ΥΠΟΥΡΓΕΙΟ ΠΑΙΔΕΙΑΣ, ΕΡΕΥΝΑΣ
ΚΑΙ ΘΡΗΣΚΕΥΜΑΤΩΝ
ΙΝΣΤΙΤΟΥΤΟ ΕΚΠΑΙΔΕΥΤΙΚΗΣ
ΠΟΛΙΤΙΚΗΣ

Γ. Πανσεληνάς, Ν. Αγγελιδάκης,
Α. Μιχαηλίδη, Χ. Μπλάτσιος,
Σ. Παπαδάκης, Γ. Παυλίδης,
Ε. Τζαγκαράκης, Α. Τζωρμπατζάκης

Εφαρμογές Πληροφορικής

Α' ΓΕΝΙΚΟΥ ΛΥΚΕΙΟΥ

ΤΟΜΟΣ 2ος

Ι. Τ. Υ. Ε. «ΔΙΟΦΑΝΤΟΣ»

ΣΤΟΙΧΕΙΑ ΑΡΧΙΚΗΣ ΕΚΔΟΣΗΣ

ΙΝΣΤΙΤΟΥΤΟ ΕΚΠΑΙΔΕΥΤΙΚΗΣ ΠΟΛΙΤΙΚΗΣ

Πρόεδρος: Γκλαβάς Σ.

ΓΡΑΦΕΙΟ ΕΡΕΥΝΑΣ ΣΧΕΔΙΑΣΜΟΥ ΚΑΙ ΕΦΑΡΜΟΓΩΝ Β΄

Προϊστάμενος: Μάραντος Π. Φ.

ΣΥΓΓΡΑΦΕΙΣ:

Νικόλαος Αγγελιδάκης, Εκπαιδευτικός Πληροφορικής (ΠΕ19) Δ/θμιας Εκπαιδευσης

Αφροδίτη Μιχαηλίδη, Εκπαιδευτικός Πληροφορικής (ΠΕ19) Δ/θμιας Εκπαιδευσης

Χαρίλαος Μπλάτσιος, Εκπαιδευτικός Πληροφορικής (ΠΕ19) Δ/θμιας Εκπαιδευσης

Γεώργιος Πανσεληνάς, Σχολικός Σύμβουλος Πληροφορικής (ΠΕ19) Δ/θμιας Εκπαιδευσης

Σταύρος Παπαδάκης, Εκπαιδευτικός Πληροφορικής (ΠΕ19) Δ/θμιας Εκπαιδευσης

Γεώργιος Παυλίδης, Εκπαιδευτικός Πληροφορικής (ΠΕ20) Δ/θμιας Εκπαιδευσης

Ελευθέριος Τζαγκαράκης, Διοικητικός υπάλληλος ΠΕ-Πληροφορικής του Υ.ΠΑΙ.Θ.

Αλέξης Τζωρμπατζάκης, Εκπαιδευτικός Πληροφορικής (ΠΕ19) Δ/θμιας Εκπαιδευσης

ΣΤΟΙΧΕΙΑ ΑΡΧΙΚΗΣ ΕΚΔΟΣΗΣ

ΕΠΙΜΕΛΕΙΑ ΣΥΝΤΟΝΙΣΜΟΣ ΟΜΑΔΑΣ:
Γεώργιος Πανσεληνάς, Σχολικός Σύμβουλος Πληροφορικής (ΠΕ19) Δ/θμιας Εκπαίδευσης

ΚΡΙΤΕΣ-ΑΞΙΟΛΟΓΗΤΕΣ:
Ιωάννης Μαυρίδης, Μέλος ΔΕΠ (συντονιστής)
Ζαχαρίας Μανουσαρίδης, Σχολικός Σύμβουλος, κλ. ΠΕ19
Φώτιος Λαζαρίνης, Εκπαιδευτικός, κλ. ΠΕ19

ΦΙΛΟΛΟΓΙΚΗ ΕΠΙΜΕΛΕΙΑ:
Ασημένια Χαρκιωτάκη, Εκπαιδευτικός Φιλόλογος (ΠΕ02) Δ/θμιας Εκπαίδευσης

ΚΑΛΛΙΤΕΧΝΙΚΗ ΕΠΙΜΕΛΕΙΑ:
Δέσποινα Αρβανίτη, Εκπαιδευτικός Πληροφορικής (ΠΕ20) Π/θμιας Εκπαίδευσης

ΕΙΚΟΝΑ ΕΞΩΦΥΛΛΟΥ:
Ελευθέριος Παναγουλόπουλος

«ΔΗΜΙΟΥΡΓΙΑ ΕΚΠΑΙΔΕΥΤΙΚΟΥ ΥΛΙΚΟΥ ΓΙΑ ΤΑ ΝΕΑ ΜΑΘΗΜΑΤΑ ΤΟΥ ΓΕΝΙΚΟΥ ΛΥΚΕΙΟΥ» της Πράξης «ΝΕΟ ΣΧΟΛΕΙΟ (ΣΧΟΛΕΙΟ 21ου αιώνα)-ΝΕΟ ΠΡΟΓΡΑΜΜΑ ΣΠΟΥΔΩΝ» ΜΕ ΚΩΔ. ΟΠΣ 295450, των Αξόνων Προτεραιότητας 1, 2 και 3 – ΟΡΙΖΟΝΤΙΑ ΠΡΑΞΗ του ΕΠΙΧΕΙΡΗΣΙΑΚΟΥ ΠΡΟΓΡΑΜΜΑΤΟΣ «ΕΚΠΑΙΔΕΥΣΗ ΚΑΙ ΔΙΑ ΒΙΟΥ ΜΑΘΗΣΗ», που συγχρηματοδοτείται από την Ευρωπαϊκή Ένωση - Ευρωπαϊκό Κοινωνικό Ταμείο και από Εθνικούς Πόρους (ΕΣΠΑ 2007 – 2013).

Η αξιολόγηση, η κρίση των προσαρμογών και η επιστημονική επιμέλεια του προσαρμοσμένου βιβλίου πραγματοποιείται από τη Μονάδα Ειδικής Αγωγής του Ινστιτούτου Εκπαιδευτικής Πολιτικής.
Η προσαρμογή του βιβλίου για μαθητές με μειωμένη όραση από το ΙΤΥΕ – ΔΙΟΦΑΝΤΟΣ πραγματοποιείται με βάση τις προδιαγραφές που έχουν αναπτυχθεί από ειδικούς εμπειρογνώμονες για το ΙΕΠ.

ΠΡΟΣΑΡΜΟΓΗ ΤΟΥ ΒΙΒΛΙΟΥ ΓΙΑ ΜΑΘΗΤΕΣ ΜΕ ΜΕΙΩΜΕΝΗ ΟΡΑΣΗ

ΙΤΥΕ - ΔΙΟΦΑΝΤΟΣ

Θεματική Ενότητα 2:

Προγραμματιστικά περιβάλλοντα -Δημιουργία εφαρμογών

- Κεφάλαιο 5: Κύκλος Ζωής Εφαρμογών
- Κεφάλαιο 6: Περιβάλλοντα Ανάπτυξης Εφαρμογών
- Κεφάλαιο 7: Υλοποίηση Εφαρμογών σε Προγραμματιστικά Περιβάλλοντα

ΚΕΦΑΛΑΙΟ 5

Κύκλος Ζωής Εφαρμογών

Διδακτικές ενότητες

5.1 Πρόβλημα και υπολογιστής
5.2 Ανάπτυξη εφαρμογών

Διδακτικοί στόχοι

Σκοπός του κεφαλαίου είναι οι μαθητές να κατανοήσουν τα βήματα που ακολουθούνται κατά την ανάπτυξη μιας εφαρμογής.

Οι μαθητές πρέπει να είναι σε θέση:

- √ να περιγράφουν τα βήματα αντιμετώπισης ενός προβλήματος.
- √ να αναγνωρίζουν τη χρησιμότητα του υπολογιστή στην επίλυση προβλημάτων.
- √ να περιγράφουν πώς από το πρόβλημα φτάνουμε στην εφαρμογή.

- √ να περιγράφουν τις φάσεις από τις οποίες αποτελείται ο κύκλος ζωής εφαρμογών.

Ερωτήματα

- √ Τι ονομάζουμε πρόβλημα;
- √ Πόσο χρήσιμος είναι και τι ρόλο παίζει ο υπολογιστής στην επίλυση προβλημάτων;
- √ Τι ονομάζουμε πρόγραμμα και τι εφαρμογή;
- √ Ποια βήματα ακολουθούμε για την ανάπτυξη μιας εφαρμογής;

Βασική ορολογία

Πρόβλημα, πρόγραμμα, εφαρμογή, κύκλος ζωής εφαρμογών

Εισαγωγή

Καθημερινά αντιμετωπίζουμε προβλήματα για τα οποία και αναζητούμε λύση τους. Θέλουμε να πραγματοποιούμε τις εργασίες μας

γρήγορα και αποδοτικά. Ο υπολογιστής μπορεί να βοηθήσει στην επίλυση πολλών προβλημάτων. Δεδομένα και προγράμματα εισάγονται σε έναν υπολογιστή, ο οποίος με τη σειρά του αναλαμβάνει την επεξεργασία των δεδομένων με βάση τις εντολές που περιέχονται στα προγράμματα, και στο τέλος εξάγονται τα αποτελέσματα της επεξεργασίας αυτής. Τα προγράμματα που χρησιμοποιούν οι χρήστες των υπολογιστών για να εκτελούν συγκεκριμένες εργασίες ονομάζονται εφαρμογές. Η ανάπτυξη των σύγχρονων εφαρμογών είναι μια απαιτητική εργασία, γι' αυτό και πραγματοποιείται ακολουθώντας μια συστηματική σειρά φάσεων που ονομάζεται κύκλος ζωής εφαρμογών.

5.1 Πρόβλημα και υπολογιστής

Η έννοια του προβλήματος

Η έννοια «πρόβλημα» συναντάται σε όλες σχεδόν τις εκφάνσεις της ζωής μας καθώς επίσης και σε όλους τους τομείς της Επιστήμης και της Τεχνολογίας. Σε κάποια προβλήματα που μας απασχολούν μπορούμε να βρούμε εύκολα τη λύση, ενώ σε άλλα πιο σύνθετα δυσκολευόμαστε ή και μερικές φορές αδυνατούμε να προσδιορίσουμε κάποια λύση.

Μερικά καθημερινά και σχετικά εύκολα παραδείγματα προβλημάτων αποτελούν η εύρεση της συντομότερης διαδρομής για ένα μέρος που επισκεπτόμαστε για πρώτη φορά, η αναζήτηση ενός βιβλίου στη βιβλιοθήκη, η αγορά ενός καινούριου υπολογιστή, η αναζήτηση

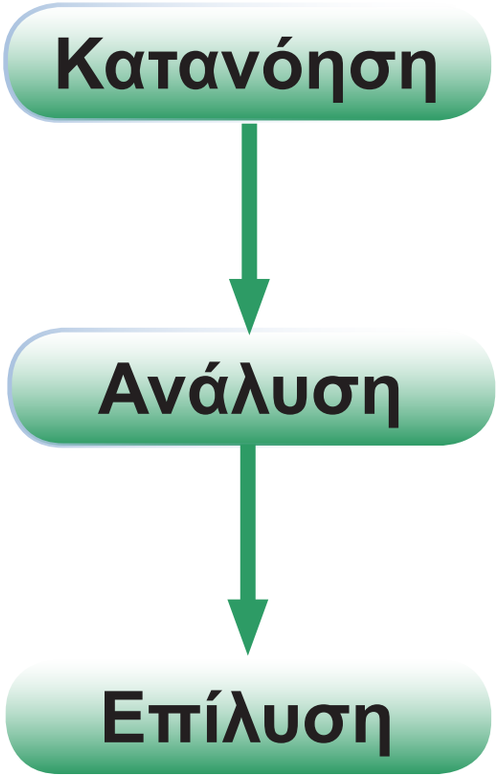
πληροφοριών για την εργασία μας σε κάποιο μάθημα, η επίλυση μιας εξίσωσης στο μάθημα των Μαθηματικών. Πιο σύνθετα παραδείγματα προβλημάτων αποτελούν η αντιμετώπιση της ρύπανσης του περιβάλλοντος, η εξοικονόμηση ενέργειας, η εξερεύνηση του διαστήματος.

Ο άνθρωπος αντιμετωπίζει προβλήματα από τα πρώτα βήματά του πάνω στη Γη. Για την επίλυσή τους κατασκευάζει διάφορα εργαλεία και μηχανισμούς. Ειδικότερα, για τα υπολογιστικά προβλήματα, η πορεία στον χρόνο ξεκίνησε από τον άβακα και τον μηχανισμό των Αντικυθήρων, για να καταλήξει στον ηλεκτρονικό υπολογιστή.

Πολλά προβλήματα είναι υπολογιστικά και απαιτούν για την επίλυσή τους λογικές σκέψεις και μαθηματικές πράξεις. Τέτοια προβλήματα είναι για παράδειγμα ο υπολογισμός της μισθοδοσίας ενός υπαλλήλου, ο υπολογισμός του μέσου όρου της βαθμολογίας ενός μαθητή, η επίλυση μιας δευτεροβάθμιας εξίσωσης, η πρόγνωση του καιρού με βάση μετεωρολογικά στοιχεία. Υπάρχουν προβλήματα που δεν μπορούμε να επιλύσουμε με τις υπάρχουσες γνώσεις μας, όπως για παράδειγμα η ακριβής πρόβλεψη των σεισμών. Τέλος, κάποια προβλήματα δεν επιλύονται, όπως ο τετραγωνισμός του κύκλου με κανόνα και διαβήτη.

Για την αντιμετώπιση των δύσκολων και σύνθετων προβλημάτων απαιτείται πρώτα η κατανόησή τους

με σαφή και πλήρη καταγραφή και αποσαφήνιση των δεδομένων και των ζητουμένων, έπειτα η ανάλυσή τους σε επιμέρους απλούστερα προβλήματα, και τέλος η εκτέλεση οργανωμένων βημάτων επίλυσης (σχήμα 5.1).



Σχήμα 5.1. Αντιμετώπιση προβλήματος

Γενικότερα, ως πρόβλημα θεωρούμε κάθε ζήτημα που τίθεται προς επίλυση, κάθε κατάσταση που μας απασχολεί και πρέπει να αντιμετωπιστεί. Η λύση ενός προβλήματος δεν μας είναι γνωστή ούτε προφανής.

Ο υπολογιστής και η επίλυση προβλημάτων

Ο υπολογιστής αποτελεί πλέον αναπόσπαστο κομμάτι των καθημερινών μας δραστηριοτήτων είτε αυτές αφορούν σε εργασία είτε σε ψυχαγωγία, και στην πραγματικότητα μάς βοηθάει στην επίλυση προβλημάτων. Ειδικά στα προβλήματα όπου έχουμε πολλά δεδομένα και ζητούμενα ή η μέθοδος επίλυσης είναι πολύπλοκη και ίσως βαρετή, ή η μέθοδος επίλυσης επαναλαμβάνεται πολλές φορές. Ας σκεφτούμε το παράδειγμα του υπολογισμού της

μισθοδοσίας μιας μεγάλης εταιρείας χωρίς χρήση υπολογιστή.

Ο υπολογιστής μπορεί να αποθηκεύσει μεγάλο πλήθος δεδομένων (αριθμούς, κείμενα, εικόνες, ήχο, βίντεο), εκτελεί υπολογισμούς και επεξεργάζεται δεδομένα ταχύτερα από τον άνθρωπο, και εκτελεί με πειθαρχία, συνέπεια και για όσες επαναλήψεις χρειαστεί μια λογική σειρά εντολών. Οι εντολές δίνονται στον υπολογιστή με τη μορφή προγραμμάτων. Ένα πρόγραμμα περιέχει εντολές (οδηγίες) που κατευθύνουν με κάθε λεπτομέρεια τον υπολογιστή, για να εκτελέσει μία συγκεκριμένη εργασία και να επιλύσει ένα πρόβλημα. Δίνουμε στον υπολογιστή δεδομένα για το πρόβλημα που θέλουμε να αντιμετωπίσουμε, αυτός τα επεξεργάζεται σύμφωνα με τις εντολές των

προγραμμάτων που εκτελεί, και στο τέλος μάς δίνει την απάντηση στο πρόβλημά μας.

 Ο υπολογιστής αποτελεί μια μηχανή επεξεργασίας δεδομένων. Στον υπολογιστή εισάγουμε δεδομένα με τη βοήθεια διάφορων συσκευών εισόδου (πληκτρολόγιο, σαρωτή κλπ.), τα οποία και επεξεργάζεται με τις κατάλληλες εντολές (οδηγίες) που του δίνουμε. Μετά την επεξεργασία, σε συσκευές εξόδου (οθόνη, εκτυπωτή κλπ.) παίρνουμε τις χρήσιμες πληροφορίες που θέλουμε.

Ας δούμε το παράδειγμα του υπολογισμού του μέσου όρου βαθμολογίας ενός μαθητή. Τα δεδομένα μας είναι ο αριθμός μητρώου του μαθητή, τα προσωπικά στοιχεία του

μαθητή (ονοματεπώνυμο, πατρώνυμο κ.λπ.) και η βαθμολογία του σε κάθε μάθημα. Στον υπολογιστή πρέπει να δώσουμε τα παραπάνω δεδομένα και το κατάλληλο πρόγραμμα, το οποίο θα περιέχει τις εντολές για τον υπολογισμό του μέσου όρου. Μετά την εκτέλεση του προγράμματος θα πάρουμε ως αποτέλεσμα τον μέσο όρο βαθμολογίας του μαθητή. Σχηματικά μπορούμε να αναπαραστήσουμε το παραπάνω παράδειγμα ως εξής:

Δεδομένα και πρό-γραμμα

Είσοδος

Επεξεργασία

Έξοδος

Απάντη-ση στο πρόβλημα (πληρο-φορία)

Επίλυση προβλήματος

Ερωτήσεις - Δραστηριότητες

1. Σας απασχολεί το πρόβλημα της αγοράς ενός καινούριου υπολογιστή. Καταγράψτε τα απλούστερα προβλήματα στα οποία μπορεί να αναλυθεί το πρόβλημα αυτό.

2. Αναζητήστε και κατόπιν καταγράψτε δύσκολα προβλήματα που μπορούν να επιλυθούν εύκολα με τη χρήση του υπολογιστή. Διαλέξτε ένα από αυτά και καταγράψτε τα δεδομένα που πρέπει να δοθούν σε έναν υπολογιστή, προκειμένου αυτός να το επιλύσει, καθώς και το αποτέλεσμα που αναμένεται μετά την επίλυσή του.

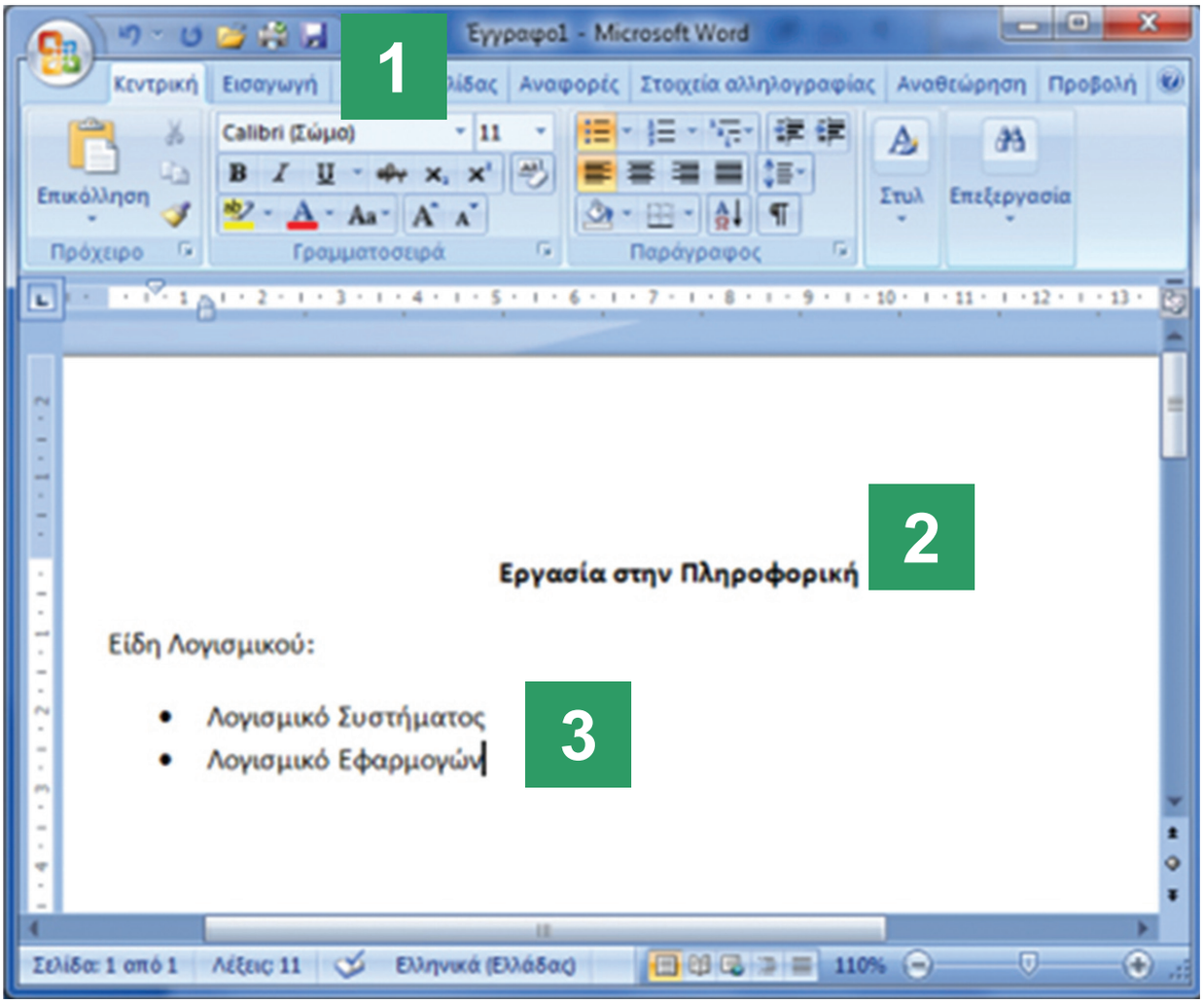
5.2 Ανάπτυξη εφαρμογών

Η έννοια της εφαρμογής

Σε προηγούμενο κεφάλαιο είδαμε ότι τα προγράμματα σε έναν υπολο-

γιστή χωρίζονται σε δύο μεγάλες κατηγορίες, στο Λογισμικό Εφαρμογών και στο Λογισμικό Συστήματος. Παραδείγματα προγραμμάτων που ανήκουν στο Λογισμικό Εφαρμογών είναι τα προγράμματα επεξεργασίας κειμένου, φωτογραφίας, ήχου, βίντεο, τα προγράμματα παρουσίασης, τα εκπαιδευτικά προγράμματα, τα παιχνίδια. Τα προγράμματα που ανήκουν στο Λογισμικό Εφαρμογών ονομάζονται απλά και εφαρμογές (applications - apps). Η λέξη «εφαρμογή» χρησιμοποιείται, επειδή κάθε πρόγραμμα έχει μία συγκεκριμένη εφαρμογή για τον χρήστη του και βασίζεται σε μία ανάγκη του. Οι εφαρμογές πρέπει να υποστηρίζουν τις καθημερινές δραστηριότητες των χρηστών των υπολογιστών και των φορητών συσκευών (έξυπνων κινητών, tablets) με

αποδοτικό και γρήγορο τρόπο. Για παράδειγμα, ο επεξεργαστής κειμένου μπορεί να χρησιμοποιηθεί από έναν μαθητή για τη συγγραφή και μορφοποίηση μιας εργασίας σε ένα μάθημα. Το πρόβλημα του μαθητή είναι η συγγραφή της εργασίας του και σε αυτό τον βοηθάει ο υπολογιστής και το πρόγραμμα επεξεργασίας κειμένου (εφαρμογή) που εκτελείται σε αυτόν.



