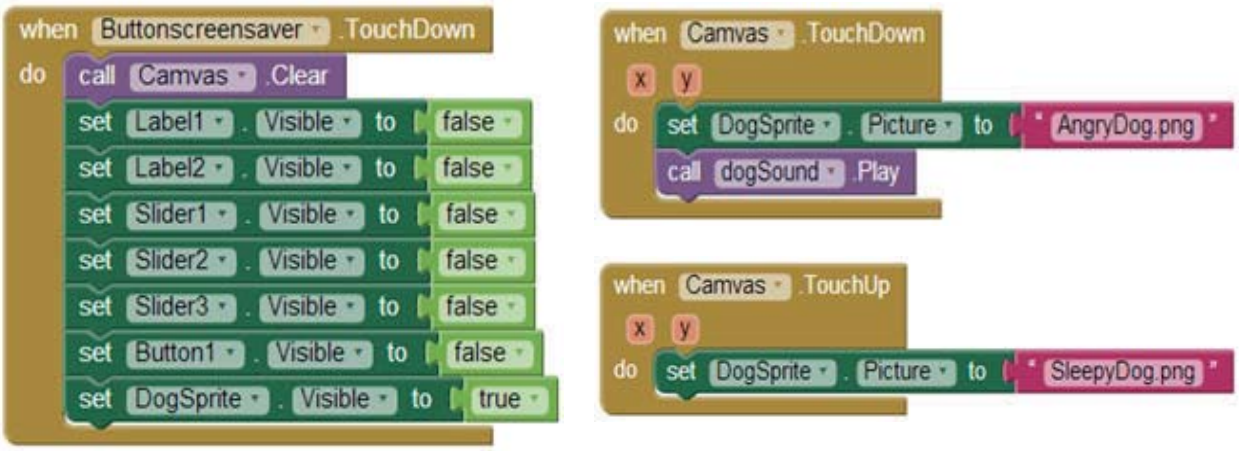
Εικόνα 7.10. Ανέβασμα εξωτερικών αρχείων που θα χρησιμοποιηθούν