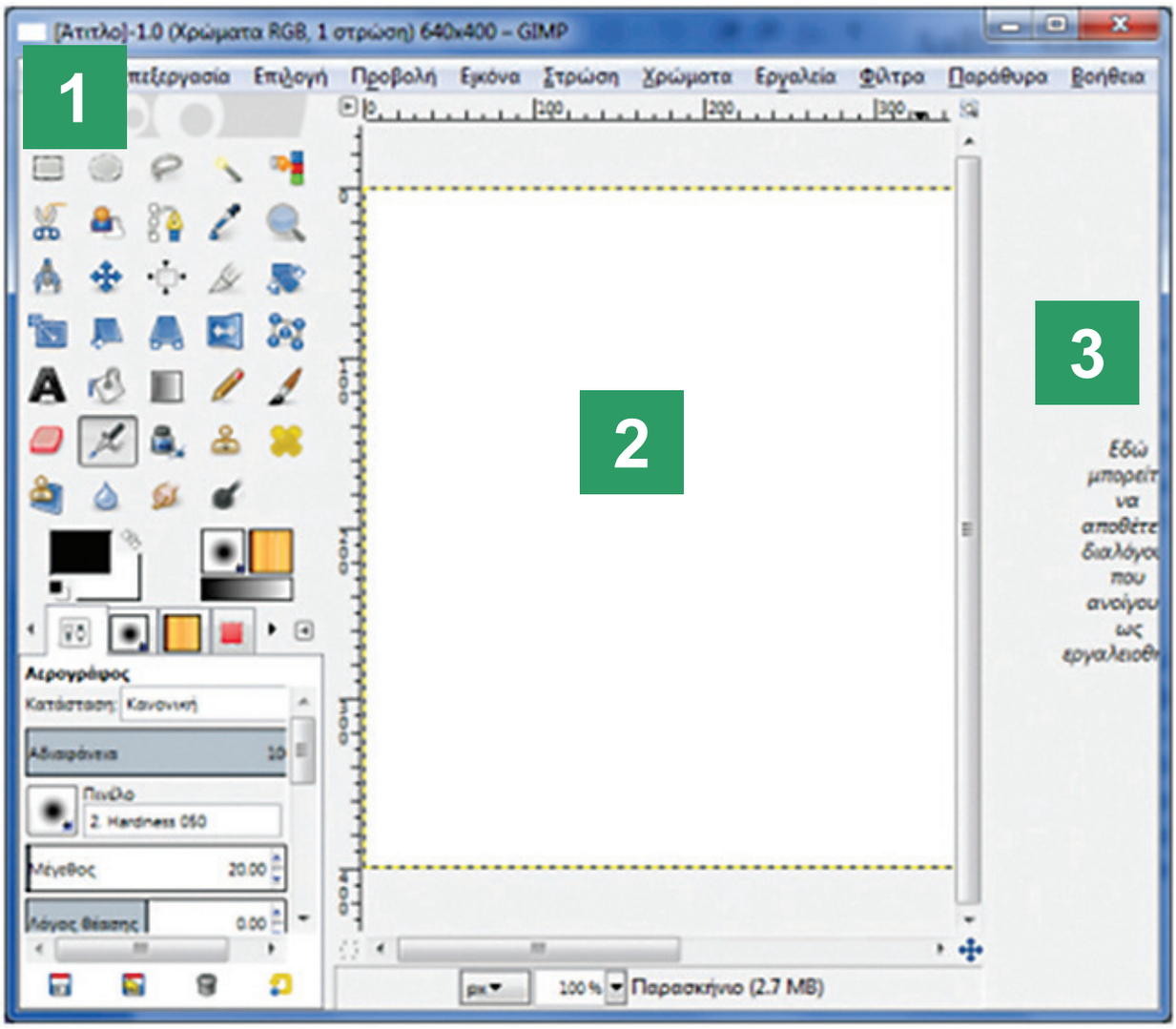
Εικόνα 5.1. Εφαρμογή επεξεργασίας κειμένου

1 Έγγραφο 1 - Microsoft Word

2 Εργασία στην Πληροφορική

3 Είδη Λογισμικού

- Λογισμικό Συστήματος
- Λογισμικό Εφαρμογών



Εικόνα 5.2. Εφαρμογή επεξεργασίας φωτογραφιών

1 Γραμμή Εργαλείων

2 Χώρος Επεξεργασίας Εικόνας

3 Εδώ μπορείτε να τοποθετείτε διαλόγους που ανοίγουν ως εργαλειοθήκη

Εικόνα 5.3. Εφαρμογή καταγραφής και διαμοιρασμού φωτογραφιών για φορητές συσκευές

Κύκλος ζωής εφαρμογών

Η ανάπτυξη εφαρμογών πρέπει να ακολουθεί μια συστηματική διαδικασία με βήματα-φάσεις, ώστε να αποφεύγονται τα σφάλματα, οι δυσλειτουργίες και οι ελλείψεις. Εξάλλου, με το πέρασμα των χρόνων, τα προγράμματα γίνονται εκτενέστερα σε μέγεθος και πιο πολύπλοκα, οπότε και η κατασκευή τους γίνεται πιο απαιτητική.

Μια εφαρμογή ξεκινάει τον κύκλο ζωής της από τη στιγμή που θα καθοριστούν οι απαιτήσεις και οι προδιαγραφές της, και τελειώνει, όταν εξαντληθούν τα περιθώρια συντήρησής της (προσθήκες, αλλαγές και βελτιώσεις). Οι εμπλεκόμενοι στη διαδικασία αυτή είναι ο πελάτης (εταιρεία, οργανισμός ή άτομο) που επενδύει στην ανάπτυξη

της εφαρμογής, ο κατασκευαστής (εταιρεία, οργανισμός ή άτομο-προγραμματιστής) που αναπτύσσει την εφαρμογή και οι χρήστες που θα χρησιμοποιήσουν την εφαρμογή. Ο κύκλος ζωής μιας εφαρμογής παρουσιάζεται διαγραμματικά στο σχήμα 5.2. Τα βελάκια δείχνουν τη σειρά των φάσεων του κύκλου ζωής, κάθε φάση οδηγεί στην επόμενη, αλλά μπορεί να οδηγήσει και στην προηγούμενη, αν παραστεί ανάγκη επανακαθορισμού κάποιων στοιχείων.

Στη φάση της **Ανάλυσης** καταγράφονται αναλυτικά τα δεδομένα και τα ζητούμενα του προβλήματος που καλείται να επιλύσει η υπό ανάπτυξη εφαρμογή. Περιγράφονται οι προδιαγραφές και οι απαιτήσεις των μελλοντικών χρηστών της εφαρμογής: ποιες λειτουργίες

θα υποστηρίζει, πώς θα εκτελούνται αυτές οι λειτουργίες, σε ποιο περιβάλλον θα δουλεύει, πόσο αποδοτική, εύχρηστη, ασφαλής και αξιόπιστη θα είναι η εφαρμογή.

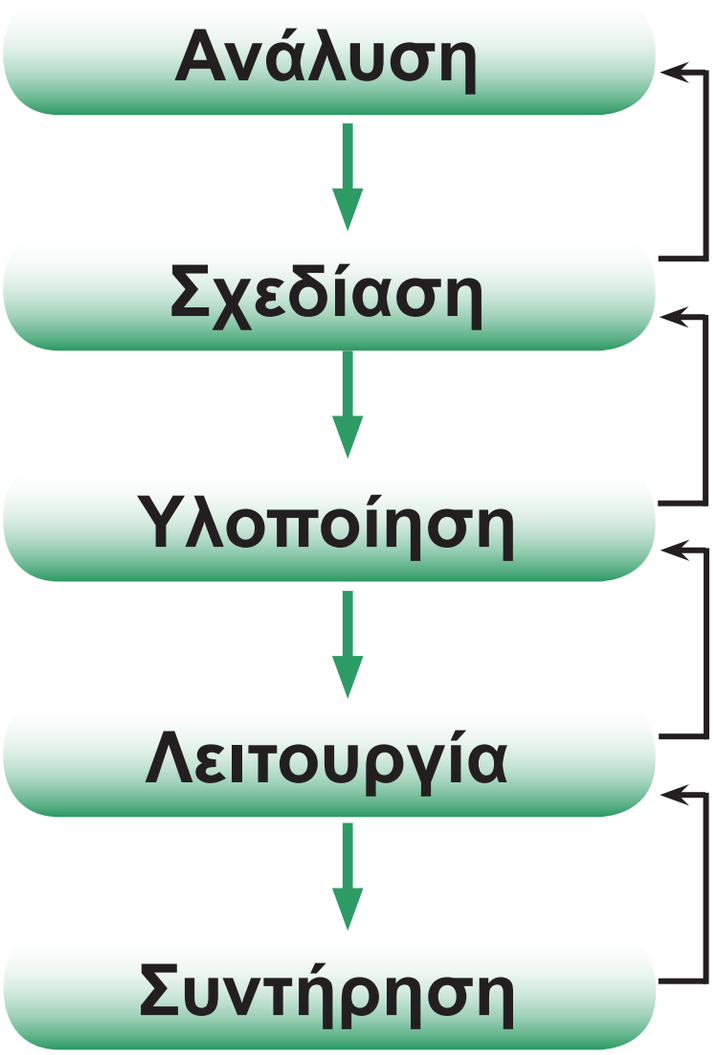
Στη φάση της **Σχεδίασης** καθορίζονται οι ενότητες (μέρη) από τις οποίες θα αποτελείται η εφαρμογή καθώς και οι σχέσεις μεταξύ τους. Σχεδιάζονται οι αλγόριθμοι και επιλέγονται οι δομές δεδομένων που θα χρησιμοποιηθούν σε κάθε ενότητα.

Στη φάση της **Υλοποίησης** επιλέγεται η γλώσσα προγραμματισμού για την υλοποίηση της εφαρμογής. Οι προγραμματιστές με βάση τους αλγόριθμους και τις δομές δεδομένων της προηγούμενης φάσης γράφουν το πρόγραμμα στην επιλεγμένη γλώσσα προγραμματισμού, αυτό εισάγεται σε ειδικό

πρόγραμμα-μεταφραστή ώστε να μετατραπεί σε «γλώσσα» κατανοητή από τον υπολογιστή, και, αν δεν υπάρχουν συντακτικά λάθη, η εφαρμογή είναι έτοιμη για εκτέλεση και χρήση.

Στη φάση της **Λειτουργίας** η εφαρμογή δίνεται αρχικά στους χρήστες για δοκιμές και ελέγχους, ώστε να βρεθούν και διορθωθούν πιθανά λάθη και αποκλίσεις από τις αρχικές προδιαγραφές, και έπειτα ξεκινάει η κανονική χρήση της.

Τέλος, στη φάση της **Συντήρησης** γίνονται όλες οι απαραίτητες προσαρμογές, αναβαθμίσεις και διορθώσεις της εφαρμογής, προκειμένου αυτή να συνεχίσει να χρησιμοποιείται απρόσκοπτα και αποδοτικά.



Σχήμα 5.2. Κύκλος ζωής εφαρμογών

Συντακτικά λάθη ονομάζουμε τα σφάλματα που σχετίζονται με τη σύνταξη μιας γλώσσας προγραμματισμού.

Αλγόριθμο ονομάζουμε ένα σύνολο εντολών (οδηγιών) που, αν εκτελεστούν με ακρίβεια, οδηγούν στην πραγματοποίηση μιας εργασίας ή την επίλυση ενός προβλήματος.

Δομή δεδομένων ονομάζουμε ένα σύνολο αποθηκευμένων δεδομένων που υφίστανται επεξεργασία από ένα σύνολο λειτουργιών (εισαγωγή, προσπέλαση, διαγραφή, αναζήτηση, ταξινόμηση κ.λπ.)

Ερωτήσεις - Δραστηριότητες

1. Γιατί είναι απαραίτητο να αναπτύσσεται μια εφαρμογή σε φάσεις;
2. Από ποιες φάσεις αποτελείται ο κύκλος ζωής μιας εφαρμογής;
3. Καταγράψτε πιθανά σφάλματα που μπορούν να γίνουν κατά την ανάπτυξη μιας εφαρμογής.

ΚΕΦΑΛΑΙΟ 6

Κύκλος Ζωής Εφαρμογών

Διδακτικές ενότητες

6.1 Γλώσσες και εργαλεία προγραμματισμού

6.2 Σύγχρονα προγραμματιστικά περιβάλλοντα

Διδακτικοί στόχοι

Σκοπός του κεφαλαίου είναι οι μαθητές να κατανοήσουν την ανάγκη ύπαρξης τόσο επαγγελματικών όσο και εκπαιδευτικών προγραμματιστικών περιβαλλόντων.

Οι μαθητές πρέπει να είναι σε θέση:

- √ να περιγράφουν τα χαρακτηριστικά των γλωσσών προγραμματισμού.
- √ να περιγράφουν τα βασικά εργαλεία προγραμματισμού που

χρησιμοποιεί ένας προγραμματιστής.

√ να αναγνωρίζουν τη χρησιμότητα και τα χαρακτηριστικά των ολοκληρωμένων περιβαλλόντων ανάπτυξης εφαρμογών.

√ να συγκρίνουν τα εκπαιδευτικά με τα επαγγελματικά προγραμματιστικά περιβάλλοντα.

√ να συνειδητοποιούν την αναγκαιότητα χρήσης ενός εκπαιδευτικού προγραμματιστικού περιβάλλοντος για την εισαγωγή ενός αρχάριου προγραμματιστή στις έννοιες του προγραμματισμού.

√ να αναγνωρίζουν τα χαρακτηριστικά των Logo-like περιβαλλόντων και των προγραμματιστικών μικρόκοσμων.

Ερωτήματα

- √ Ποιες γλώσσες προγραμματισμού γνωρίζετε;
- √ Τι εργαλεία χρειάζεται ένας προγραμματιστής, για να αναπτύξει μία εφαρμογή;
- √ Ποια επαγγελματικά προγραμματιστικά περιβάλλοντα γνωρίζετε;
- √ Ποια εκπαιδευτικά προγραμματιστικά περιβάλλοντα γνωρίζετε;
- √ Το περιβάλλον προγραμματισμού Scratch είναι επαγγελματικό ή εκπαιδευτικό;

Βασική ορολογία

Γλώσσες προγραμματισμού, προγραμματισμός, εργαλεία προγραμματισμού, προγραμματιστικό περιβάλλον, ολοκληρωμένο περιβάλλον ανάπτυξης, εκπαιδευτικό προγραμματιστικό περιβάλλον,

προγραμματιστικοί μικρόκοσμοι, οπτικός προγραμματισμός

Εισαγωγή

Τα περιβάλλοντα ανάπτυξης εφαρμογών μάς παρέχουν τη δυνατότητα να σχεδιάζουμε και να υλοποιούμε εφαρμογές. Κάθε περιβάλλον υποστηρίζει συγκεκριμένες γλώσσες προγραμματισμού και έχει ιδιαίτερα χαρακτηριστικά. Αυτό δεν αποκλείει την κατηγοριοποίησή τους με βάση κάποια κοινά χαρακτηριστικά. Έτσι, έχουμε τα επαγγελματικά προγραμματιστικά περιβάλλοντα που χρησιμοποιούνται κυρίως από έμπειρους επαγγελματίες προγραμματιστές με στόχο την παραγωγή εφαρμογών για εμπορική χρήση. Ένα τέτοιο περιβάλλον δεν προτείνεται για εκμάθηση προγραμματισμού σε

αρχάριους χρήστες, όπως οι μαθητές. Ειδικά γι' αυτό τον σκοπό έχουν σχεδιαστεί απλούστερα περιβάλλοντα, που διευκολύνουν τη σύνταξη των εντολών, για να είναι εύκολη η εκμάθηση της ανάπτυξης ενός προγράμματος, και ονομάζονται εκπαιδευτικά προγραμματιστικά περιβάλλοντα.

6.1 Γλώσσες και εργαλεία προγραμματισμού

Γλώσσες προγραμματισμού

Οι εντολές των προγραμμάτων γράφονται από τους **προγραμματιστές** σε τεχνητές γλώσσες που ονομάζονται **γλώσσες προγραμματισμού**. Οι γλώσσες προγραμματισμού εξελίσσονται με την πάροδο του χρόνου και οποιαδήποτε εφαρμογή βλέπετε στον υπολογιστή σας έχει αναπτυχθεί με χρήση μιας από αυτές.

Κάθε υπολογιστής μπορεί να κατανοήσει και να εκτελέσει εντολές που είναι διατυπωμένες με έναν καθορισμένο τρόπο, ο οποίος έχει σχέση με τον επεξεργαστή του. Οι γλώσσες προγραμματισμού, στις οποίες γράφονται οι εντολές αυτές, ονομάζονται **γλώσσες μηχανής**, και αποτελούνται από μια ακολουθία δυαδικών ψηφίων (0 και 1). Τα προγράμματα που είναι γραμμένα σε γλώσσα μηχανής προορίζονται μόνο για τον υπολογιστή για τον οποίο δημιουργήθηκαν, μιας και κάθε τύπος υπολογιστή (με διαφορετικό επεξεργαστή) έχει τη δική του γλώσσα μηχανής. Μια γλώσσα μηχανής διακρίνεται για την άμεση και γρήγορη εκτέλεση των εντολών από τον επεξεργαστή του υπολογιστή, αλλά ταυτόχρονα είναι δύσκολη η χρήση της, ο εντοπισμός και η διόρθωση τυχόν λαθών.

Για τη διευκόλυνση της εργασίας του προγραμματισμού δημιουργήθηκαν οι **συμβολικές γλώσσες ή γλώσσες χαμηλού επιπέδου**, όπου οι εντολές είναι συντομογραφίες λέξεων της Αγγλικής γλώσσας. Οι συμβολικές γλώσσες είναι και αυτές στενά συνδεδεμένες με την αρχιτεκτονική των υπολογιστών, και έτσι ένα πρόγραμμα γραμμένο για έναν τύπο υπολογιστή δεν μπορεί να μεταφερθεί και να εκτελεστεί σε άλλο τύπο υπολογιστή. Ένα πρόγραμμα γραμμένο σε συμβολική γλώσσα, για να εκτελεστεί από έναν υπολογιστή, πρέπει να μεταφραστεί στη γλώσσα μηχανής του με ένα ειδικό πρόγραμμα που ονομάζεται **συμβολομεταφραστής**.

Η ανάγκη για ακόμα ευκολότερη συγγραφή, διόρθωση και συντήρηση προγραμμάτων, ανεξάρτητων

από τον τύπο του υπολογιστή στον οποίο θα εκτελεστούν, οδήγησε στη δημιουργία των **γλωσσών υψηλού επιπέδου**. Οι γλώσσες υψηλού επιπέδου μοιάζουν με τη φυσική μας γλώσσα και έχουν το δικό τους **αλφάβητο**, **λεξιλόγιο** και **συντακτικό**. Μερικές από τις πιο δημοφιλείς γλώσσες για ανάπτυξη γενικών ή εξειδικευμένων εφαρμογών είναι η C, η C++, η Java, η PHP, η C#, η Python, η JavaScript, η Perl, η Visual Basic, η Ruby, η Lisp, η Pascal, η Prolog και η MATLAB. Ανάλογα με το είδος της εφαρμογής που θέλουμε να αναπτύξουμε (γενικής χρήσης, εμπορική, επιστημονική, τεχνητής νοημοσύνης, παιχνίδι κ.λπ.) επιλέγουμε την κατάλληλη γλώσσα προγραμματισμού.

```
00000011 11000011
00101010 11011010
```

Εικόνα 6.1. Απόσπασμα προγράμματος σε γλώσσα μηχανής

```
LDAA $D000
ADDA $D001
STAA $D000
```

Εικόνα 6.2. Απόσπασμα προγράμματος σε συμβολική γλώσσα

C

```
#include <stdio.h>
main()
{
    printf(“Hello world!”);
}
```

C++

```
#include <iostream.h>
void main()
{
    cout << “Hello
world!”;
}
```

Python

```
print(“Hello world!”)
```

Java

```
public class HelloWorld
{
 public static void
main(String[] args) {
  System.out.
println(“Hello World!”);
 }
}
```

Εικόνα 6.3. Το πρόγραμμα που εμφανίζει το μήνυμα «Hello World!» σε διάφορες γλώσσες προγραμματισμού.

Εργαλεία προγραμματισμού

Η εργασία σύνταξης ενός προγράμματος ονομάζεται **προγραμματισμός** ή **κωδικοποίηση**, και είναι μια εξαιρετικά δημιουργική δραστηριότητα.

Τα κύρια εργαλεία που χρησιμοποιεί ένας προγραμματιστής για να αναπτύξει μία εφαρμογή σε μία συγκεκριμένη γλώσσα προγραμματισμού υψηλού επιπέδου είναι:

- ένας **συντάκτης κειμένων (editor)** με τον οποίο και γράφει το αρχικό πρόγραμμα, που ονομάζεται **πηγαίο πρόγραμμα ή κώδικας (source code)**.
- ένα **μεταφραστικό πρόγραμμα** (μεταγλωττιστή ή διερμηνευτή), το οποίο μεταφράζει το πηγαίο πρόγραμμα σε **αντικείμενο πρόγραμμα** ή **κώδικα (object code)**. Το μεταφραστικό πρόγραμμα ελέγχει το πηγαίο πρόγραμμα για **συντακτικά λάθη**, εμφανίζει κατάλληλα διαγνωστικά μηνύματα, εάν βρεθούν λάθη, και μόνο αν δεν υπάρχουν λάθη παράγεται το αντικείμενο πρόγραμμα. Το αντικείμενο

πρόγραμμα είναι σε γλώσσα μηχανής, αλλά δεν είναι ακόμη εκτελέσιμο από τον υπολογιστή και πρέπει να περάσει από κάποιες άλλες διαδικασίες.

- ένα ειδικό πρόγραμμα που ονομάζεται **συνδέτης (linker)**, το οποίο πολλές φορές συνδέει το αντικείμενο πρόγραμμα ή ένα σύνολο από αντικείμενα προγράμματα με έτοιμα υποπρογράμματα της βιβλιοθήκης της γλώσσας προγραμματισμού ή του προγραμματιστή. Το τελικό πρόγραμμα που παράγεται είναι το **εκτελέσιμο πρόγραμμα ή κώδικας (executable code)**, είναι διατυπωμένο σε γλώσσα μηχανής και μπορεί να εκτελεστεί άμεσα από τον επεξεργαστή του υπολογιστή.
- **εργαλεία εντοπισμού λαθών (debuggers)** με τα οποία ο

προγραμματιστής παρακολουθεί τι ακριβώς συμβαίνει στο παρασκήνιο κατά την εκτέλεση ενός προγράμματος.

Μεταφραστικά προγράμματα:

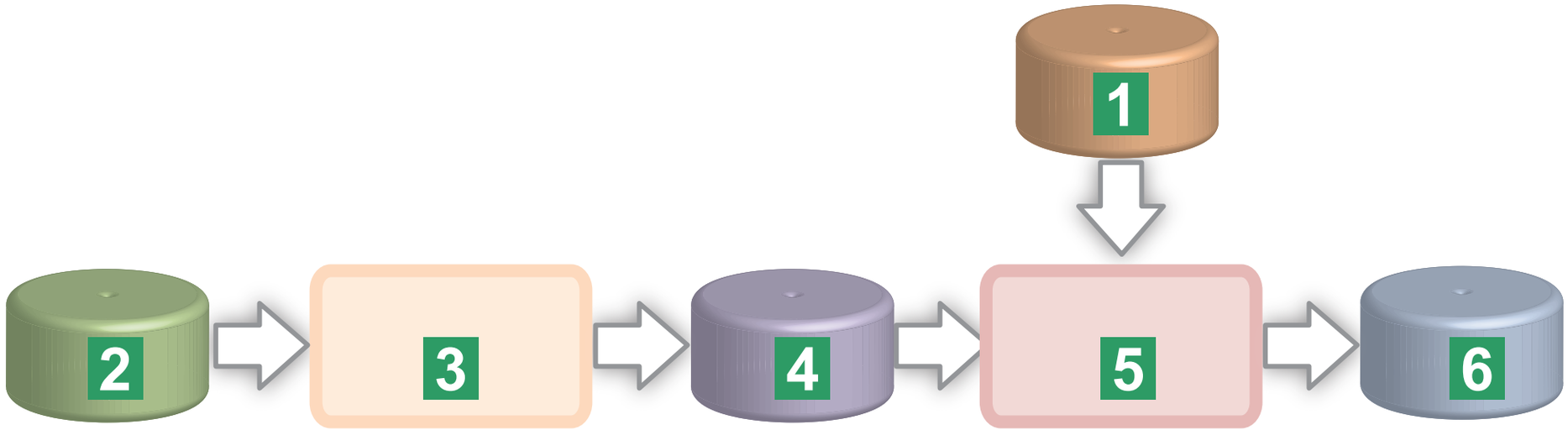
Μεταγλωττιστής:

ελέγχει όλο το πηγαίο πρόγραμμα για συντακτικά λάθη και μετά το μετατρέπει σε γλώσσα μηχανής.

Διερμηνευτής:

ελέγχει μία εντολή κάθε φορά, την εκτελεί κι ύστερα ελέγχει την επόμενη.

Ένα περιβάλλον (λογισμικό) που περιλαμβάνει τα παραπάνω εργαλεία και χρησιμοποιείται για την ανάπτυξη εφαρμογών ονομάζεται **προγραμματιστικό περιβάλλον** ή **περιβάλλον ανάπτυξης εφαρμογών.**



Σχήμα 6.1. Μεταγλώττιση και σύνδεση προγράμματος

1. Βιβλιοθήκη
2. Πηγαίο Πρόγραμμα
3. Μεταγλωτιστής
4. Αντικείμενο Πρόγραμμα
5. Συνδέτης
6. Εκτελέσιμο Πρόγραμμα

Ερωτήσεις - Δραστηριότητες:

1. Ποια είναι τα βασικά χαρακτηριστικά των γλωσσών υψηλού επιπέδου;
2. Αναζητήστε πληροφορίες για το είδος των εφαρμογών που μπορούν να αναπτυχθούν για καθεμία από τις παρακάτω γλώσσες προγραμματισμού: C++, Java, Python, PHP, JavaScript.
3. Περιγράψτε τα βήματα της διαδικασίας μεταγλώττισης και σύνδεσης προγράμματος.

6.2 Σύγχρονα προγραμματιστικά περιβάλλοντα

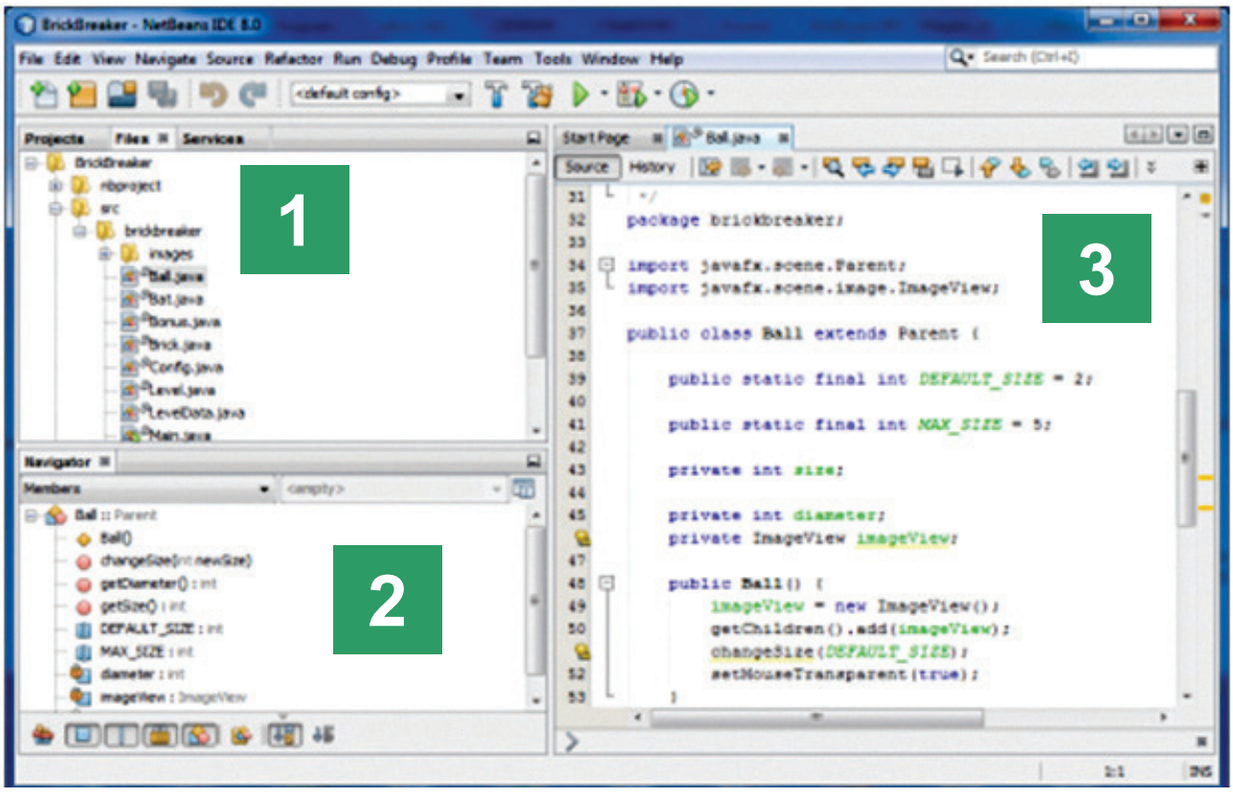
Επαγγελματικά προγραμματιστικά περιβάλλοντα

Η ανάπτυξη των σύγχρονων εφαρμογών είναι μια απαιτητική

και δύσκολη διαδικασία. Οι επαγγελματίες προγραμματιστές χρησιμοποιούν για τη σχεδίαση, την κωδικοποίηση, τον έλεγχο λαθών και τη συντήρηση μιας εφαρμογής ένα **ολοκληρωμένο περιβάλλον ανάπτυξης (Integrated Development Environment -IDE)**, όπως Dev-C++, BlueJ, JBuilder, NetBeans IDE, Eclipse, Microsoft Visual Studio, Apple Xcode και Komodo IDE & EDIT. Στα ολοκληρωμένα περιβάλλοντα ανάπτυξης συνυπάρχουν σε ένα ενοποιημένο περιβάλλον διάφορα από τα εργαλεία που αναφέρθηκαν παραπάνω, και έτσι διευκολύνεται και επιταχύνεται η ανάπτυξη μιας εφαρμογής είτε από έναν είτε από ομάδα προγραμματιστών.

Μάλιστα κάποια από αυτά διαθέτουν και γραφικά εργαλεία σχεδίασης των εφαρμογών, για παράδειγμα

οπτικοποίηση με διαγράμματα των τμημάτων μιας εφαρμογής και της μεταξύ τους αλληλεπίδρασης, αυτόματη συμπλήρωση κώδικα (εντολών), δημιουργία των αντικειμένων της γραφικής διεπαφής χρήστη (μενού επιλογών, κουμπιά, πλαίσια διαλόγου κ.λπ.) με χρήση κατάλληλων εργαλειοθηκών. Επίσης, αρκετά εγκαθίστανται σε διάφορα Λειτουργικά Συστήματα (Windows, Linux, Mac OS X), υποστηρίζουν σχεδόν όλες τις δημοφιλείς γλώσσες προγραμματισμού (Java, C/C++, Python, PHP, JavaScript) και χρησιμοποιούνται για την ανάπτυξη αυτόνομων εφαρμογών για υπολογιστή, εφαρμογών για το Διαδίκτυο και εφαρμογών για φορητές συσκευές (έξυπνα κινητά, tablets).

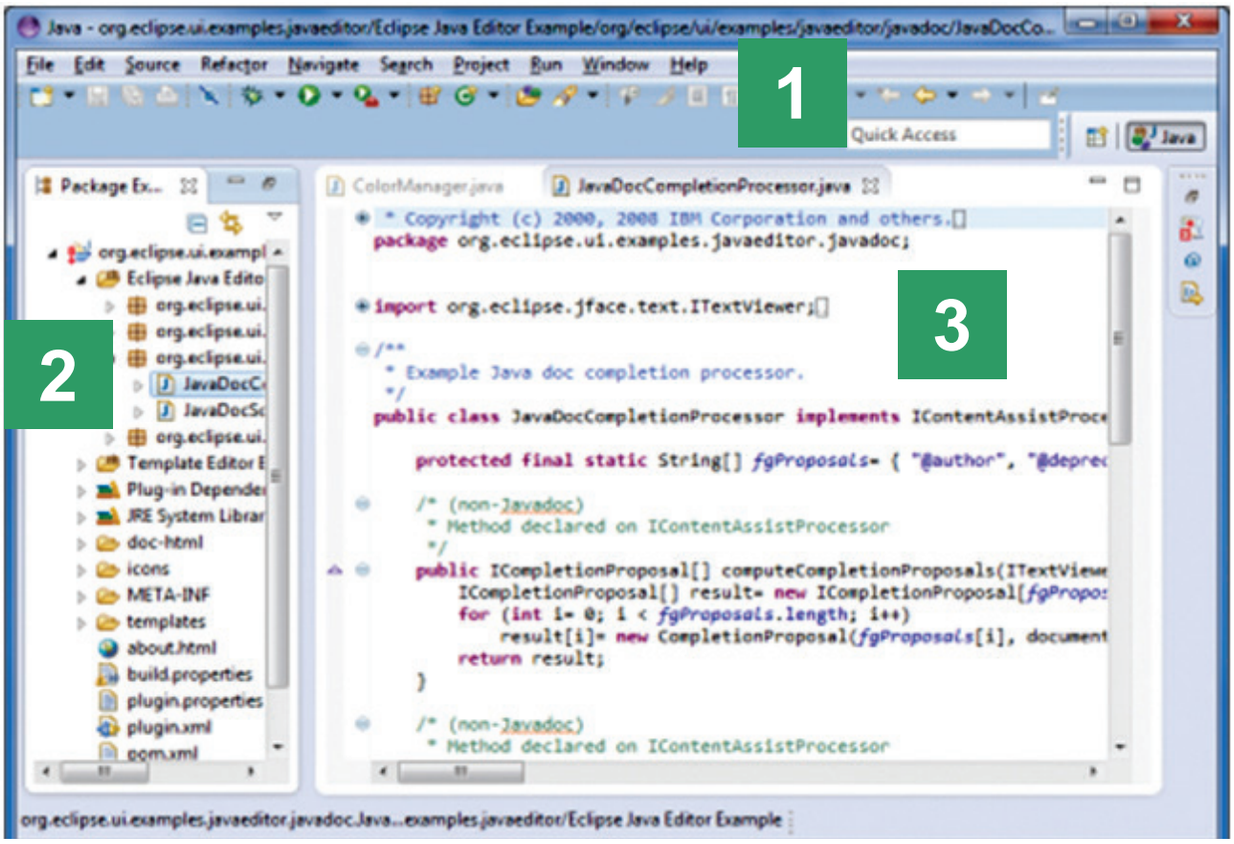


Εικόνα 6.4. Το περιβάλλον NetBeans IDE

1 Files

2 Navigator

3 Κώδικας του προγράμματος Ball.java



Εικόνα 6.5. Το περιβάλλον Eclipse

1. **Γραμμή Εργαλείων**
2. **Package Ex.**
3. **Κώδικας του προγράμματος JavaDoc.Completion Processor. java**

Εκπαιδευτικά προγραμματιστικά περιβάλλοντα

Οι αρχάριοι προγραμματιστές συνήθως αντιμετωπίζουν προβλήματα στα πρώτα τους βήματα με μια επαγγελματική γλώσσα προγραμματισμού και τα αντίστοιχα περιβάλλοντα ανάπτυξης εφαρμογών. Για τον λόγο αυτό έχουν αναπτυχθεί εκπαιδευτικές γλώσσες προγραμματισμού και αντίστοιχα εκπαιδευτικά προγραμματιστικά περιβάλλοντα για την εισαγωγή στις βασικές αρχές του προγραμματισμού και την ανάπτυξη μικρών και απλών εφαρμογών που μπορούν να ονομαστούν και μικροεφαρμογές.

Παράδειγμα εκπαιδευτικής γλώσσας αποτελεί η LOGO, και τα προγραμματιστικά περιβάλλοντα που βασίζονται σε αυτήν ονομάζονται

Logo-like. Στο ίδιο πλαίσιο εντάσσονται και τα προγραμματιστικά περιβάλλοντα που ονομάζονται προγραμματιστικοί μικρόκοσμοι. Τα βασικά χαρακτηριστικά των περιβαλλόντων αυτών είναι:

- √ **Ένας «πρωταγωνιστής-κεντρικός ήρωας» (χελώνα, ποντίκι, ρομπότ, πασχαλίτσα κ.λπ.) κινείται σε έναν χώρο, για να πετύχει έναν στόχο.**
- √ **Διαθέτουν περιορισμένο ρεπερτόριο εντολών με απλή σύνταξη και απλές δομές δεδομένων, ενώ κάθε επαγγελματική γλώσσα προγραμματισμού διαθέτει κατά κανόνα μεγάλο ρεπερτόριο εντολών με πολύπλοκους κανόνες σύνταξης.**
- √ **Επειδή η κίνηση του ήρωα είναι άμεση και εμφανής, ο χρήστης διαπιστώνει εύκολα αν πέτυχε η εκτέλεση του προγράμματος τον**

προκαθορισμένο στόχο (π.χ. αν η χελώνα σχημάτισε ένα συγκεκριμένο σχήμα) και μπορεί να διορθώσει το πρόγραμμα σε περίπτωση λάθους.

√ Διευκολύνουν τη δημιουργία παιχνιδιών.

√ Σε κάποιους μικρόκοσμους η σύνταξη των εντολών γίνεται χωρίς πληκτρολόγηση, αλλά με σύρσιμο και τοποθέτηση σε μια σειρά πλακιδίων (blocks) (π.χ. Scratch, BYOB). Πρόκειται για οπτικά περιβάλλοντα προγραμματισμού, όπου ο προγραμματιστής δεν πληκτρολογεί εντολές, αλλά επιλέγει και τοποθετεί κατάλληλα γραφικά στοιχεία.

√ Ορισμένοι μικρόκοσμοι παρέχουν τρισδιάστατη απεικόνιση (π.χ. Kodu, Yenka, StarLogo TNG).

Για λίγο πιο προχωρημένους προγραμματιστές μερικά δημοφιλή προγραμματιστικά περιβάλλοντα είναι τα:

- √ Game Maker: οπτικό περιβάλλον προγραμματισμού για την ανάπτυξη παιχνιδιών.
- √ Alice: 3D περιβάλλον για την ανάπτυξη εικονικών κόσμων με δυναμικές κινήσεις χαρακτήρων και αλληλεπίδραση με τον χρήστη.
- √ App Inventor: οπτικό περιβάλλον προγραμματισμού με πλακίδια (blocks) για ανάπτυξη εφαρμογών για φορητές συσκευές (έξυπνα κινητά, tablets) με Λειτουργικό Σύστημα Android.

Εντολές σε LOGO για τη δημιουργία ενός τετραγώνου:

στκ

μπ 100

δε 90

μπ 100

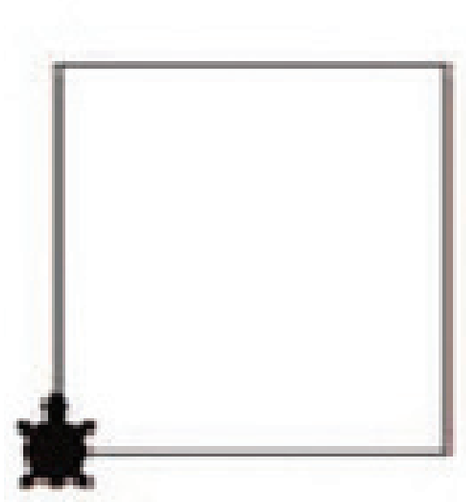
δε 90

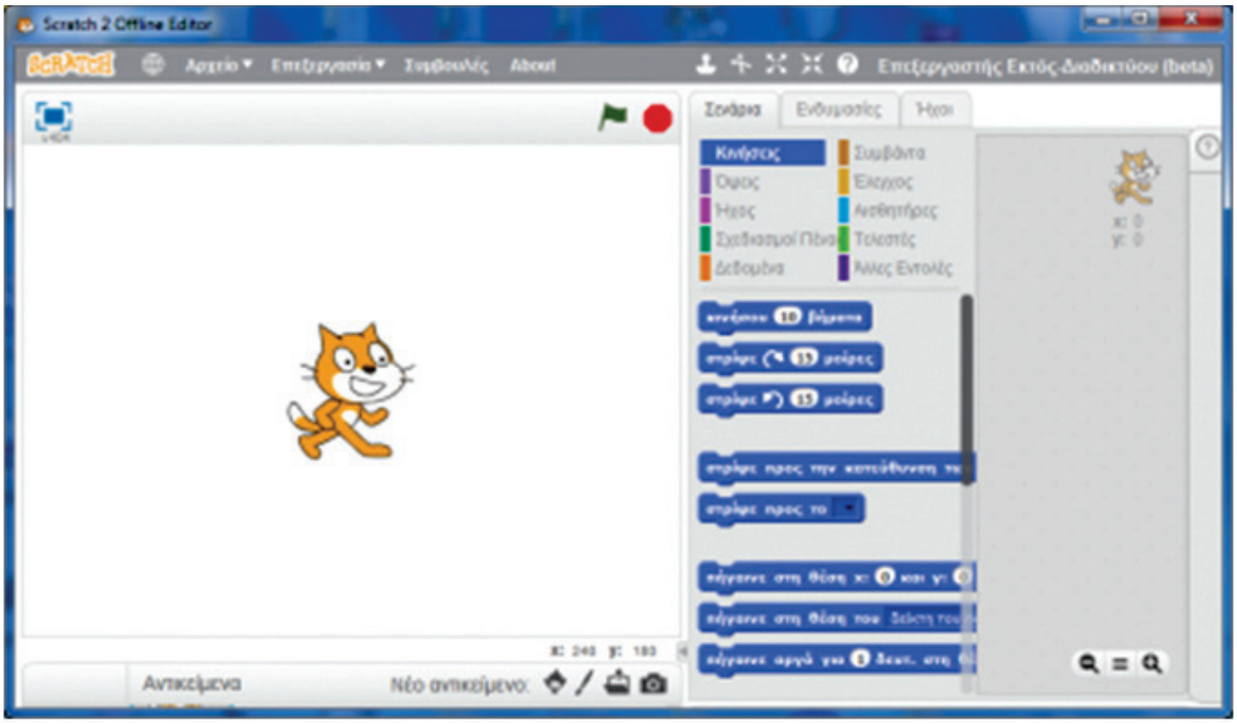
μπ 100

δε 90

μπ 100

δε 90





Εικόνα 6.6. Το περιβάλλον Scratch

Εικόνα 6.7. Το περιβάλλον Kodu

Ερωτήσεις - Δραστηριότητες:

4. Περιγράψτε τα βασικότερα εργαλεία προγραμματισμού που χρησιμοποιεί ένας προγραμματιστής για την ανάπτυξη μιας εφαρμογής.

5. Γιατί οι επαγγελματίες προγραμματιστές χρησιμοποιούν τα ολοκληρωμένα περιβάλλοντα ανάπτυξης εφαρμογών;

6. Καταγράψτε τις βασικές διαφορές μεταξύ των επαγγελματικών και των εκπαιδευτικών προγραμματιστικών περιβαλλόντων.

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

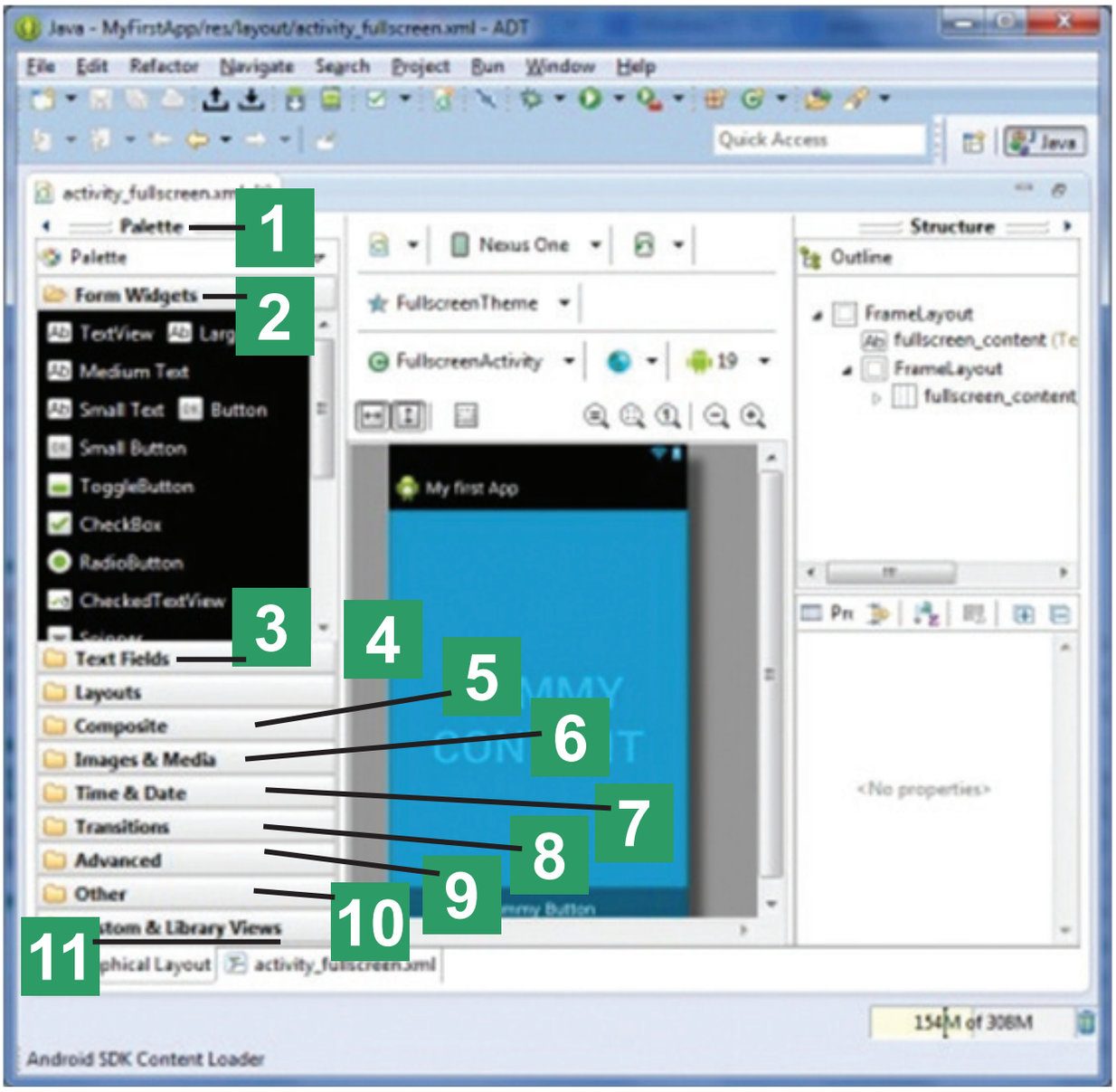
Οι φορητές συσκευές, κυρίως τα έξυπνα κινητά (smartphones) και οι ταμπλέτες (tablets), έχουν διεισδύσει σε πολλούς τομείς της ανθρώπινης δραστηριότητας, όπως είναι η ενημέρωση, η ψυχαγωγία και η εργασία. Οι συσκευές αυτές γίνονται ολοένα και πιο δημοφιλείς και χρηστικές χάρη στο πλήθος εφαρμογών και 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.