Πίνακας 7.1 Τα συστατικά μέρη της εφαρμογής και οι ιδιότητες τους

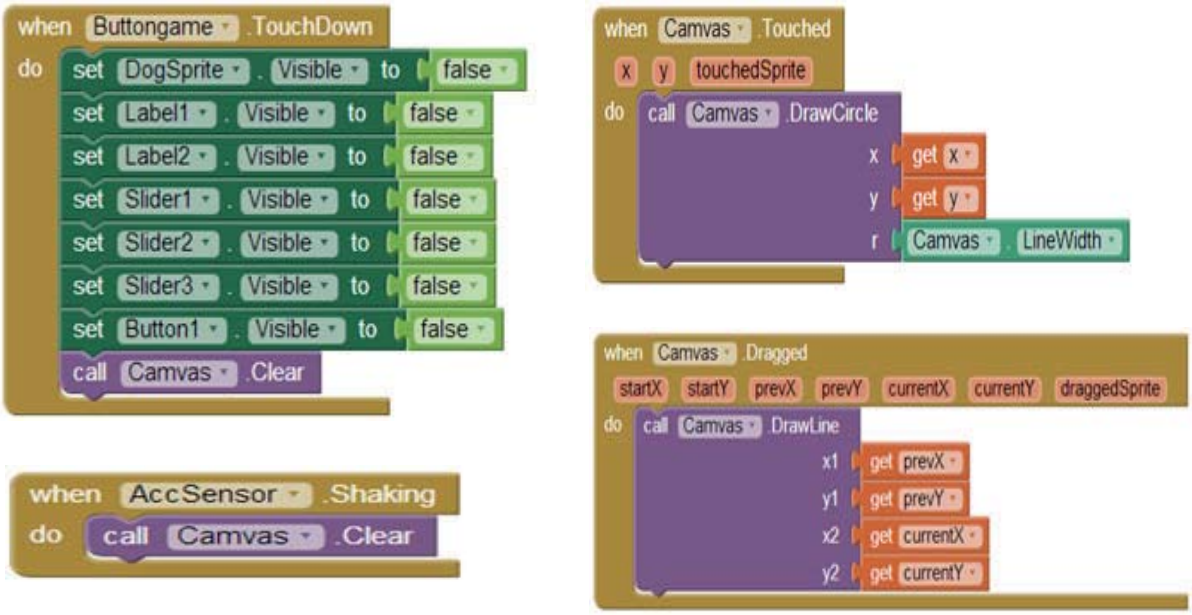
Από την **ομάδα** του πλαισίου **Palette**	Επιλέγουμε το **αντικείμενο** και το μεταφέρουμε στο πλαίσιο **Viewer**	Αλλάζουμε το **Όνομα** με το κουμπί **Rename**	Μεταβάλλουμε τις ιδιότητες - **Properties**
	Screen1 (Είναι ήδη εγκατεστημένο, οπότε αλλάζουμε μόνο τις ιδιότητες)		BackgroundColor: LightGrey Screen Orientation: Portrait Scrollable: (No) Title:Κεντρικό μενού
Layout	Horizontal Arrangement	Horizontal Arrangement1	PhotoArrangement Width: Fill Parent AlignHorizontal: Center
User Interface	Button (το μεταφέρουμε μέσα στο πλαίσιο Horizontal Arrangement)	Buttonscreensaver	BackgroundColor: Cyan Shape: Rounded Text: Screensaver
User Interface	Button (το μεταφέρουμε μέσα στο πλαίσιο Horizontal Arrangement)	Buttongame	BackgroundColor: Green Shape: Rounded Text: Game
User Interface	Button (το μεταφέρουμε μέσα στο πλαίσιο Horizontal Arrangement)	ButtonMO	BackgroundColor: Red Shape: Rounded Text: Μέσος Όρος
User Interface	Label	Label1	Visible: Hidden
User Interface	Slider	Slider1	MaxValue: 20.0 MinValue: 1.0 ThumbPosition: 5 Visible: hidden Width: Fill parent
User Interface	Slider	Slider2	MaxValue: 20.0 MinValue: 1.0 ThumbPosition: 10 Visible: hidden Width: Fill parent
User Interface	Slider	Slider3	MaxValue: 20.0 MinValue: 1.0 ThumbPosition: 15 Visible: hidden Width: Fill parent
User Interface	Label	Label2	Visible: Hidden
User Interface	Button	Button1	Text: Τελικό αποτέλεσμα TextAlignment: center Visible: hidden Width: Fill parent
Drawing and Animation	Canvas	Canvas	Paint Color: Blue Width: Fill Parent Height: Fill Parent
Sensors	Accelerometer Sensor	AccSensor	
Drawing and Animation	ImageSprite	DogSprite	Interval: 10 Picture: SleepyDog.jpg Rotates: (No) Visible: (Yes)
Media	Sound	DogSound	Source: Bark.mp3 MinimumInterval: 300

Στη συνέχεια επιλέγουμε από πάνω δεξιά το κουμπί Blocks και μεταφερόμαστε στο περιβάλλον εργασίας όπου προγραμματίζουμε (App Inventor Blocks Editor). Δημιουργούμε τα παρακάτω σενάρια (blocks εντολών). Συγκεκριμένα, για να προγραμματίσουμε για ένα αντικείμενο, το επιλέγουμε από το πλαίσιο Blocks και από το συρτάρι εντολών που ανοίγει επιλέγουμε την εντολή και τη μεταφέρουμε στο πλαίσιο Viewer. Το περιβάλλον μάς βοηθάει να αποφύγουμε συντακτικά λάθη, μιας και σε αυτή την περίπτωση δεν «κουμπώνουν» οι εντολές μεταξύ τους.

(Α) Δημιουργούμε τα παρακάτω σενάρια εντολών για την επιλογή προστασίας της οθόνης.



(Β) Δημιουργούμε τα παρακάτω σενάρια για την επιλογή του παιχνιδιού σχεδίασης. Όπου ακουμπάει ο χρήστης ζωγραφίζει μια κουκκίδα και, όταν σύρει το δάκτυλο, ζωγραφίζει γραμμή.



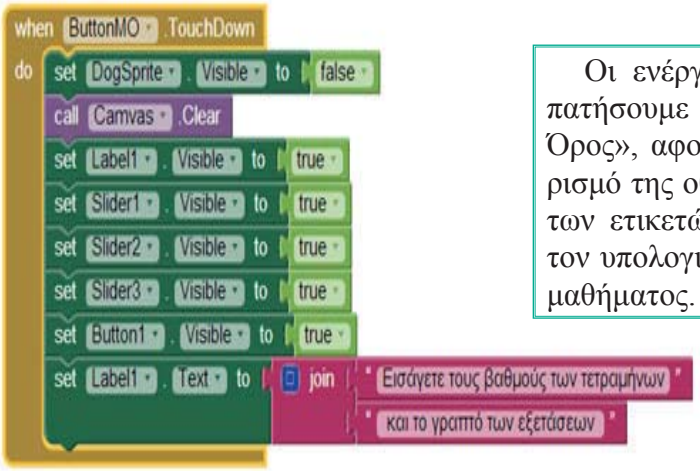
Για να μπορούμε όμως να σχεδιάσουμε κάτι άλλο από την αρχή, θα πρέπει να καθαρίσουμε την οθόνη. Αυτό γίνεται, αν κουνήσουμε τη συσκευή.