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 7.1. Δημοφιλή Λειτουργικά Συστήματα για φορητές συσκευές

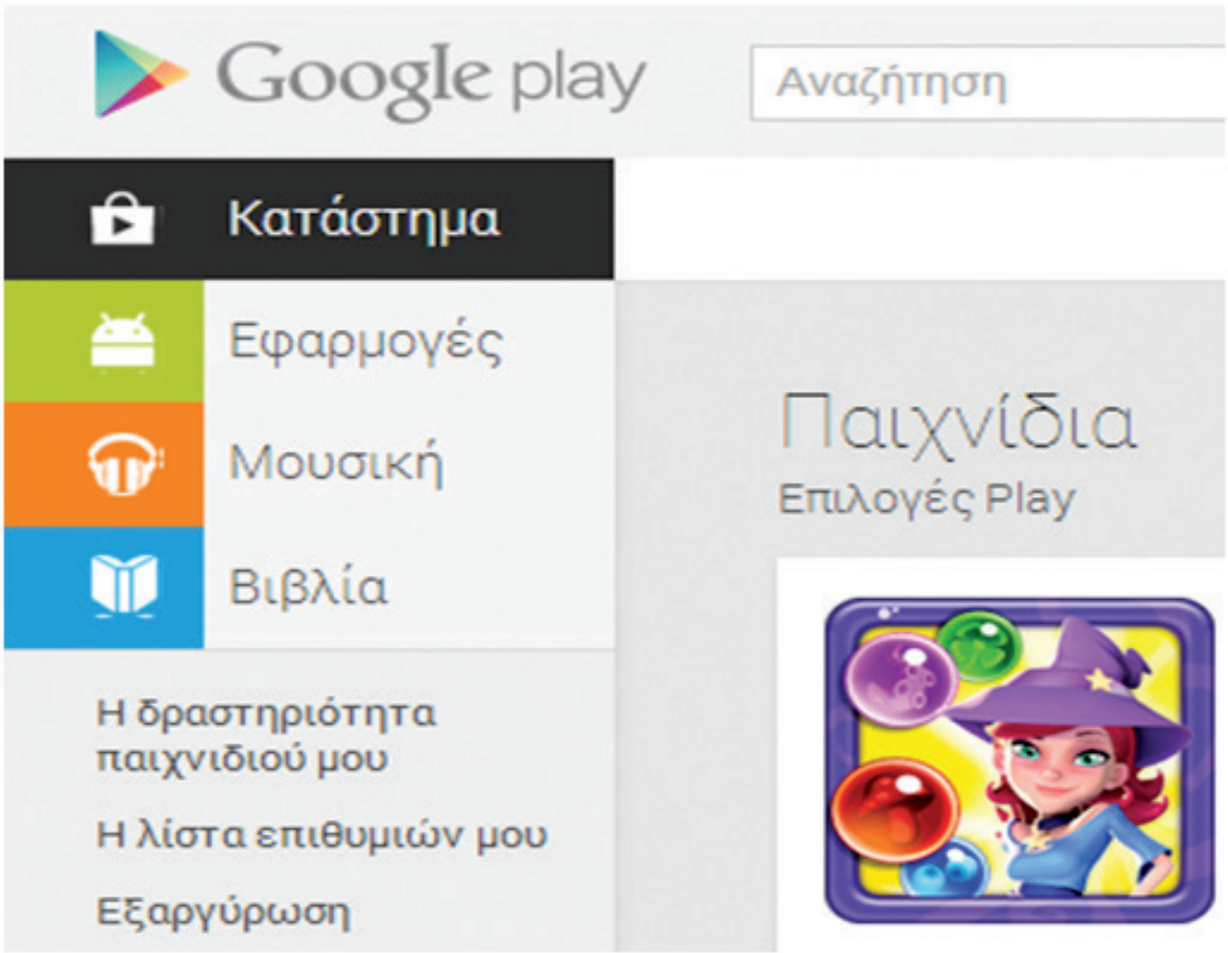


Εικόνα 7.2. Περιβάλλον ανάπτυξης εφαρμογών για φορητές συσκευές με Android

1 Palette
2 Form Widgets
3 Text Fields
4 Layouts
5 Composite
6 Images & Media
7 Time & Date
8 Transitions
9 Advanced
10 Other
11 Custom & Library Views

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

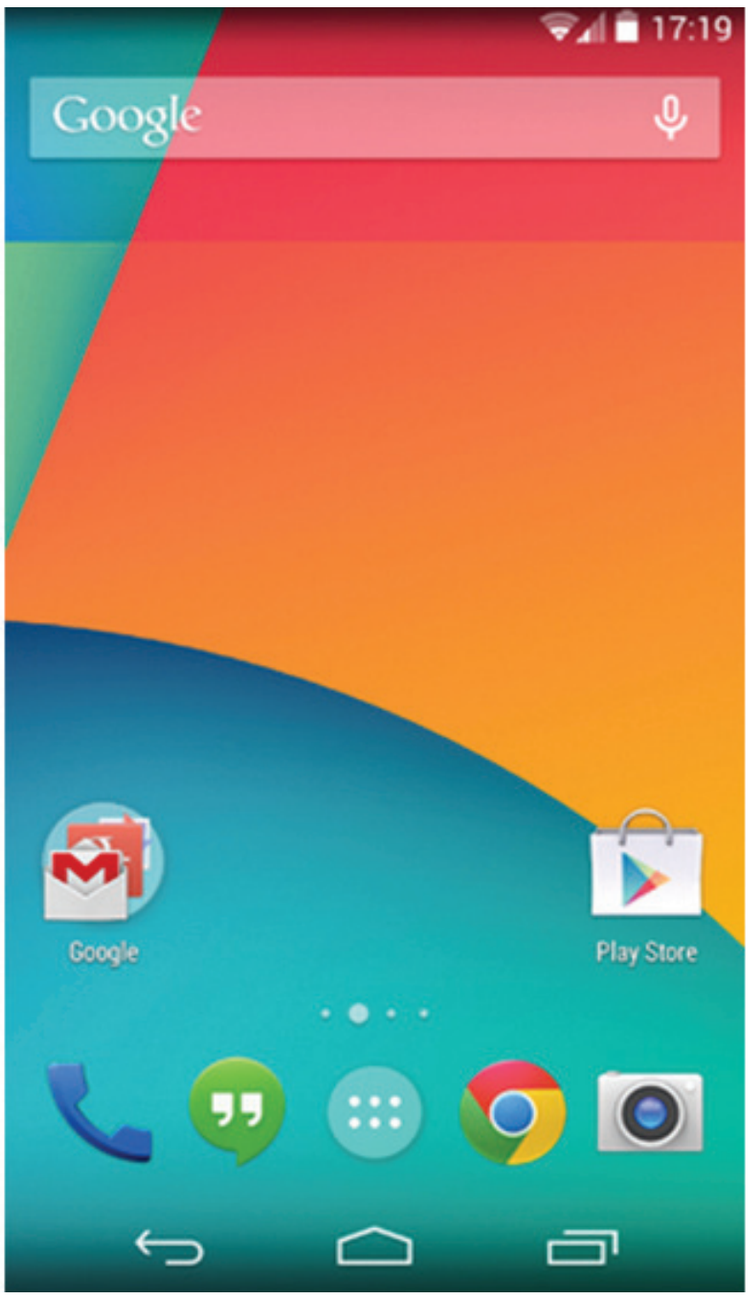
Το εκπαιδευτικό περιβάλλον ανάπτυξης εφαρμογών App Inventor

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 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).

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.

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

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.

Εικόνα 7.5. Η κλασική δομή του App Inventor

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.

Επίσημο site του App Inventor:

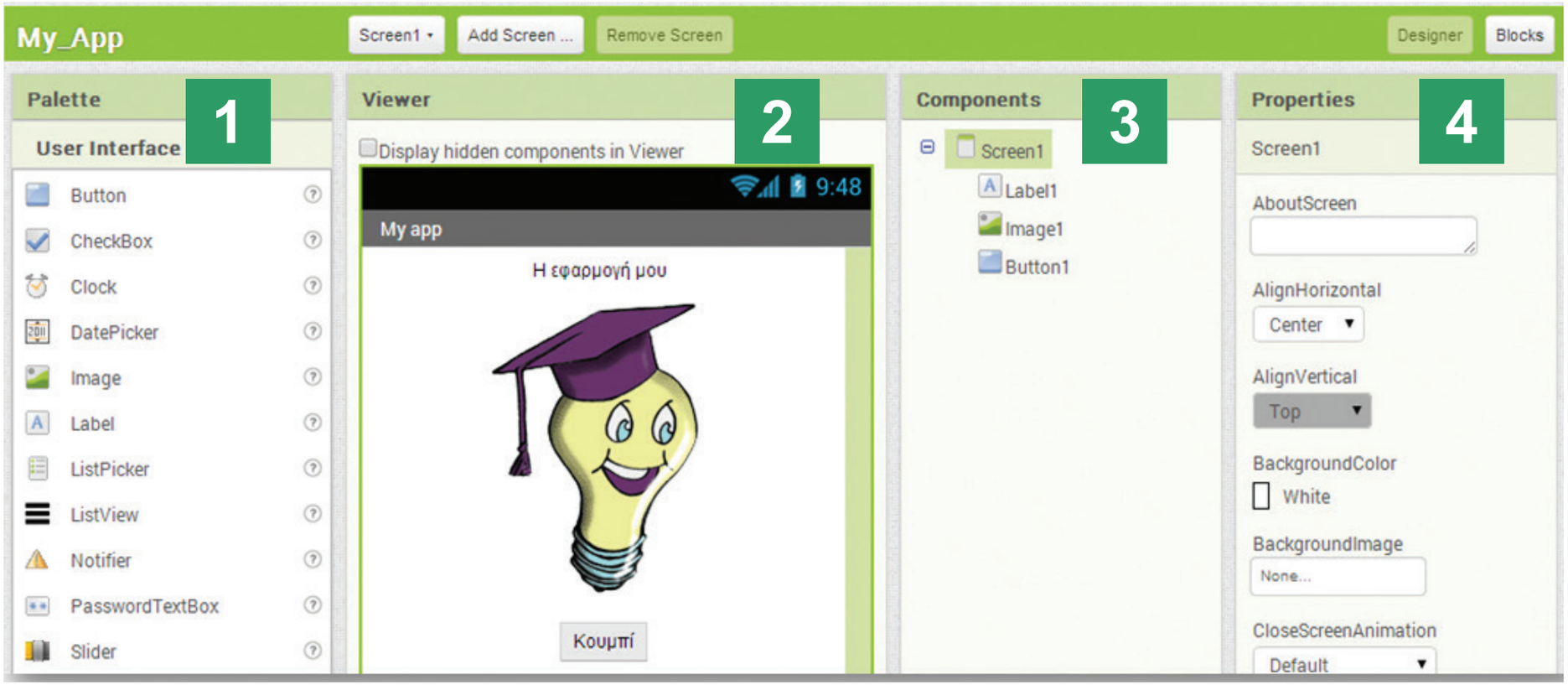
http://appinventor.mit.edu/explore

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

1 Palette

2 Viewer

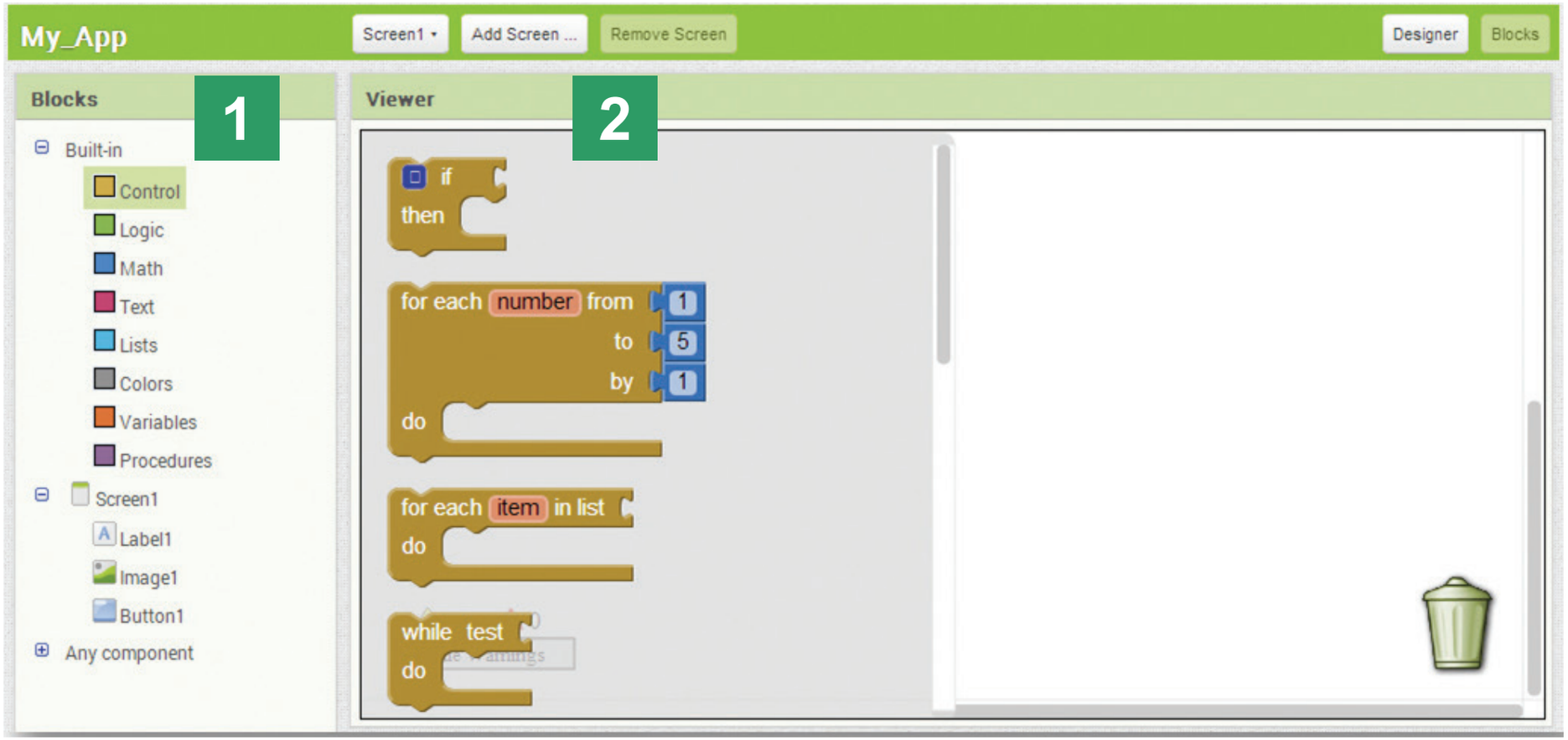
3 Components

4 Properties

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.



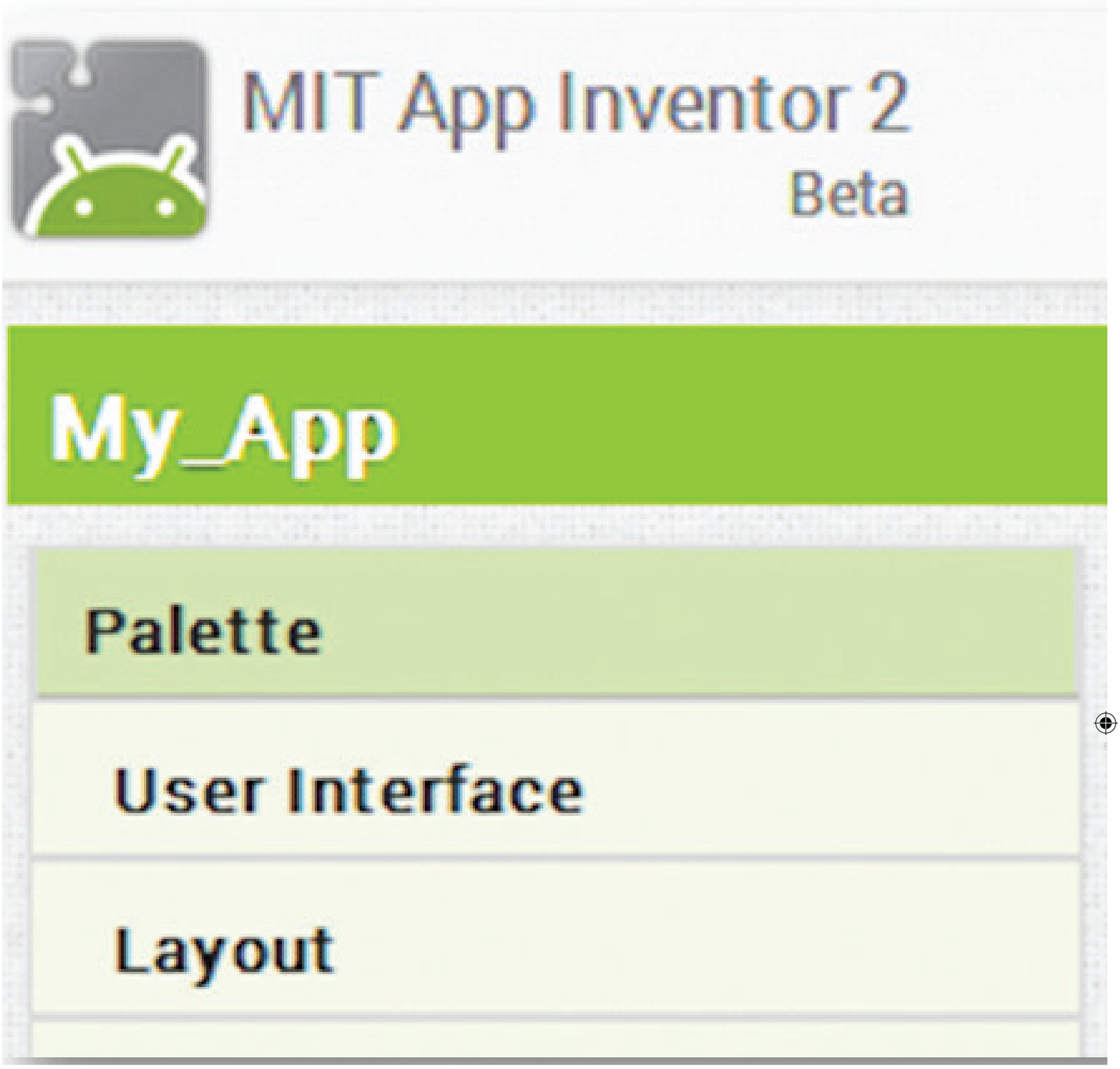
Εικόνα 7.7. App Inventor Blocks Editor

1 **Blocks**

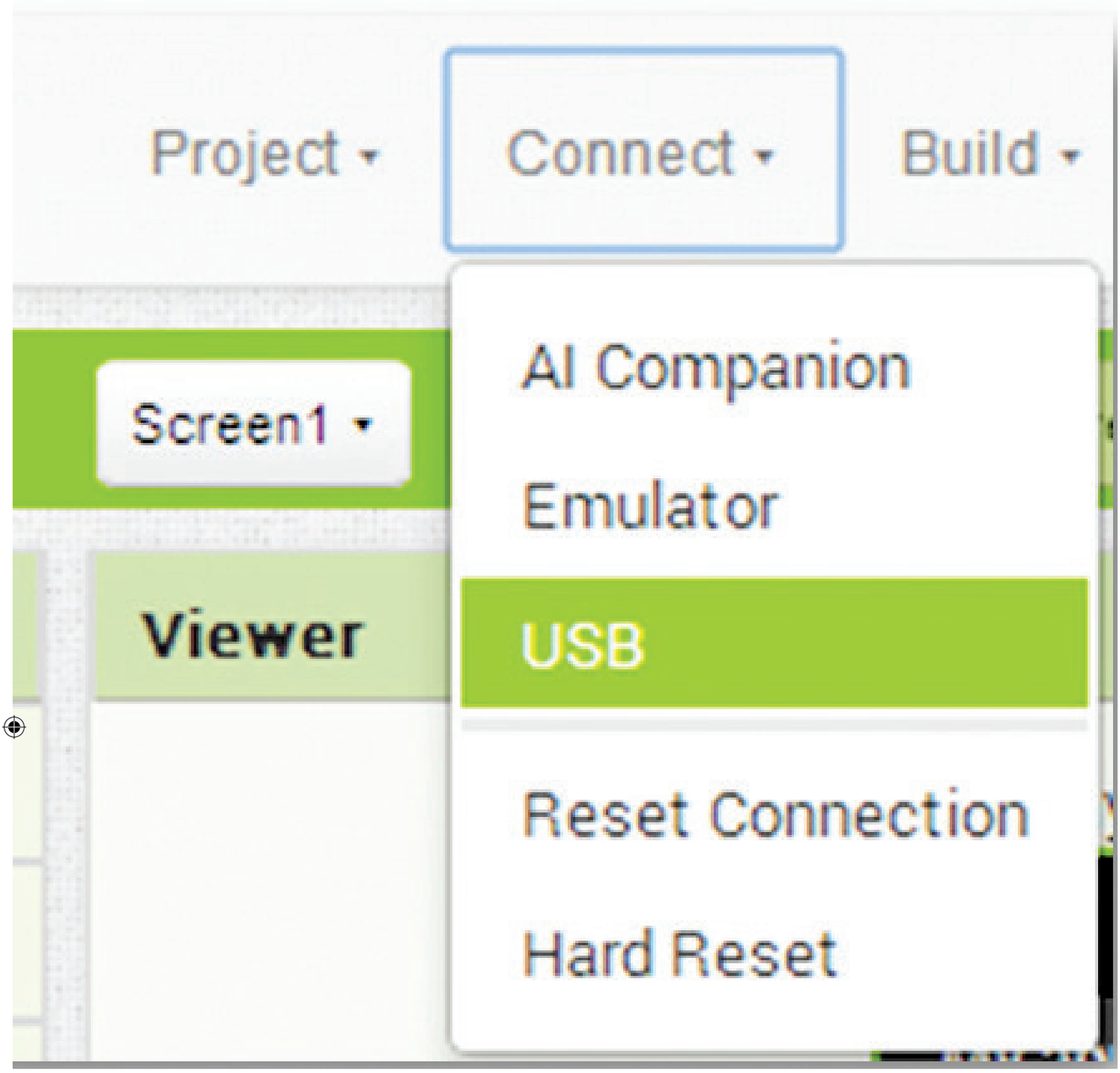
2 **Viewer**

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 7.9. Ο Emulator



Εικόνα 7.8. Σύνδεση φορητής συσκευής



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

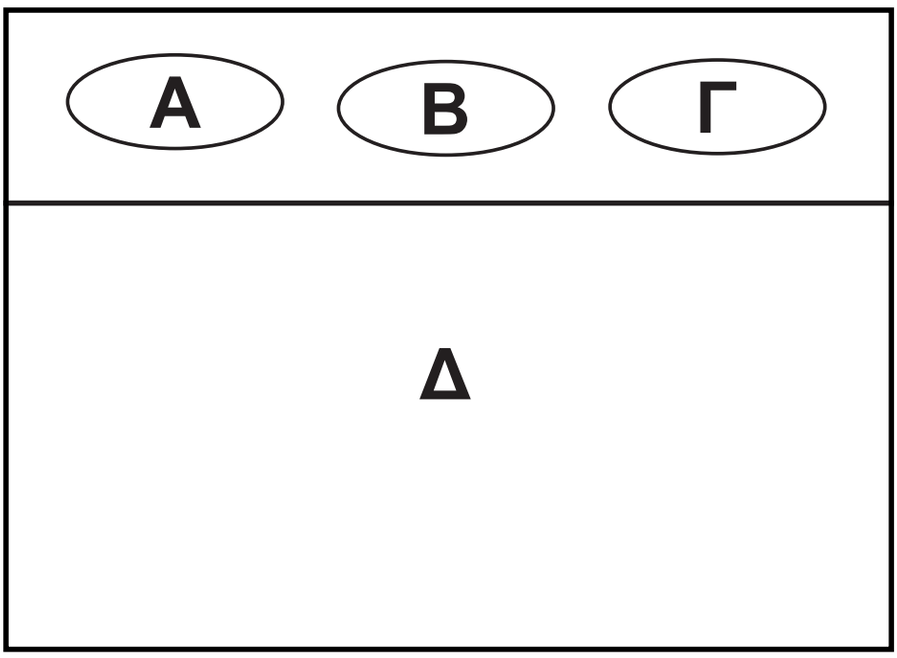
Ένας πελάτης της εταιρείας μάς ζητάει να φτιάξουμε μια εφαρμογή για φορητές συσκευές που λειτουργούν με λειτουργικό σύστημα Android. Η εφαρμογή απαιτείται να είναι πρωτότυπη, για να προκαλέσει το ενδιαφέρον των εφήβων–μαθητών στους οποίους απευθύνεται. Προτιμάμε να γίνει η υλοποίησή της με το

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.



Σχήμα 7.1. Σχεδίαση οθόνης κινητού

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:

- **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

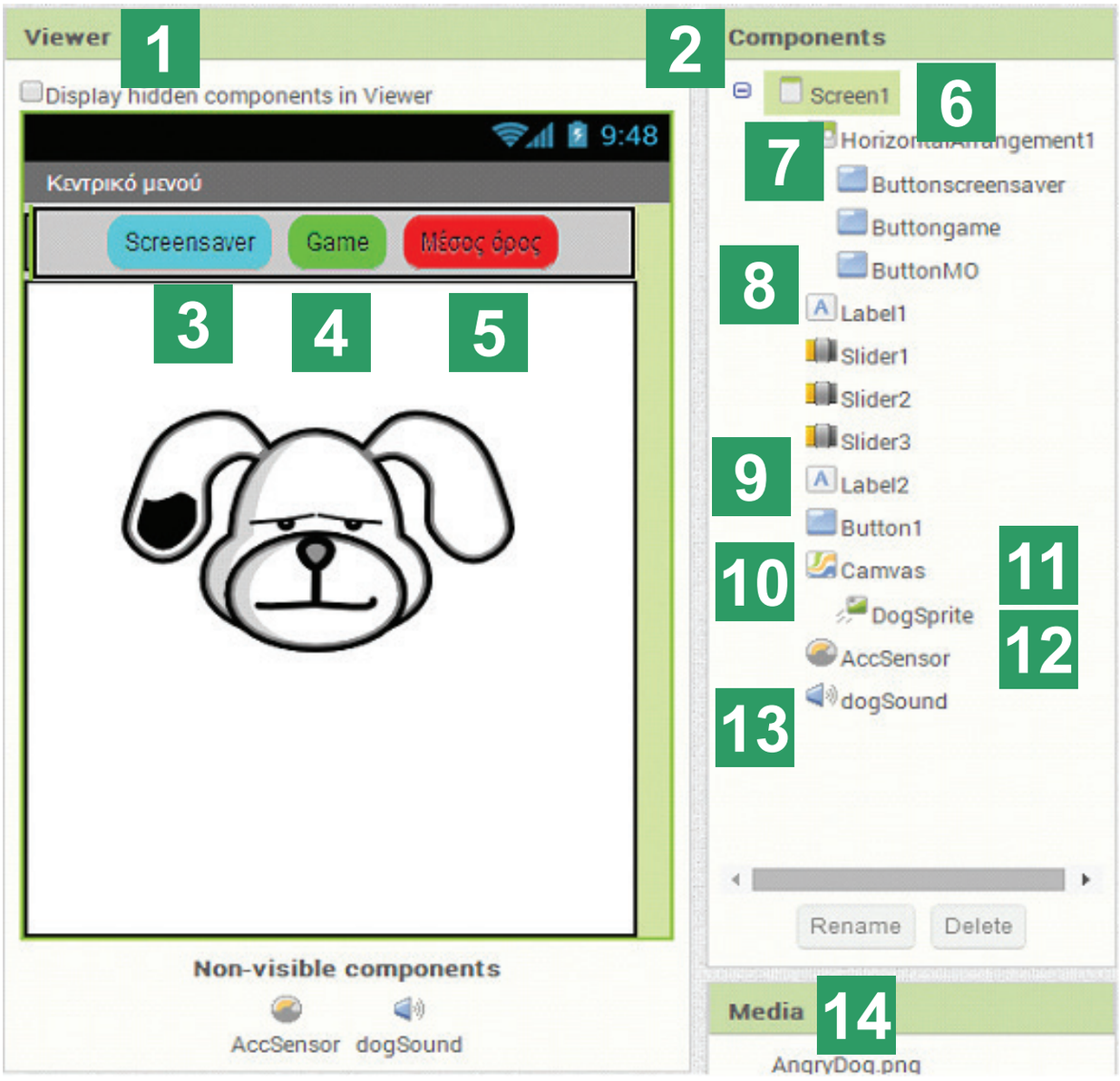
Εμφάνισε τον μέσο όρο του μαθήματος.
Αν ο ΜΟ είναι μεγαλύτερος ή ίσος από 10, τότε
    εμφάνισε ότι ο μαθητής πέρασε το μάθημα
αλλιώς
    εμφάνισε ότι ο μαθητής δεν πέρασε το μάθημα.
Τέλος_αν

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

1. Viewer
2. Components
3. Screen Saver

4 Game

5 Μέσος Όρος

6 Screen 1

7 Horizontal Arrangement 1

Buttonscreensaver

Buttongame

ButtonMO

8 Label 1

Slider 1

Slider 2

Slider 3

9 Label 2

Button 1

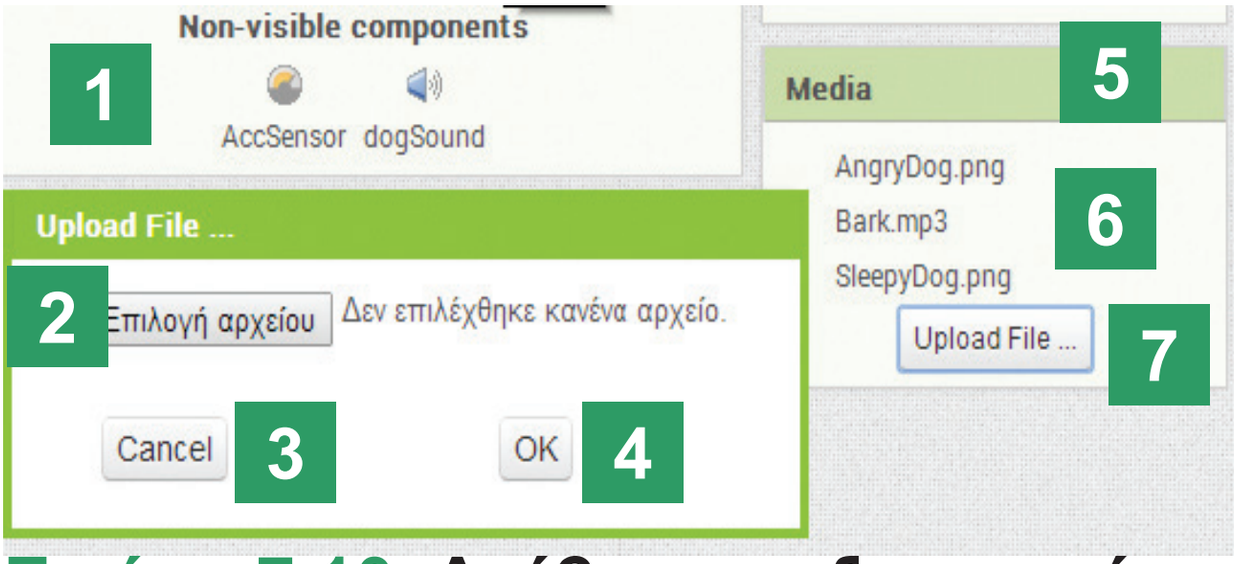
10 Camvas

11 DogSprite

12 AccSensor

13 dogSound

14 Media



Εικόνα 7.10. Ανέβασμα εξωτερικών αρχείων που θα χρησιμοποιηθούν

1. Non visible components (AccSensor DogSound)
2. Επιλογή αρχείου
3. Cancel
4. OK
5. Media
6. AngryDog.png
   Bark.mp3
   StepyDog.png
7. Upload File

Πίνακας 7.1 Τα συστατικά μέρη της		
Από την ομάδα του πλαισίου **Palette**	Επιλέγουμε το **αντικείμενο** και το μεταφέρουμε στο πλαίσιο **Viewer**	
	Screen1 (Είναι ήδη εγκατεστημένο, οπότε αλλάζουμε μόνο τις ιδιότητες)	
Layout	Horizontal Arrangement	
User Interface	Button (το μετα-φέρουμε μέσα στο πλαίσιο Horizontal Arrangement)	

εφαρμογής και οι ιδιότητες τους		
	Αλλάζουμε το Όνομα με το κουμπί Rename	**Μεταβάλλουμε τις ιδιότητες - Properties**
		BackgroundColor: LightGrey **Screen Orientation: Portrait** **Scrollable: (No)** **Title:Κεντρικό μενού**
	Horizontal Arrangement1	**PhotoArrangement** **Width: Fill Parent** **AlignHorizontal: Center**
	Button screensaver	**BackgroundColor: Cyan** **Shape: Rounded** **Text: Screensaver**

User Interface	Button (το μετα-φέρουμε μέσα στο πλαίσιο Horizontal Arrangement)	
User Interface	Button (το μετα-φέρουμε μέσα στο πλαίσιο Horizontal Arrangement)	
User Interface	Label	
User Interface	Slider	
User Interface	Slider	
User Interface	Slider	

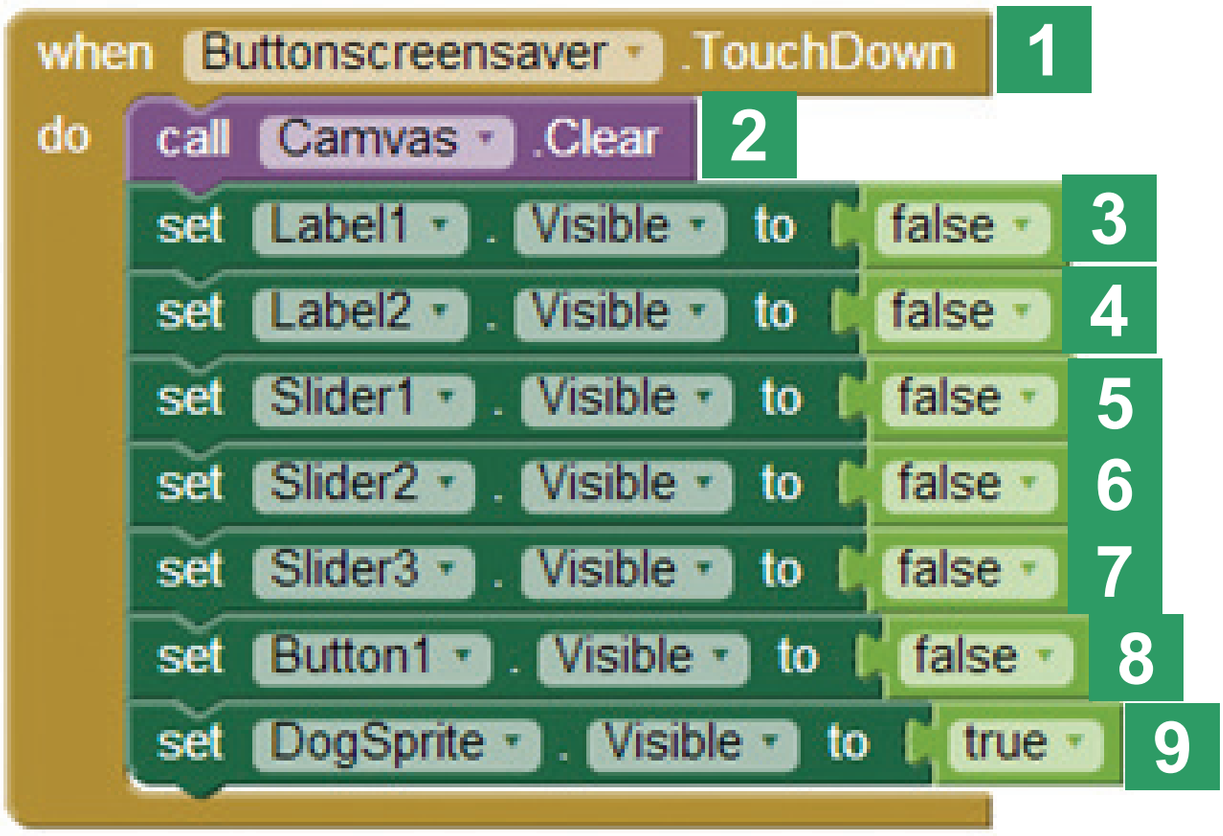
	Buttongame	BackgroundColor: Green Shape: Rounded Text: Game
	ButtonMO	BackgroundColor: Red Shape: Rounded Text: Μέσος Όρος
	Label1	Visible: Hidden
	Slider1	MaxValue: 20.0 MinValue: 1.0 ThumbPosition: 5 Visible: hidden Width: Fill parent
	Slider2	MaxValue: 20.0 MinValue: 1.0 ThumbPosition: 10 Visible: hidden Width: Fill parent
	Slider3	MaxValue: 20.0 MinValue: 1.0 ThumbPosition: 15

User Interface	**Label**	
User Interface	**Button**	
Drawing and Animation	**Canvas**	
Sensors	**Accelerometer Sensor**	
Drawing and Animation	**ImageSprite**	
Media	**Sound**	

		Visible: hidden Width: Fill parent
	Label2	Visible: Hidden
	Button1	Text: Τελικό αποτέλε-σμα TextAlignment: center Visible: hidden Width: Fill parent
	Canvas	Paint Color: Blue Width: Fill Parent Height: Fill Parent
	AccSensor	
	DogSprite	Interval: 10 Picture: Sleepy Dog.jpg Rotates: (No) Visible: (Yes)
	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.



1 when Buttonscreesaver .TouchDown

2 do call Canvas .Clear

3 set Label1. Visible to false

4 set Label2. Visible to false

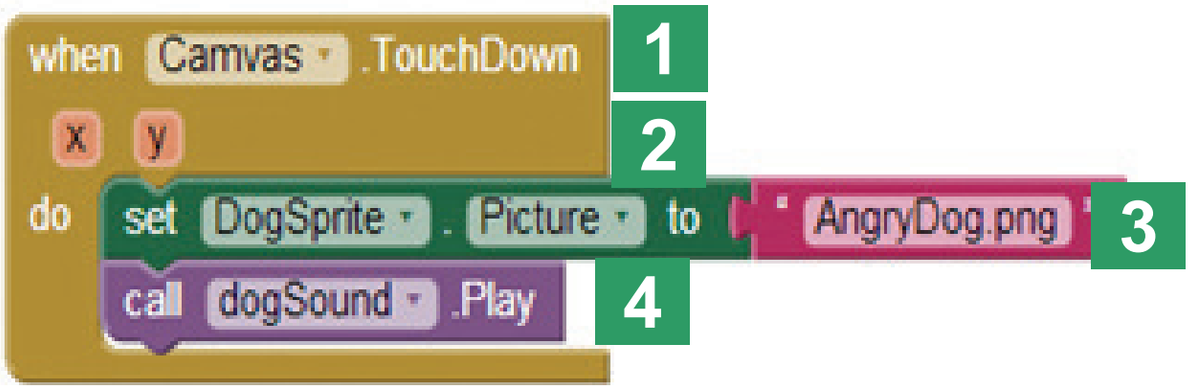
5 set Slider1. Visible to false

6 set Slider2. Visible to false

7 set Slider3. Visible to false

8 set Button1. Visible to false

9 set DogSprite Visible to false

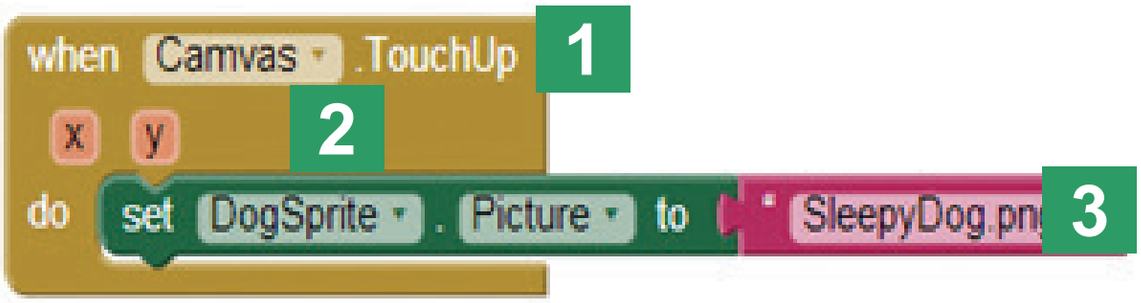


1 when Camvas .TouchDown

2 x y

3 do set DogSprite . Picture to AngryDog.png

4 call dogSound . Play



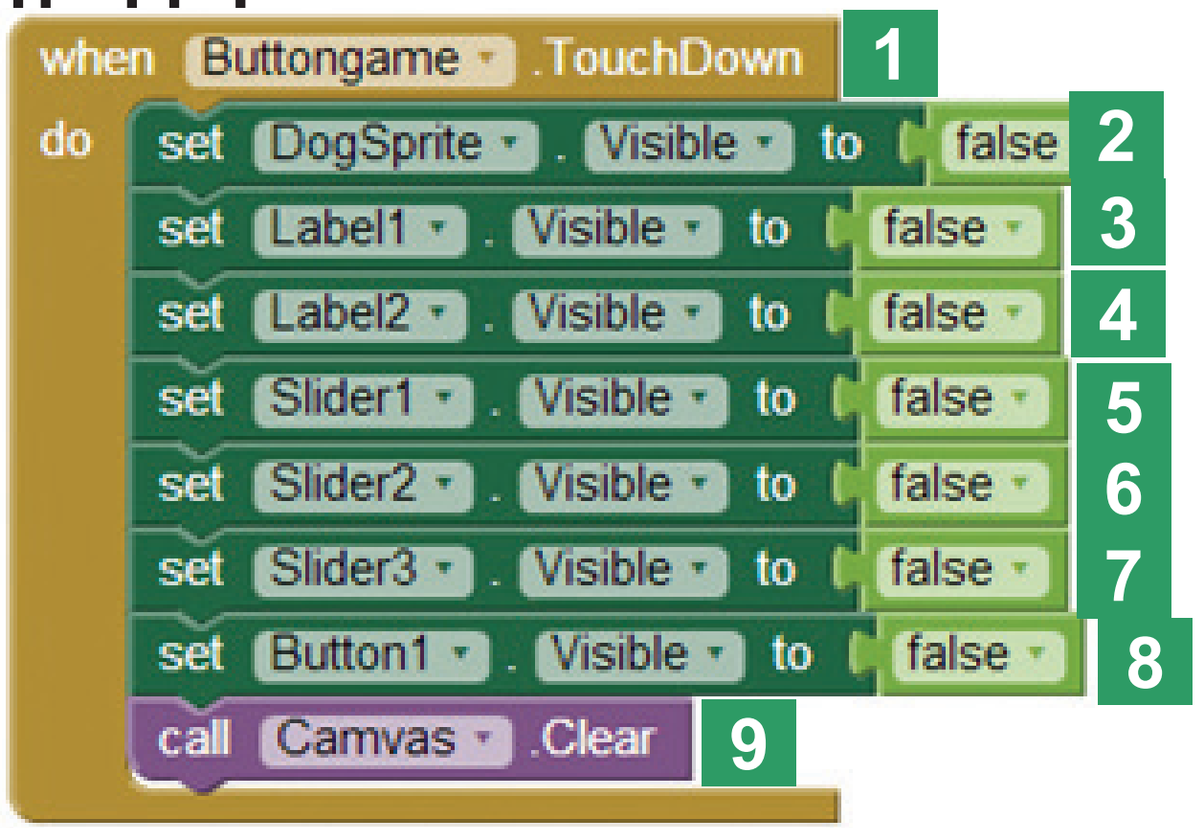
1 when Camvas .TouchUp

2 x y

3 do set DogSprite . Picture to SleepyDog.png

(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.