(Γ) Δημιουργούμε τα παρακάτω σενάρια για την επιλογή του υπολογισμού του Μέσου Όρου (ΜΟ) και των αποτελεσμάτων προαγωγής του μαθητή σε ένα μάθημα.

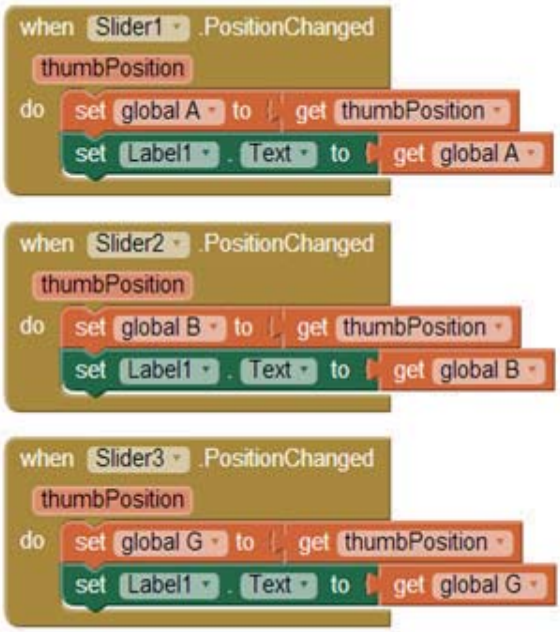
Αντιστοίχιση των μεταβλητών

Βαθμός Α τετραμήνου - A
Βαθμός Β τετραμήνου – B
Βαθμός γραπτού - G
Μέσος όρος μαθήματος - mo

Δημιουργία και αρχικοποίηση των μεταβλητών A, B, G, mo.