1 when Buttongame .TouchDown

2 do set Dogsprite . Visible to false

3 set Label1 . Visible to false

4 set Label2 . Visible to false

5 set Slider1 . Visible to false

6 set Slider2 . Visible to false

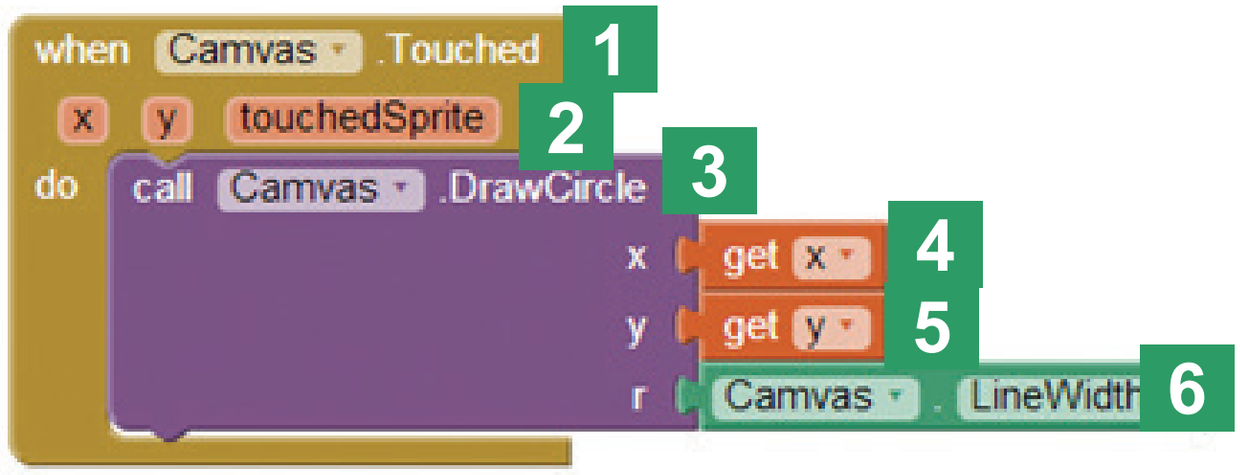
7 set Slider3 . Visible to false

8 set Button1 . Visible to false

9 call Camvas . Clear

1 when AccSensor . Shaking

2 do call Camvas . Clear



1 when Camvas . Touched

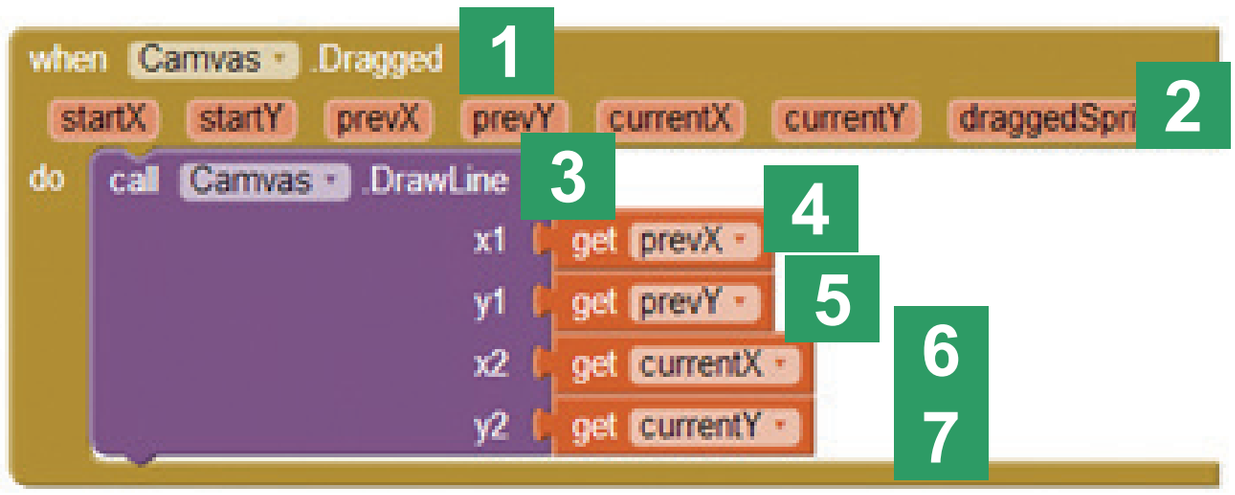
2 x y touchedSprite

3 do call Camvas . DrawCircle

4 x get x

5 y get y

6 r Camvas . LineWidth



1 when Camvas . Dragged

2 startX startY prevX prevY currentX currentY draggedSprite

3 do call Camvas . DrawLine

4 x1 get prevX

5 y1 get prevY

6 x2 get currentX

7 y2 get currentY

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

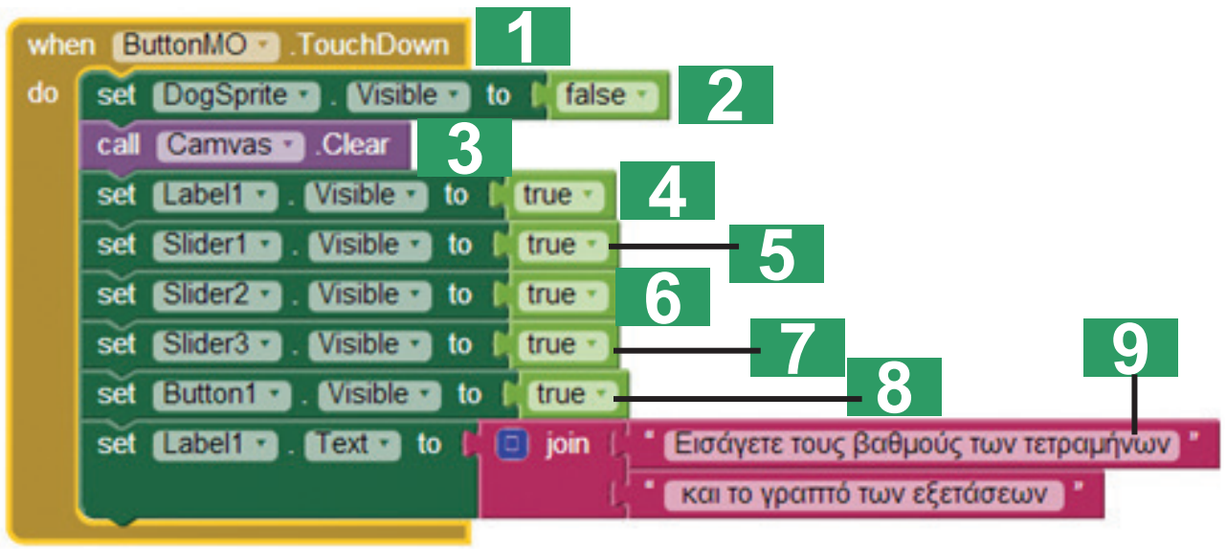
(Γ) Δημιουργούμε τα παρακάτω σενάρια για την επιλογή του υπολογισμού του Μέσου Όρου (ΜΟ) και των αποτελεσμάτων προαγωγής του μαθητή σε ένα μάθημα.

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.

1 initialize global A to 0
2 initialize global B to 0
3 initialize global G to 0
4 initialize global mo to 0



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.

1 when ButtonMO . TouchDown

2 do set DogSprite . Visible to false

3 call Camvas . Clear

4 set Label1 . Visible to true

5 set Slider1 . Visible to true

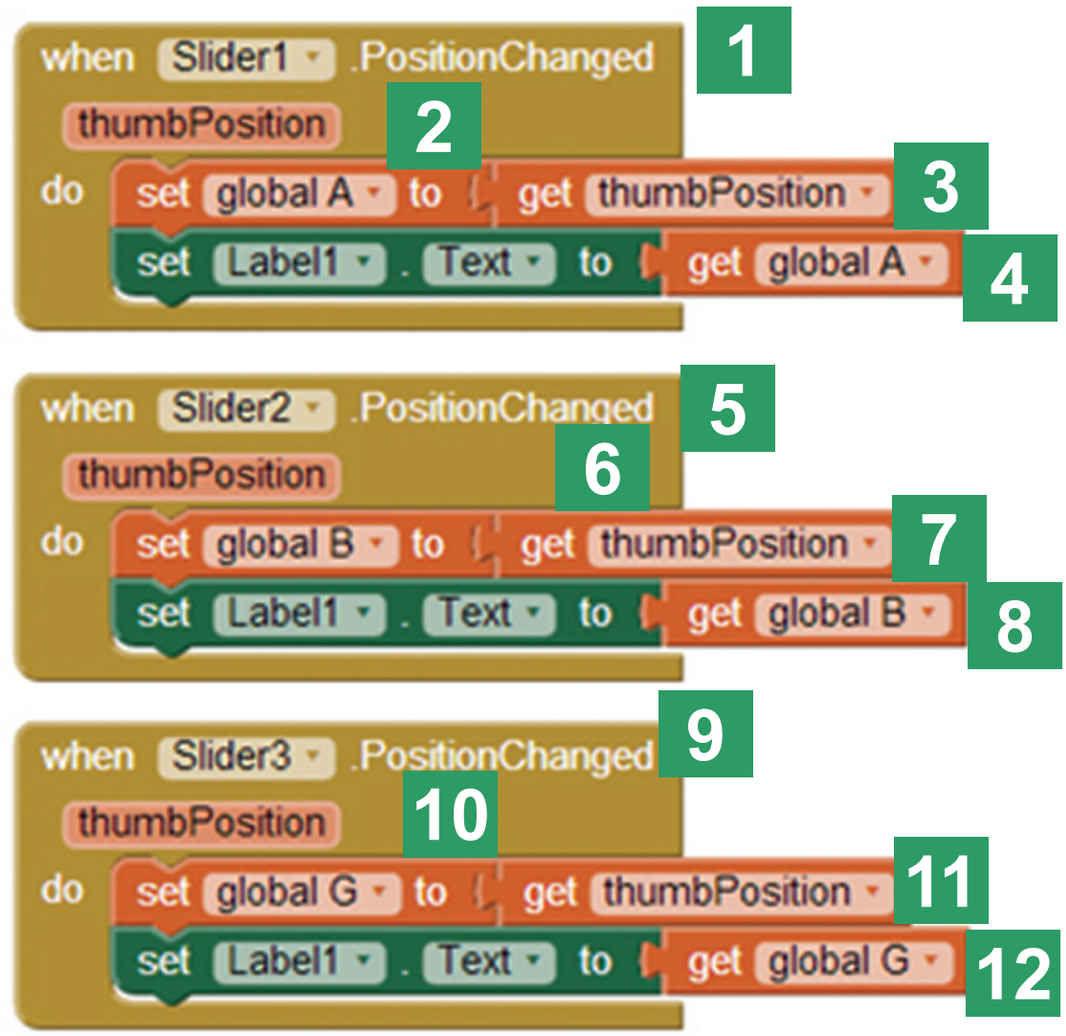
6 set Slider2 . Visible to true

7 set Slider3 . Visible to true

8 set Button1 . Visible to true

9 set Label1 . Text to join

Εισάγετε τους βαθμούς των τε-
τραμήνων και το γραπτό των
εξετάσεων



```
1  when Slider1 . PositionChanged
2  thumbPosition
3  do set globalA to
       get thumbPosition
4  set Label1 . Text to get globalA
5  when Slider2 . PositionChanged
6  thumbPosition
7  do set globalB to
       get thumbPosition
8  set Label1 . Text to get globalB
9  when Slider3 . PositionChanged
10 thumbPosition
11 do set globalG to
       get thumbPosition
12 set Label1 . Text to    get globalG
```

Με τον προηγούμενο 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

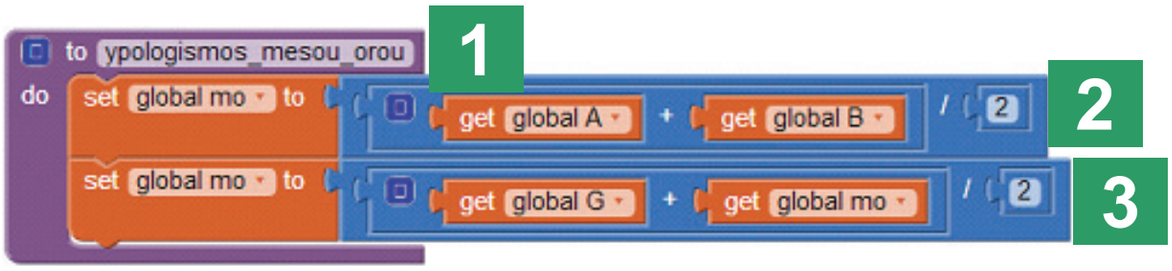
Όταν πατήσουμε το κουμπί «Τελικό αποτέλεσμα», τότε καλούμε τις Procedures (διαδικασίες) για τον υπολογισμό του μέσου όρου του μαθήματος και για την εμφάνιση του τελικού αποτελέσματος προαγωγής ή απόρριψης του μαθητή στο μάθημα. Αξίζει να σημειωθεί ότι κατά την κλήση της διαδικασίας «teliko_apotelesma»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

1 when Button1 . TouchDown

2 do call ypologismos_mesou_orou

3 call teliko_apotelesma

4 gmo get global mo



1 to ypologismos_mesou_orou

2 do set global mo to
get globalA + get globalB

3 set global mo to
get globalG + global mo

1 to teliko_apotelesma gmo

2 do set Label1 . Text to join
Ο μέσος όρος προφορικών και γραπτού είναι
get gmo

3 if get gmo ≥ 10

4 then set Label2 . Text to Ο μαθητής περνάει το μάθημα

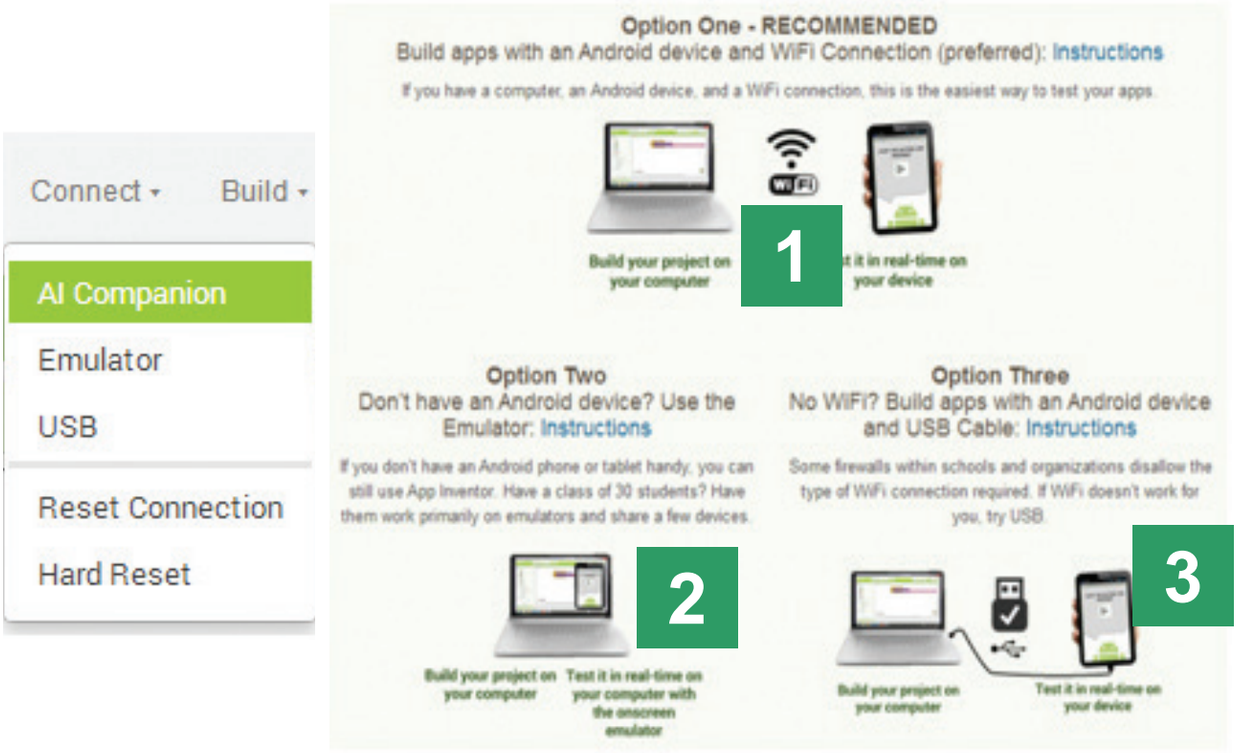
5 else set Label2 . Text to Ο μαθητής δεν περνάει το μάθημα

6 set Label2 . Visible to True

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)

1 Android Device & Wi Fi Connection

2 Emulator

3 Android Device & USB Cable

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.

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**.

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 ανήκει. Πρώτα ορίζονται οι κλάσεις και μετά δημιουργούνται τα αντικείμενα. Τα χαρακτηριστικά μιας κλάσης αντικειμένων ονομάζονται **ιδιότητες (properties)** και οι διαδικασίες που ορίζουν τις συμπεριφορές της ονομάζονται **μέθοδοι (methods)**. Οι μέθοδοι στις οποίες εκτελούνται μόνο εντολές και δεν επιστρέφεται κάποια τιμή ονομάζονται **διαδικασίες (procedures)**, ενώ οι μέθοδοι στις οποίες επιστρέφεται κάποια τιμή ονομάζονται **συναρτήσεις (functions)**. Για παράδειγμα, σε ένα πρόγραμμα προσομοίωσης ρομποτικών συσκευών εξερεύνησης μπορούμε να ορίσουμε ως κλάση το «ρομπότ» και στη συνέχεια να

δημιουργήσουμε αντικείμενα ρομπότ για διάφορες μορφές εξερεύνησης. (π.χ. ρομπότ εξερεύνησης βυθού, ρομπότ εξερεύνησης ηφαιστείου).

Δημοφιλείς γλώσσες αντικειμενοστρεφούς προγραμματισμού είναι η Java, η C++ και η Python.

Κλάση: ρομπότ
Ιδιότητες: θερμοκρασία, θέση, ταχύτητα
Μέθοδοι:
- εκκίνηση εξερεύνησης
- έλεγχος τρέχουσας θερμοκρασίας
- αναφορά τρέχουσας θέσης
- ορισμός ταχύτητας

Το περιβάλλον προγραμματισμού Alice

Το **Alice** είναι ένα ελεύθερα διαθέσιμο και καινοτόμο 3D (τρισδιάστατο) περιβάλλον προγραμματισμού που καθιστά εύκολη τη δημιουργία κινούμενων γραφικών (animation) για την αφήγηση μιας ιστορίας, την ανάπτυξη διαδραστικών παιχνιδιών ή τη δημιουργία βίντεο που μπορεί να διαμοιραστεί στο Διαδίκτυο. Ακολουθεί την αντικειμενοστρεφή προσέγγιση προγραμματισμού. Στο Alice, 3D αντικείμενα (π.χ. σκηνικά, άνθρωποι, ζώα, φυτά, οχήματα) σχηματίζουν έναν εικονικό κόσμο και ο προγραμματιστής δημιουργεί οπτικά ένα πρόγραμμα με σύρσιμο και ταίριασμα κατάλληλων πλακιδίων (tiles ή blocks) για τον ορισμό των ιδιοτήτων, των συμπεριφορών και των

αλληλεπιδράσεων των παραπάνω αντικειμένων. Τα αντικείμενα αποτελούν στιγμιότυπα κλάσεων που οργανώνονται με σχέσεις ιεραρχίας μεταξύ τους και στα οποία ισχύουν οι αρχές της κληρονομικότητας. Επίσης, στο Alice έχουμε **προγραμματισμό οδηγούμενο από γεγονότα (event-driven programming)**. Κάθε φορά που ο χρήστης κάνει κλικ με το ποντίκι ή πατάει ένα πλήκτρο, δημιουργείται ένα γεγονός που προκαλεί μια απάντηση. Για παράδειγμα, αν κάνουμε «κλικ σε ένα όχημα» (γεγονός), αυτό «αρχίζει να κινείται» (απάντηση). Ο χειρισμός των γεγονότων γίνεται με κατάλληλες μεθόδους.

Στο επίσημο site του Alice http://www.alice.org μπορείτε να «κατεβάσετε» το αρχείο εγκατάστασής του, να βρείτε οδηγούς εκμάθησης, ομάδες συζητήσεων, δημοσιεύσεις κ.ά. Το Alice διατίθεται δωρεάν, αναπτύσσεται στη Java από το Πανεπιστήμιο Carnegie Mellon και στηρίζεται οικονομικά από σημαντικές εταιρείες του χώρου της Πληροφορικής όπως οι Oracle, Electronic Arts, Sun Microsystems, Intel και Microsoft.

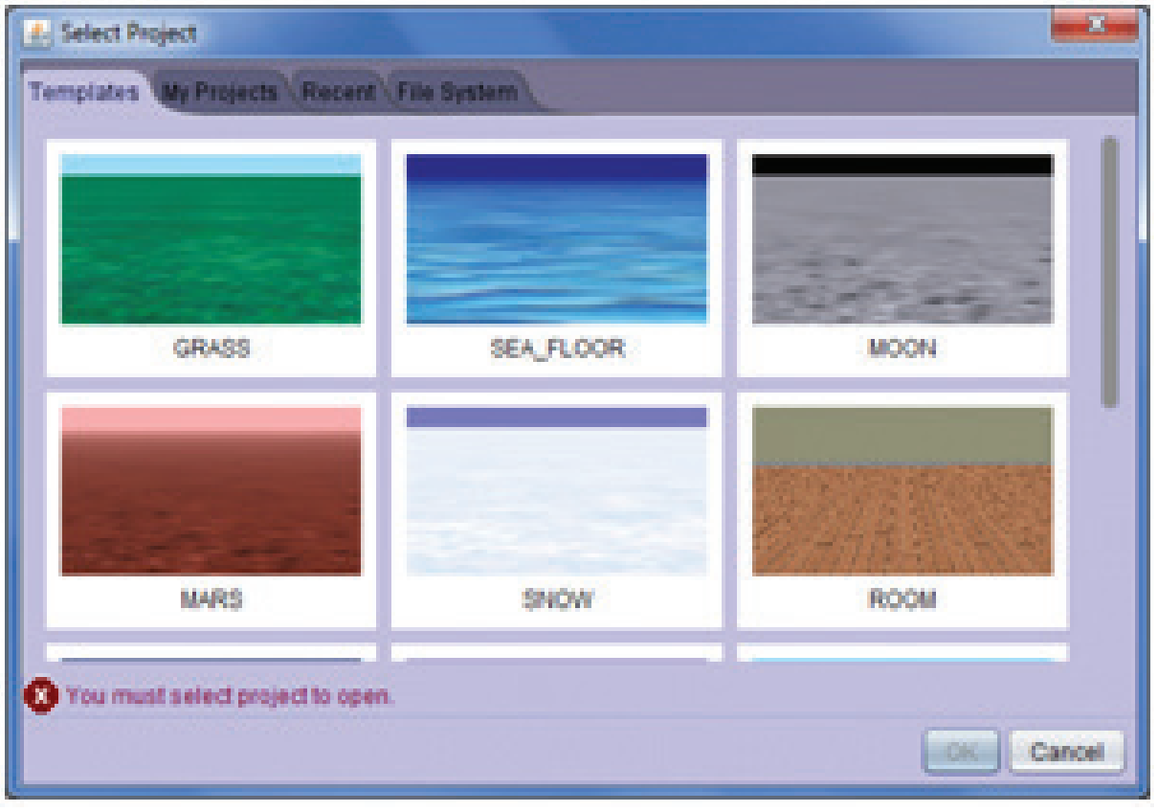
Διαδικασία δημιουργίας μιας εφαρμογής στο Alice.

Για να δημιουργήσουμε μια εφαρμογή στο Alice, ακολουθούμε τα παρακάτω βήματα:

1. Ενεργοποιούμε το Alice3, το οποίο έχουμε κατεβάσει από τον

ιστότοπο http://www.alice.org/ και το έχουμε εγκαταστήσει τοπικά στον υπολογιστή μας.

2. Στο πλαίσιο διαλόγου **Select Project** (εικόνα 7.14) επιλέγουμε την αρχική σκηνή (**Templates**) του εικονικού μας κόσμου (επιφάνεια και ατμόσφαιρα).



Εικόνα 7.14. Επιλογή σκηνής του εικονικού κόσμου

3. Στην οθόνη μας εμφανίζεται το παράθυρο της εικόνας 7.15 όπου

προγραμματίζουμε. Χωρίζεται σε 4 μέρη:

- √ χώρος Α: **Σκηνή (Scene)**, όπου αναπαριστάται ο εικονικός κόσμος που σχεδιάζουμε.
- √ χώρος Β: **Συντάκτης Κώδικα (Code Editor)**, όπου προγραμματίζουμε μεταφέροντας τις μεθόδους από τους χώρους Γ και Δ, αφού πρώτα επιλέξουμε τη σωστή καρτέλα (περιοχή Β1) στην οποία συντάσσουμε τον κώδικα.
- √ χώρος Γ: **Μέθοδοι (Procedures, Functions)**. Τις χρησιμοποιούμε, για να αποκτήσει συμπεριφορές το εκάστοτε αντικείμενο που έχουμε επιλέξει στο Γ1.
- √ χώρος Δ: **Μέθοδοι, Έλεγχος (Control)**. Τις χρησιμοποιούμε, όταν θέλουμε να ομαδοποιήσουμε έναν αριθμό μεθόδων βάσει κάποιου ελέγχου ή συνθήκης (π.χ. if_, while_).

4. Αρχικά όμως πρέπει να σχεδιάσουμε τον εικονικό μας κόσμο, ώστε να μπορούμε στη συνέχεια να τον προγραμματίσουμε. Επιλέγουμε στη σκηνή Α το κουμπί Setup Scene και μεταφερόμαστε στο αντίστοιχο παράθυρο (εικόνα 7.16). Χωρίζεται σε 3 μέρη και η αρίθμηση με την οποία αναφέρονται είναι και η σειρά που ακολουθούμε, για να σχεδιάσουμε τον κόσμο μας:

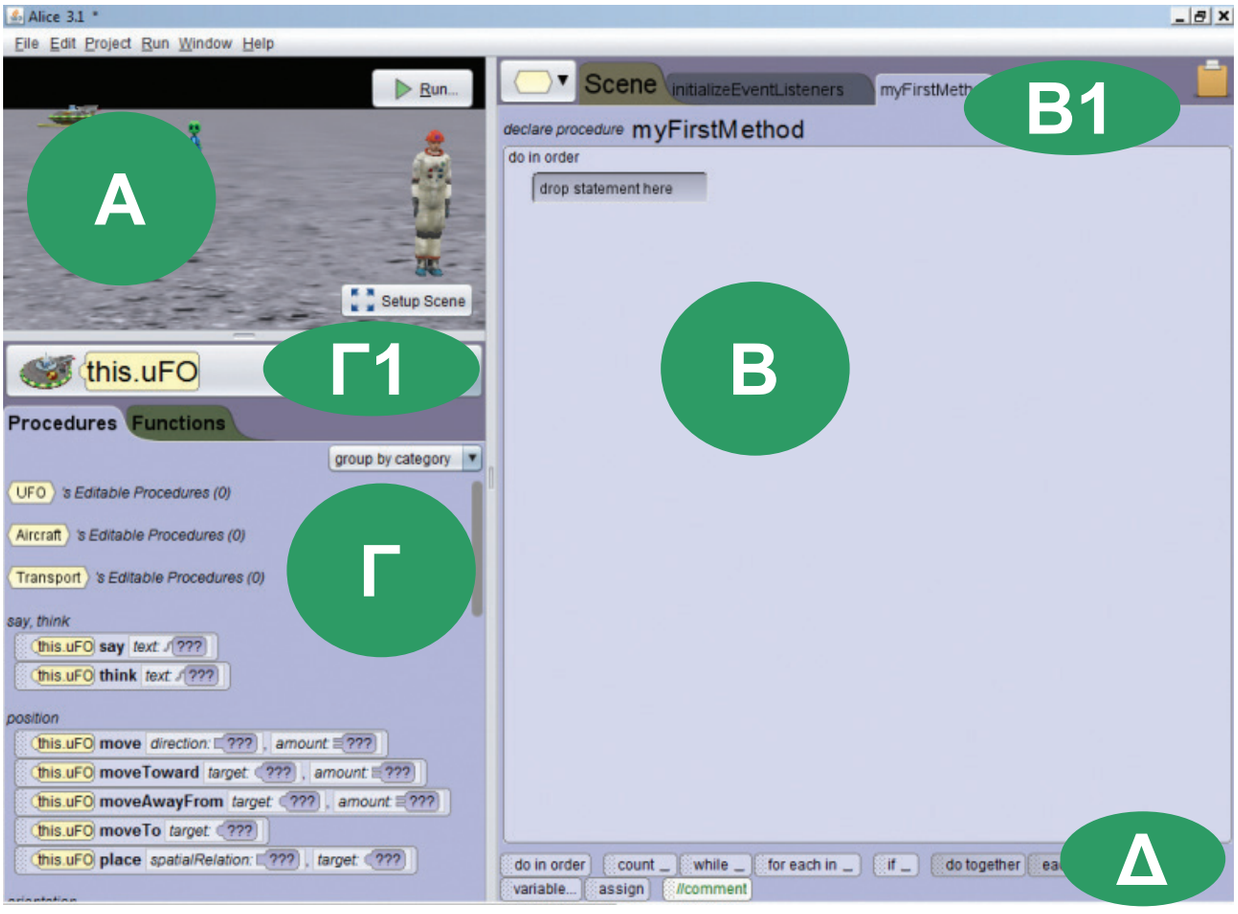
i. Αρχικά από τον χώρο Χ επιλέγουμε αντικείμενα και τα τοποθετούμε στη σκηνή (Ψ). Τα αντικείμενα είναι ομαδοποιημένα σε κατηγορίες - κλάσεις (Biped-δίποδα, Flyer-ιπτάμενα, Prop-υποστηρικτικά, Quadruped-τετράποδα, Swimmer-υποθαλάσσια και Transport-μέσα μεταφοράς). Για ορισμένα αντικείμενα (άνθρωποι) έχουμε τη δυνατότητα να επιλέξουμε την ενδυμασία τους.

ii. Τοποθετούμε τα αντικείμενα στην επιθυμητή θέση στη σκηνή Ψ.

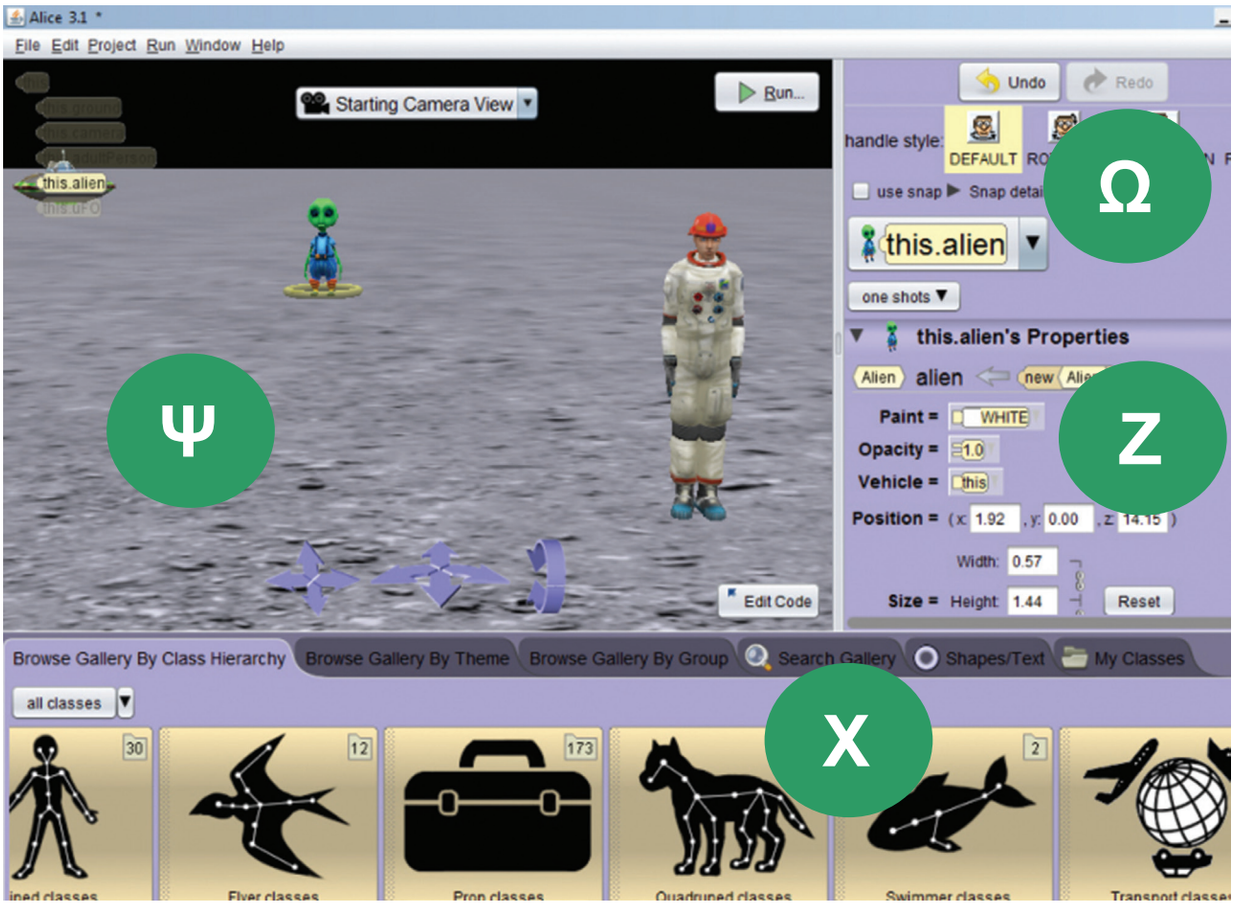
iii. Ρυθμίζουμε τις αρχικές ιδιότητες του κάθε αντικειμένου στο χώρο Z.

5. Για να επιστρέψουμε και να προγραμματίσουμε, επιλέγουμε το κουμπί Edit Code από τη σκηνή Ψ.

6. Οποιαδήποτε στιγμή επιθυμούμε να δούμε τα αποτελέσματα της εκτέλεσης του κώδικα επιλέγουμε από τη σκηνή Α το κουμπί Run.



Εικόνα 7.15. Παράθυρο ανάπτυξης του προγράμματος



Εικόνα 7.16. Παράθυρο σχεδίασης του εικονικού κόσμου

Project με την εφαρμογή Alice.

Στη συνέχεια θα δημιουργήσουμε μια ολοκληρωμένη εφαρμογή. Η περιγραφή του σεναρίου έχει ως εξής. Υπάρχει ένας αστροναύτης (**AdultPerson**) στη Σελήνη, αντιλαμβάνεται ότι πλησιάζει ένα σκάφος και στρέφεται προς εκείνη την

κατεύθυνση. Όμως το σκάφος είναι Άγνωστης Ταυτότητας Ιπτάμενο Αντικείμενο (**UFO**). Γίνεται η προσσελήνωση, κατεβαίνει ένας εξωγήινος (**Alien**) και πλησιάζει τον αστροναύτη. Ο αστροναύτης καλεί σε βοήθεια το κέντρο ελέγχου και ζητάει από τον χρήστη να τον απομακρύνει από τον Alien. Αφού δεν έχει επιτευχθεί ο επιθυμητός διάλογος μεταξύ τους, ο εξωγήινος απάγει τον αστροναύτη στο σκάφος και αναχωρούν για τον πλανήτη του.

Τα βήματα που πρέπει να ακολουθήσουμε είναι τα παρακάτω:

1. Ενεργοποιούμε το Alice και από τα templates επιλέγουμε το **Moon** (εικόνα 7.14). Για να μην ξεχάσουμε να αποθηκεύσουμε την εργασία μας, επιλέγουμε μενού **File→Save As** και για κάθε επόμενη φορά το File→Save.
2. Μεταφερόμαστε στο παράθυρο

σχεδίασης του κόσμου, πατώντας το κουμπί Setup Scene. Εισάγουμε τον αστροναύτη από την **κλάση Biped**, χρησιμοποιώντας τη μέθοδο **new Adult(...)**. Αφού του φορέσουμε τη στολή, τον τοποθετούμε στη σκηνή Ψ μπροστά και δεξιά. Τον μετονομάζουμε σε **astronaut** (εικόνα 7.16).

3. Εισάγουμε τον εξωγήινο από την **κλάση Biped**, χρησιμοποιώντας την μέθοδο **new Alien()**. Τον τοποθετούμε στη μέση της σκηνής και ορίζουμε την ιδιότητα Opacity (ορατότητα) σε 0.0. Αρχικά δεν φαίνεται ο alien, για να μπορέσουμε να τον εμφανίσουμε αργότερα και να δημιουργείται η εντύπωση ότι βγαίνει από το UFO.

4. Εισάγουμε το UFO από την **κλάση Transport** και την υποκλάση **Aircraft**, χρησιμοποιώντας τη μέθοδο **new UFO()**. Τον τοποθετούμε

στη σκηνή Ψ πίσω και αριστερά.

5. Μπορούμε να μετακινήσουμε, περιστρέψουμε, ανυψώσουμε, αλλάξουμε μέγεθος κ.ά. στα αντικείμενά μας με τις επιλογές Default, Rotation, Translation, Resize (Ω, εικόνα 7.16), καθώς επίσης και με τα μπλε βελάκια που βρίσκονται μπροστά και στο κέντρο της σκηνής Ψ.

6. Για να επιστρέφουμε στο αρχικό παράθυρο, πατάμε το κουμπί **Edit Code**.

7. Συντάσσουμε τον κώδικα της εικόνας 7.17 στη δική μας μέθοδο **MyFirstMethod** (Β1) Να σημειώσουμε ότι μετά από το σύρσιμο μιας μεθόδου στον συντάκτη κώδικα τροποποιούμε τις παραμέτρους της. Επιπλέον δυνατότητες ορισμού παραμέτρων μάς δίνει η επιλογή του **add detail**.

8. Για να δημιουργήσουμε μια procedure, επιλέγουμε το εξάγωνο από

την περιοχή B1→UFO →Add UFO Procedure. Πληκτρολογούμε το όνομα της διαδικασίας abduction, οπότε δημιουργούνται 2 νέες καρτέλες (UFO, abduction). Εισάγουμε τις μεθόδους όπως φαίνονται στην εικόνα 7.18.

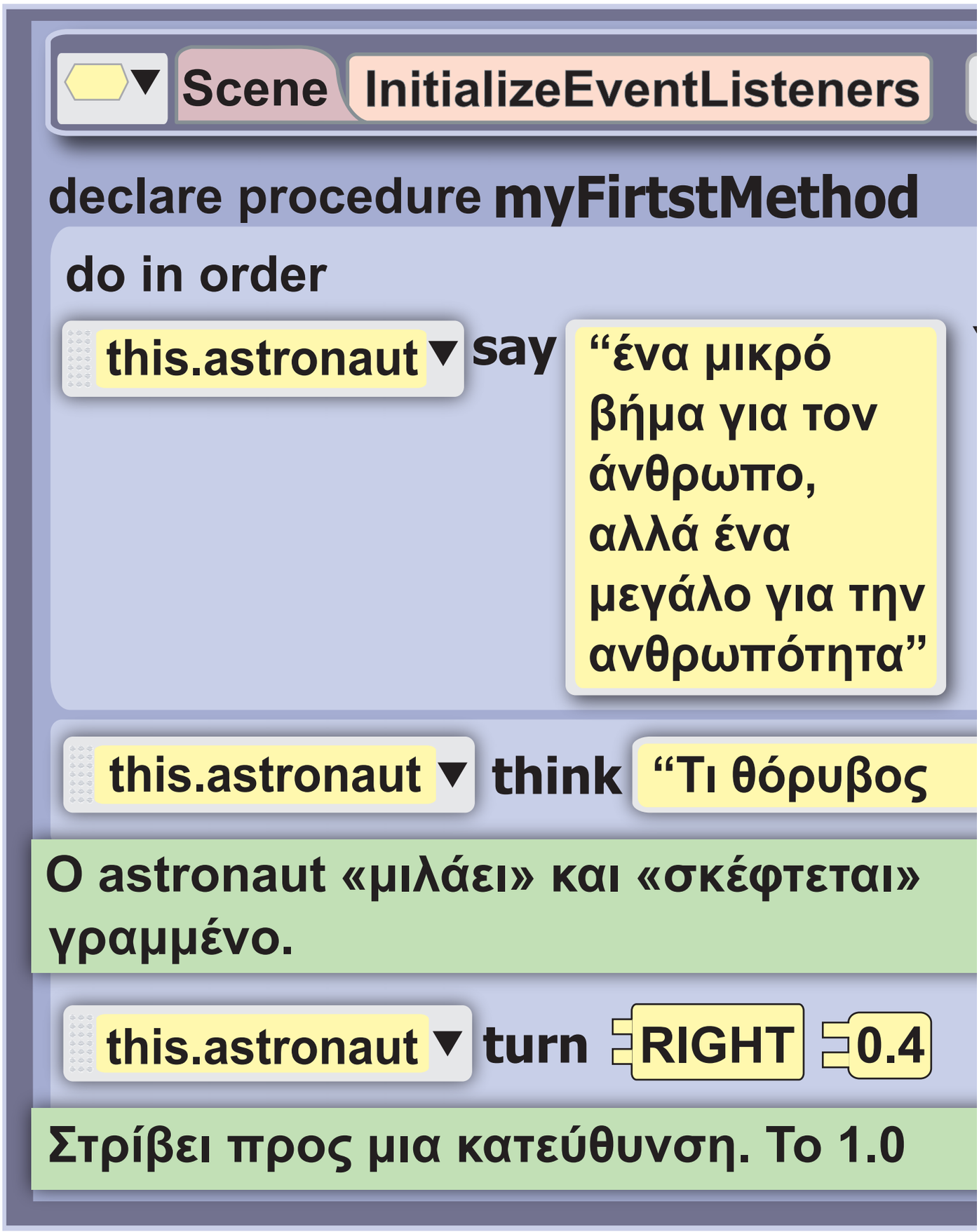
9. Για να δώσουμε τη δυνατότητα να χειρίζεται ο χρήστης με το δεξί και αριστερό βελάκι του πληκτρολογίου τον astronaut, επιλέγουμε καρτέλα InitializeEventListeners→ Add Event Listeners→ Keyboard→ addArrowKeyPressListeners. Στη συνέχεια μεταφέρουμε τη μέθοδο ελέγχου **if_** από τον χώρο Δ στον συντάκτη κώδικα (B) και δημιουργούμε τον κώδικα όπως φαίνεται στην εικόνα 7.19. Η λογική που εφαρμόζουμε είναι «Αν (if) πατηθεί το αριστερό πλήκτρο, τότε (then) να κινηθεί ο astronaut αριστερά, αλλιώς, αν (if) πατηθεί το δεξί βελάκι,

τότε (then) να κινηθεί ο astronaut δεξιά, αλλιώς (else) να εκφράσει τη δυσαρέσκειά του.

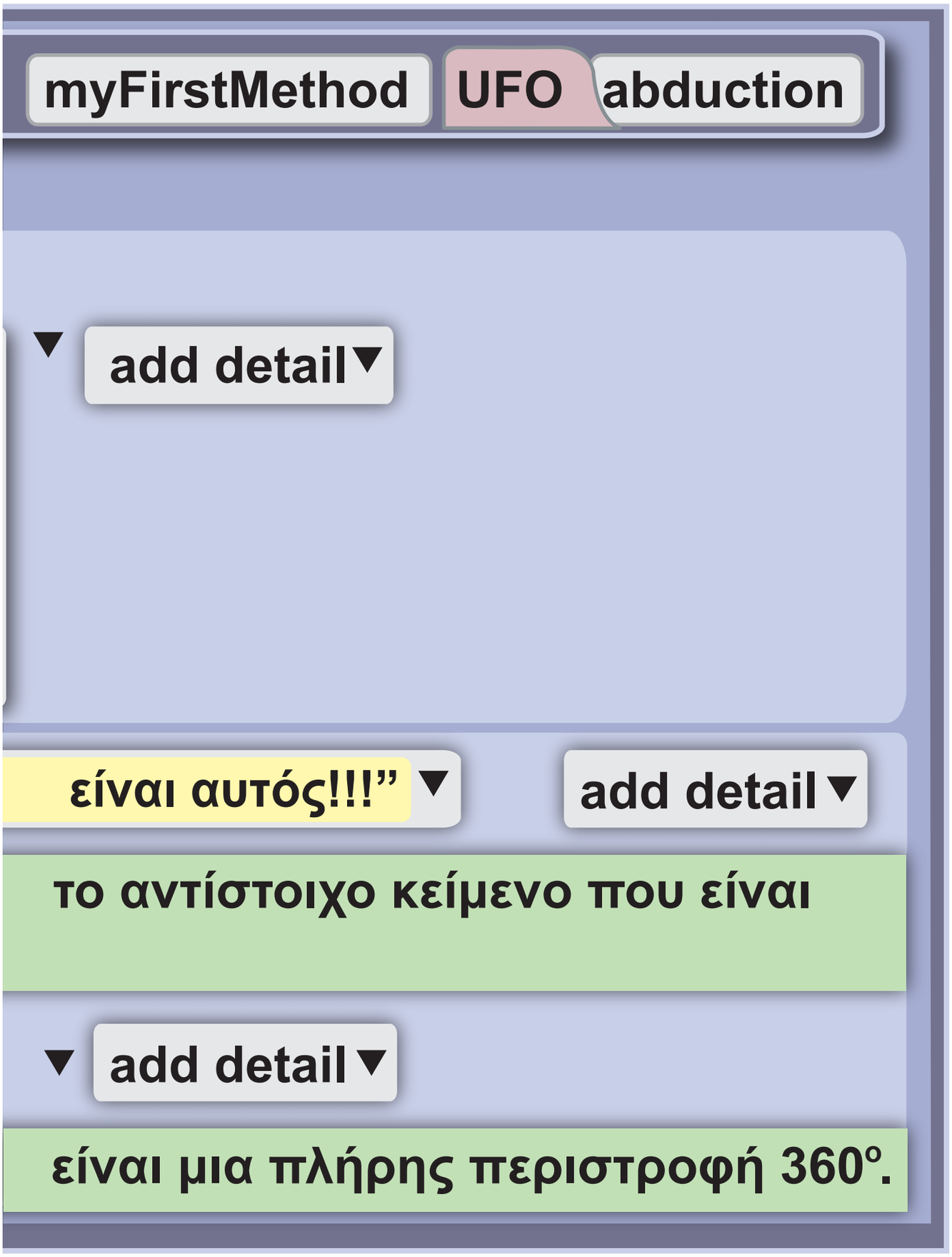
Ο λόγος που επιλέξαμε για να υλοποιηθεί αυτός ο κώδικας σε αυτή την καρτέλα και όχι εντός της μεθόδου MyFirstMethod είναι ότι ο κώδικας που γράφεται σε αυτό τη σημείο εκτελείται από την αρχή εκτέλεσης της εφαρμογής μέχρι το τέλος αυτής.

10. Οποιαδήποτε στιγμή μπορούμε να εκτελέσουμε τον κώδικα και να ελέγξουμε τα αποτελέσματα που εμφανίζονται με τη μορφή βίντεο.

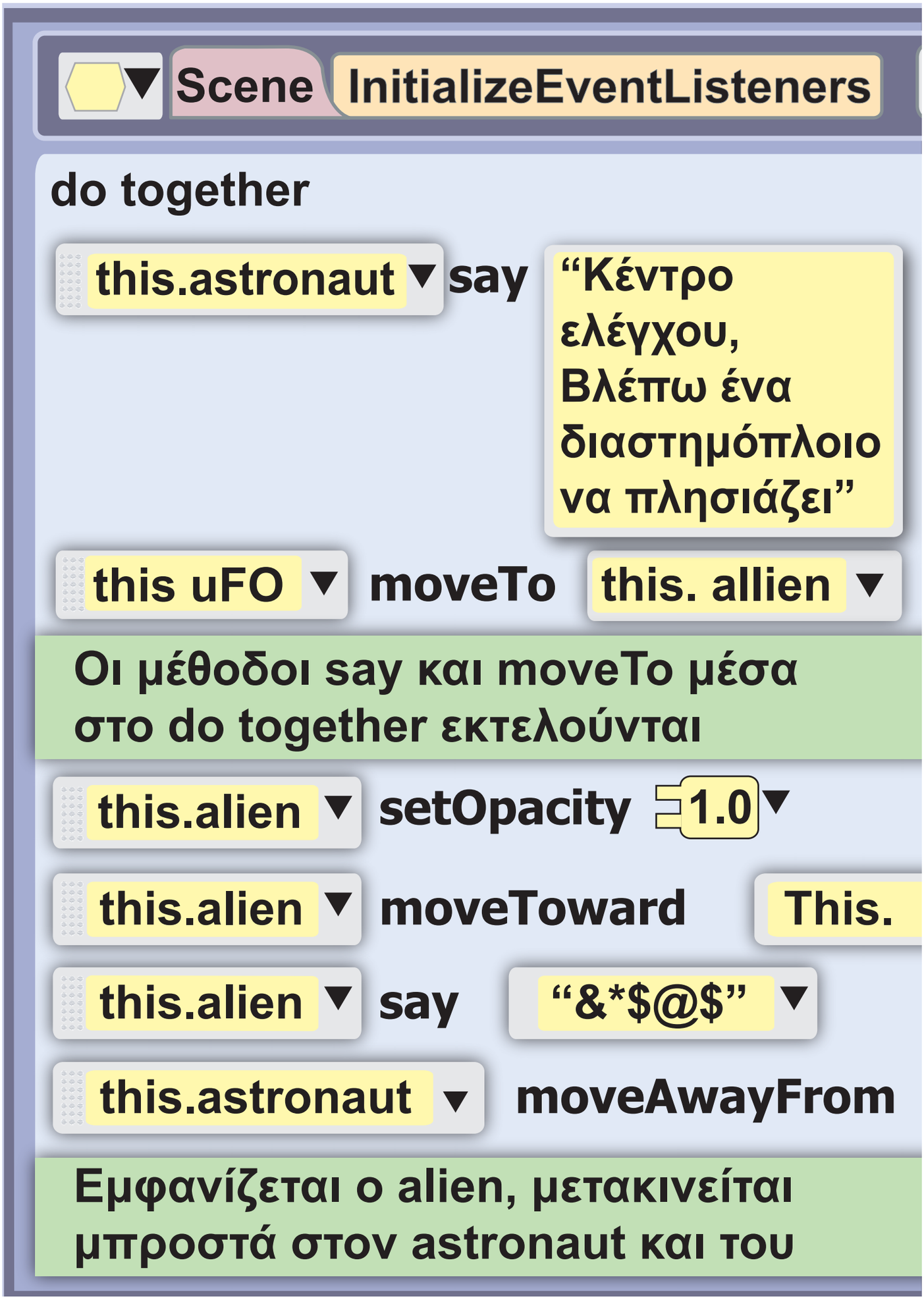
11. Δεν ξεχνάμε κατά διαστήματα και στο τέλος να αποθηκεύσουμε την εφαρμογή μας.