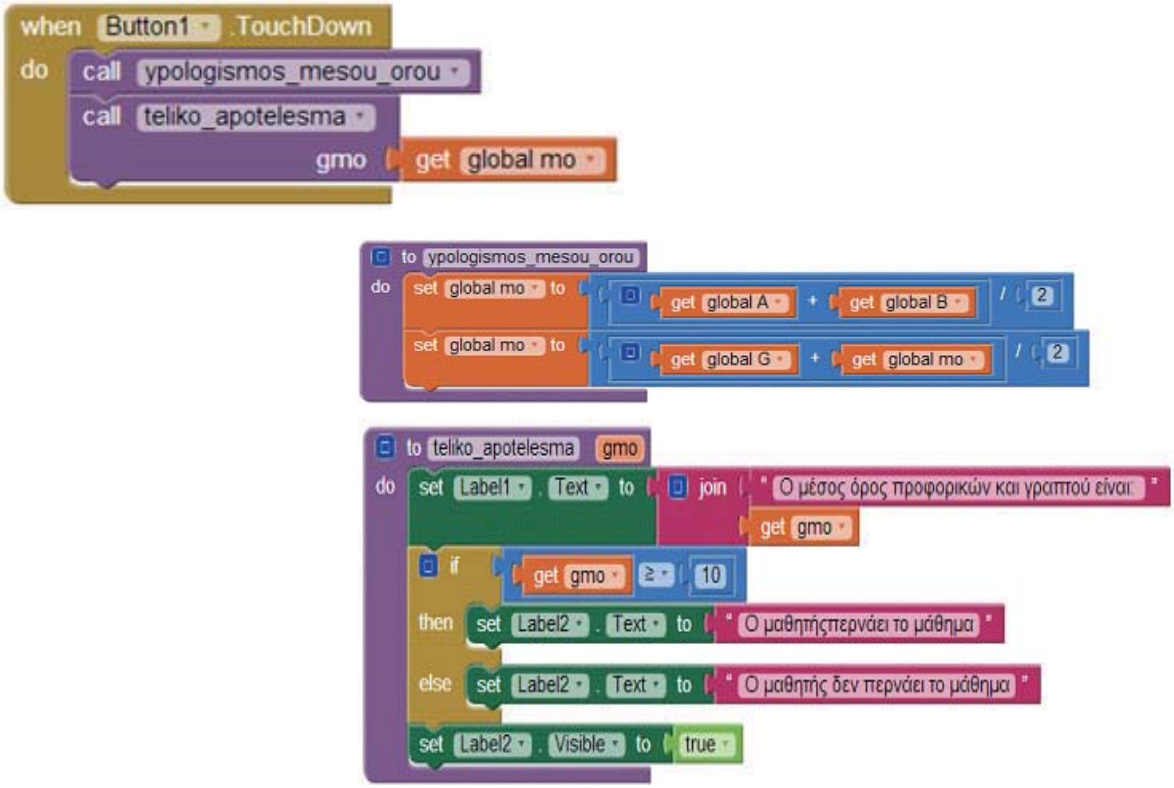


Οι ενέργειες που κάνουμε, όταν πατήσουμε το κουμπί «Μέσος Όρος», αφορούν κυρίως στον καθαρισμό της οθόνης και στην εμφάνιση των ετικετών και του κουμπιού για τον υπολογισμό του μέσου όρου του μαθήματος.



Με τον διπλανό κώδικα επιτυγχάνουμε, μετακινώντας την μπάρα πάνω σε μια κλίμακα από 1 έως 20, να ενημερώνονται οι αντίστοιχες μεταβλητές που αφορούν στους 3 βαθμούς του μαθήματος:
Slider1 για τη μεταβλητή A
Slider2 για τη μεταβλητή B
Slider3 για τη μεταβλητή G

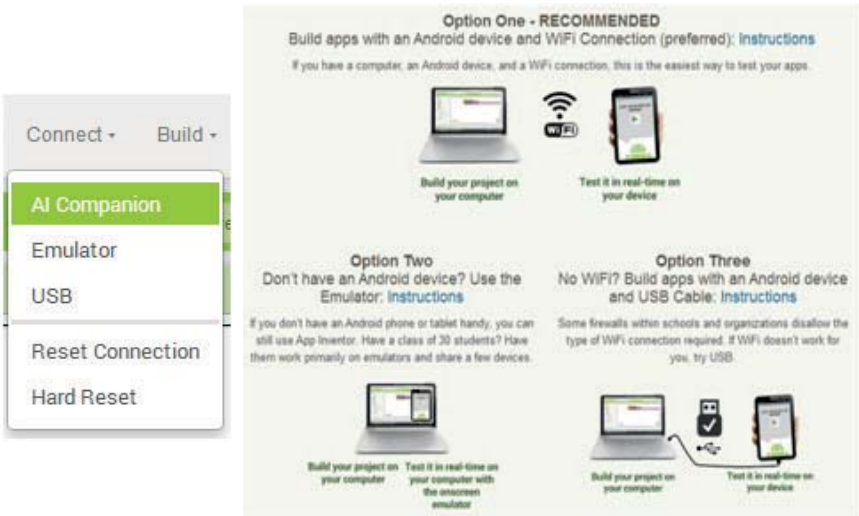
Όταν πατήσουμε το κουμπί «Τελικό αποτέλεσμα», τότε καλούμε τις Procedures (διαδικασίες) για τον υπολογισμό του μέσου όρου του μαθήματος και για την εμφάνιση του τελικού αποτελέσματος προαγωγής ή απόρριψης του μαθητή στο μάθημα. Αξίζει να σημειωθεί ότι κατά την κλήση της διαδικασίας «teliko_apotelesma» περνάμε ως όρισμα τη μεταβλητή mo, η οποία μεταβιβάζει την τιμή της στην gmo.



Φάση 4η: Λειτουργία

Οποιαδήποτε στιγμή μπορούμε να κάνουμε έλεγχο της λειτουργίας της εφαρμογής μας, για να εντοπίσουμε πιθανά λάθη και να επιστρέψουμε στη φάση της υλοποίησης ή ακόμη και της σχεδίασης για διορθώσεις ή και βελτιώσεις.

Επιλέγουμε από το μενού **Connect** όποια από τις επιλογές μπορούμε να δοκιμάσουμε. Όπως φαίνεται και στην εικόνα 7.12, αν επιλέξουμε: 1) **AI Companion**, μας δίνεται η δυνατότητα να εκτελέσουμε τον κώδικα στη φορητή συσκευή με WiFi σύνδεση, 2) **Emulator**, βλέπουμε τα αποτελέσματα της εκτέλεσης με τη χρήση προσομοιωτή στην οθόνη του υπολογιστή και 3) **USB,** μας δίνεται η δυνατότητα να εκτελέσουμε τον κώδικα στη φορητή συσκευή με ενσύρματη σύνδεση μέσω USB θύρας.



Εικόνα 7.12. Οι τρόποι σύνδεσης για την εκτέλεση της εφαρμογής σε φορητές συσκευές ή προσομοίωση στον υπολογιστή (http://appinventor.mit.edu/explore/ai2/setup.html)

Φάση 5η: Συντήρηση

Κατά τη φάση αυτή είναι δυνατό να γίνουν προτάσεις και από τους χρήστες για αλλαγές και βελτιώσεις της εφαρμογής. Για παράδειγμα, ορισμένες προτάσεις θα μπορούσαν να είναι:

1. Να κινείται ο σκύλος επάνω στην οθόνη.
2. Να γίνεται λήψη μιας φωτογραφίας από την κάμερα του κινητού και ο χρήστης να ζωγραφίζει πάνω στην εικόνα αυτή.
3. Να προστεθούν δυνατότητες αποθήκευσης και εκτύπωσης της ζωγραφιάς.
4. Να έχουμε δυνατότητα επιλογής χρώματος.
5. Να δημιουργηθεί μια λίστα με τα μαθήματα της τάξης, ώστε να φαίνεται και ο τίτλος του μαθήματος.
6. Να ενημερώσουμε την εφαρμογή σε οποιαδήποτε περίπτωση αλλαγής του τρόπου υπολογισμού του μέσου όρου.

Εικόνα 7.13. Διεπαφή χρήσης της εφαρμογής

Ερωτήσεις - Δραστηριότητες:

1. Επεκτείνατε και εμπλουτίστε την εφαρμογή, υλοποιώντας τις προτάσεις που αναφέρονται στη φάση της συντήρησης.
2. Ποια εφαρμογή πιστεύετε ότι θα μπορούσατε να φτιάξετε με την ομάδα σας; Συμβουλευτείτε τον καθηγητή σας και υλοποιήστε τη.

7.2 Αντικειμενοστρεφής προγραμματισμός σε 3D περιβάλλον

Αντικειμενοστρεφής προγραμματισμός

Με το πέρασμα των χρόνων τα προγράμματα γίνονται μεγαλύτερα σε μέγεθος και πιο πολύπλοκα σε δομή και λειτουργίες. Επίσης, ο προσδιορισμός των απαιτήσεων και η συντήρηση του λογισμικού δυσκολεύει. Ο **αντικειμενοστρεφής προγραμματισμός (object-oriented programming)** αποτελεί μια διαδεδομένη προσέγγιση για δημιουργία προγραμμάτων, η οποία προσφέρει καλύτερη αντιμετώπιση των παραπάνω προβλημάτων. Αντικειμενοστρεφής είναι ο χαρακτηρισμός που σημαίνει «στραμμένος (προσανατολισμένος) σε αντικείμενα». Όταν κάποιος προγραμματίζει με αντικειμενοστρεφή τρόπο, διασπά ένα πρόβλημα στα συστατικά του στοιχεία. Κάθε στοιχείο μετατρέπεται σε ένα αυτοτελές **αντικείμενο (object)**, το οποίο περιέχει τις δικές του εντολές και τα δεδομένα που σχετίζονται με αυτό το αντικείμενο. Με αυτή τη διαδικασία μειώνεται η πολυπλοκότητα και γίνεται ευκολότερος ο χειρισμός των μεγάλων προγραμμάτων.

Μια **κλάση (class)** είναι ένα πρότυπο (καλούπι) που χρησιμοποιείται για τη δημιουργία ενός αντικειμένου. Κάθε αντικείμενο που δημιουργείται από την ίδια κλάση έχει παρόμοια, αν όχι ίδια, χαρακτηριστικά. Ένα αντικείμενο αποτελεί ένα μοναδικό και συγκεκριμένο **στιγμιότυπο (instance)** της κλάσης στην οποία

Κληρονομικότητα: η διεργασία μέσω της οποίας μια κλάση μπορεί να αποκτήσει (κληρονομήσει) τις ιδιότητες και μεθόδους μιας άλλης κλάσης. Έτσι δημιουργείται μια ιεραρχική ταξινόμηση. Π.χ. κλάση **Φρούτο**, υποκλάση **Μήλο** και υποκλάση **Φιρίκι** (Ελληνική ποικιλία). Επειδή το φιρίκι έχει κληρονομήσει όλα τα ποιοτικά χαρακτηριστικά των φρούτων, χρειάζεται να ορίσουμε γι' αυτό μόνο τα χαρακτηριστικά που το κάνουν μοναδικό. Άλλο παράδειγμα, κλάση **Μέσο μεταφοράς**, υποκλάση **Όχημα**, υποκλάση **Αυτοκίνητο**.