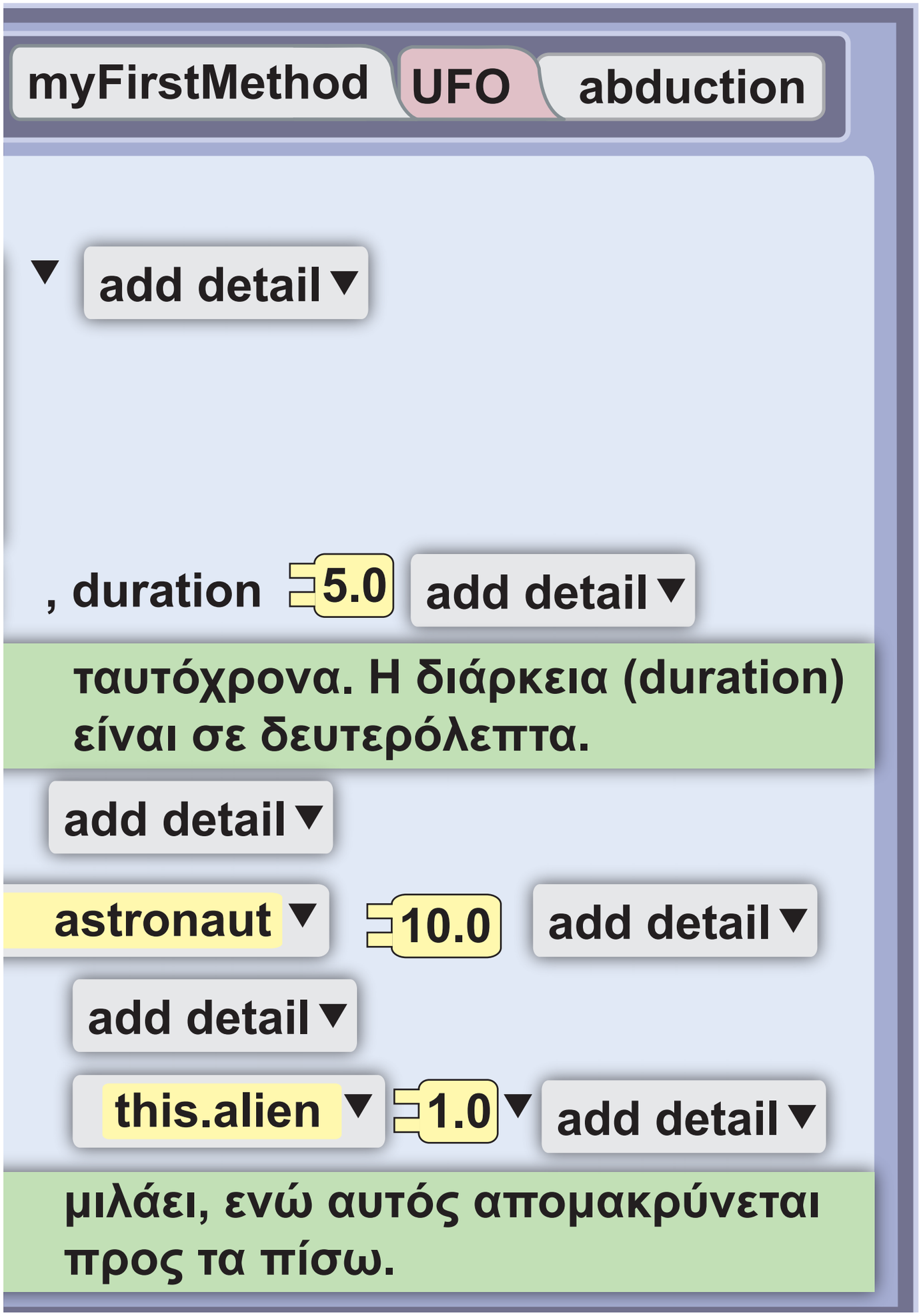


Εικόνα 7.17. Ο κώδικας της μεθόδου



MyFirstMethod





myFirstMethod UFO abduction

add detail ▼

add detail ▼

(δομή count up to)

χρήστη βελάκια !!!”

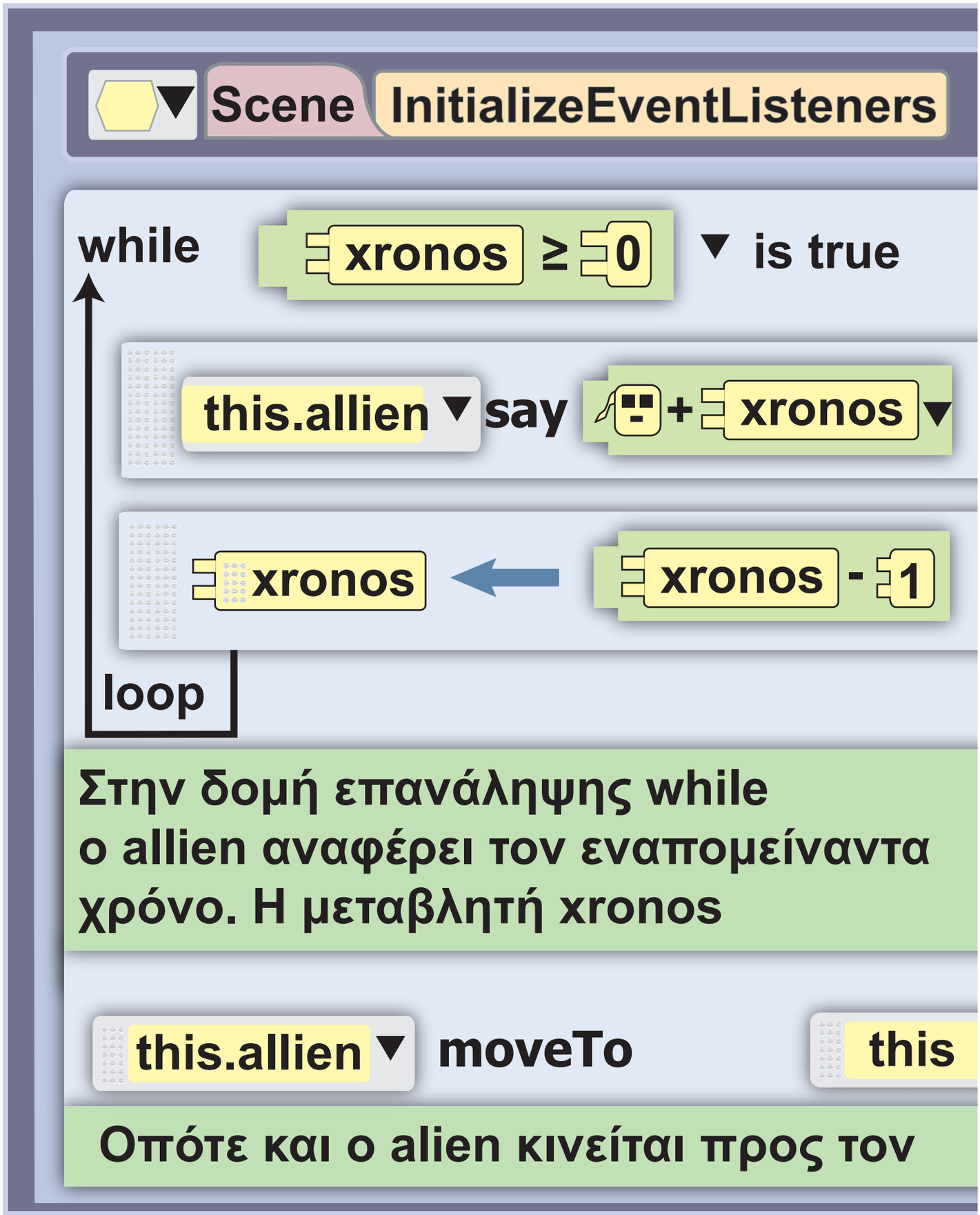
, duration 2.0

add detail ▼

add detail ▼

10

Μετά την κλήση για βοήθεια ορίζουμε μια μεταβλητή xronos με αρχική τιμή 10.



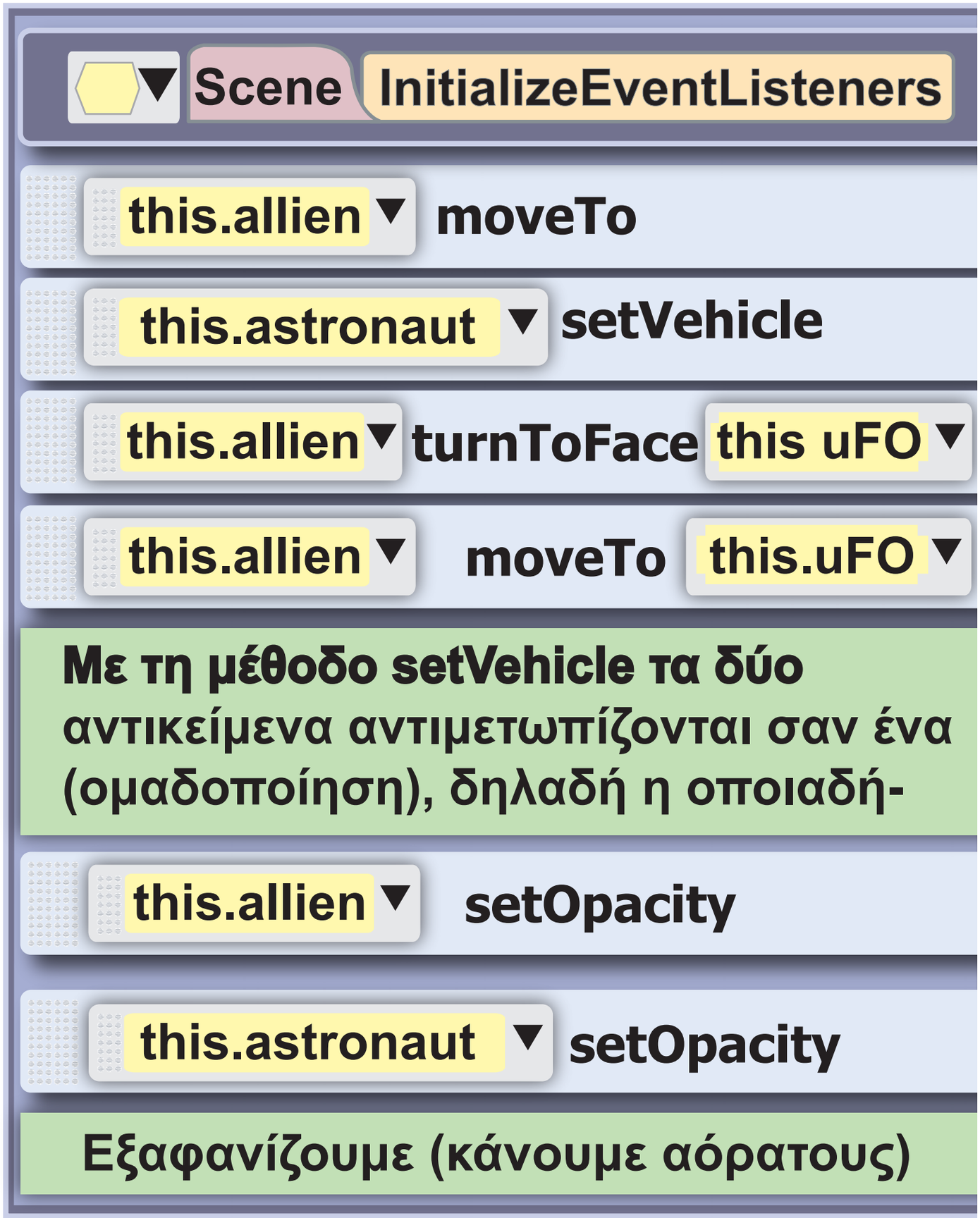
myFirstMethod | UFO | abduction

add detail ▼

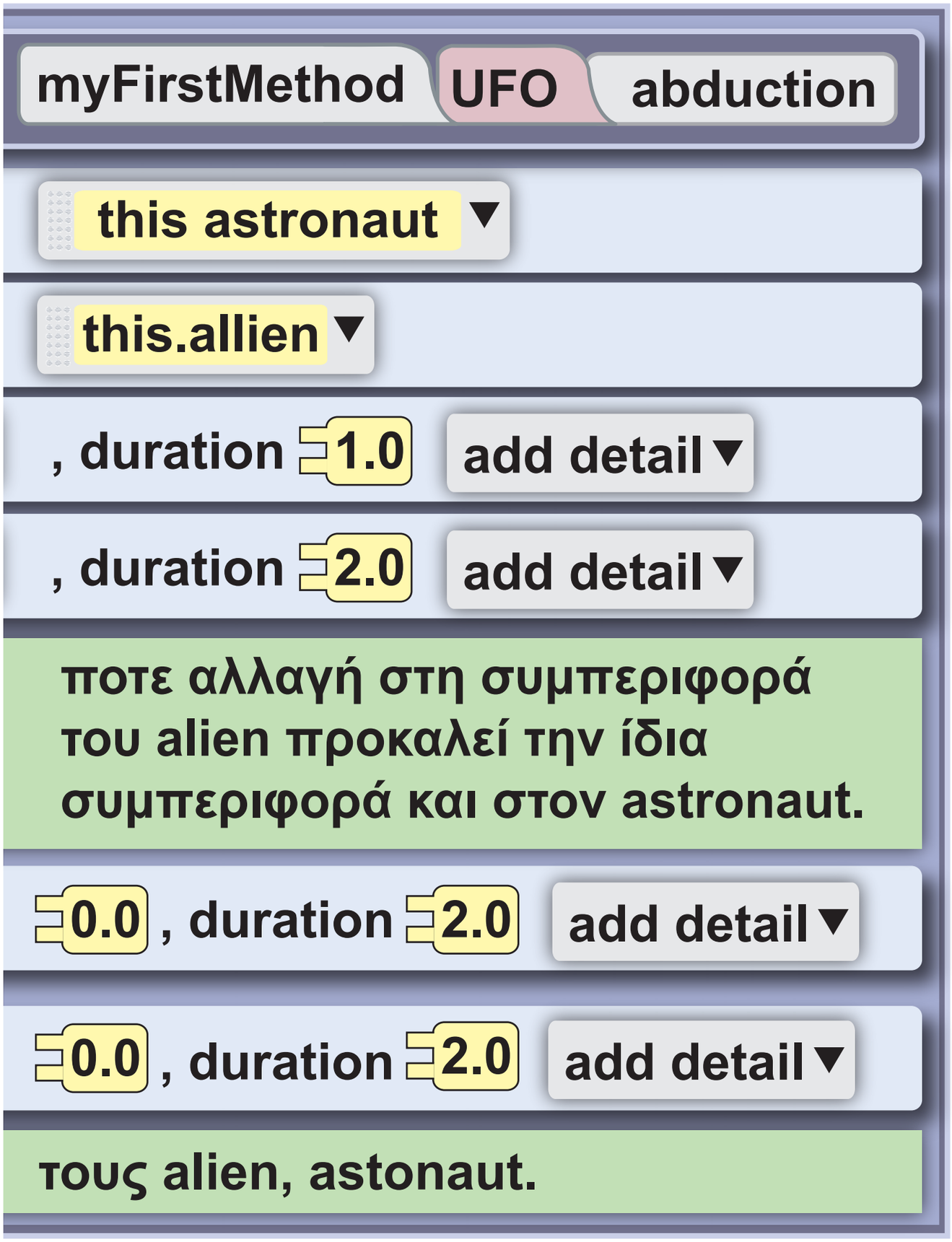
μειώνεται κατά ένα σε κάθε επανάληψη, μέχρι να μηδενιστεί.

astronaut ▼ | add detail ▼

astronaut.

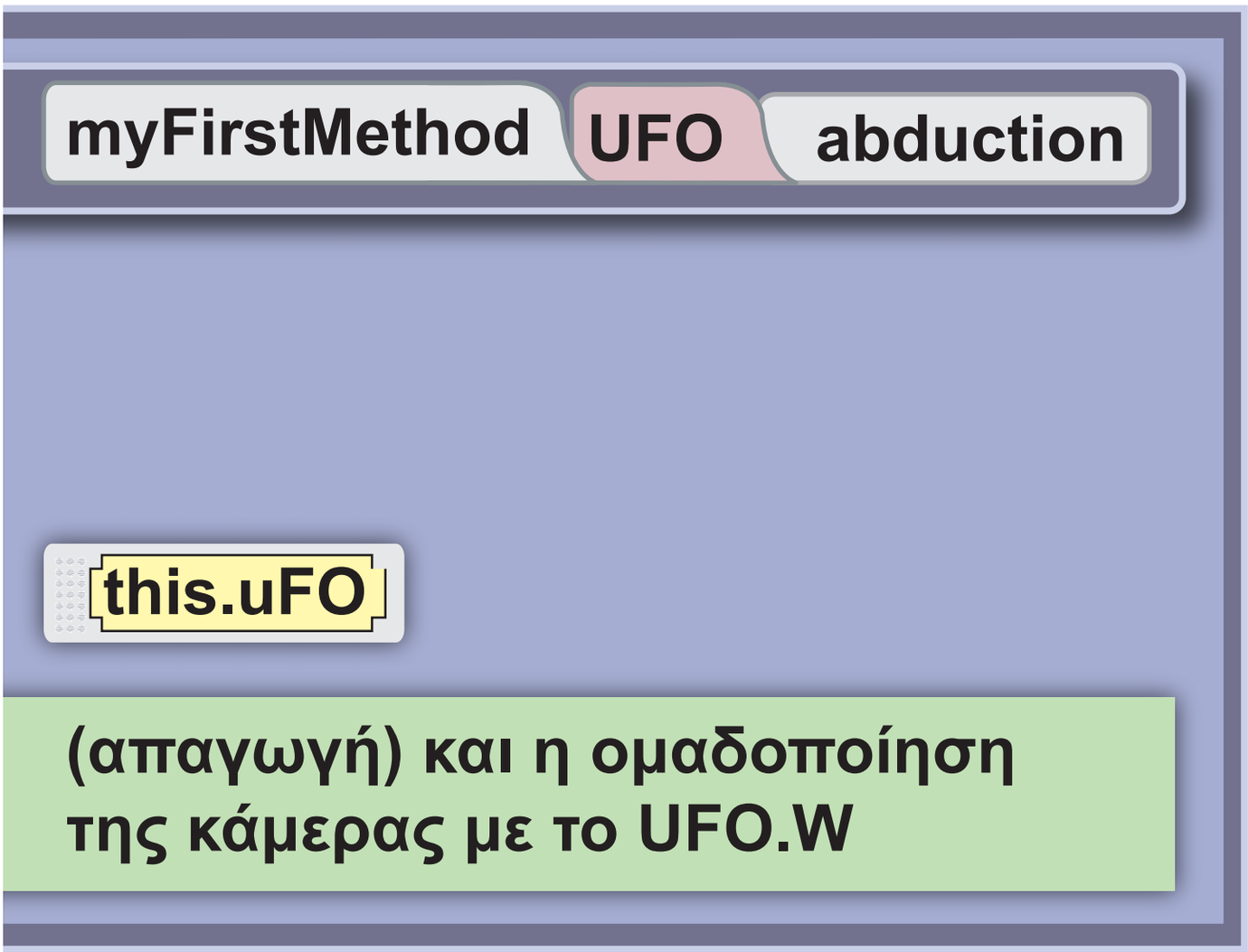


Εικόνα 7.17. Ο κώδικας της μεθόδου

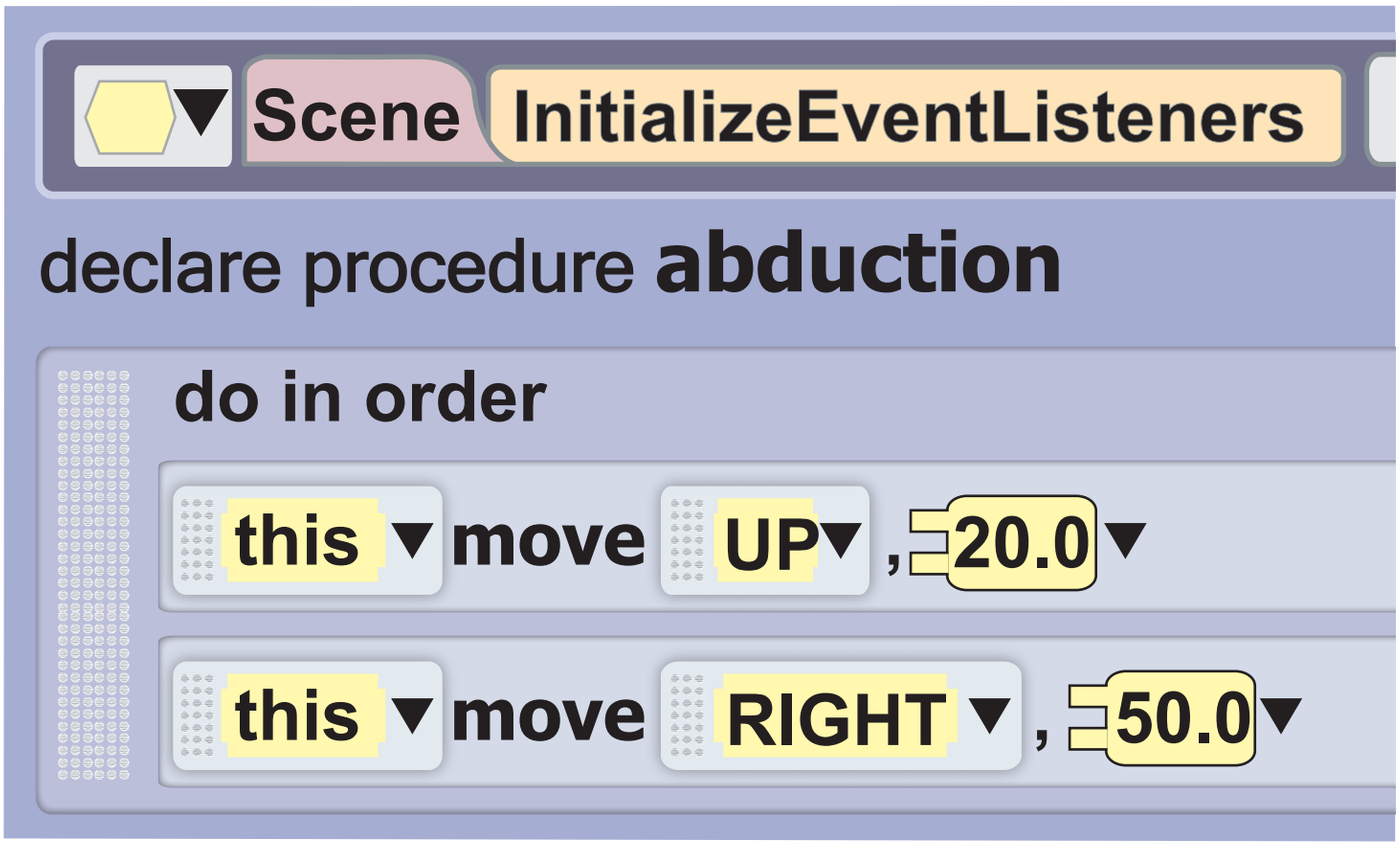


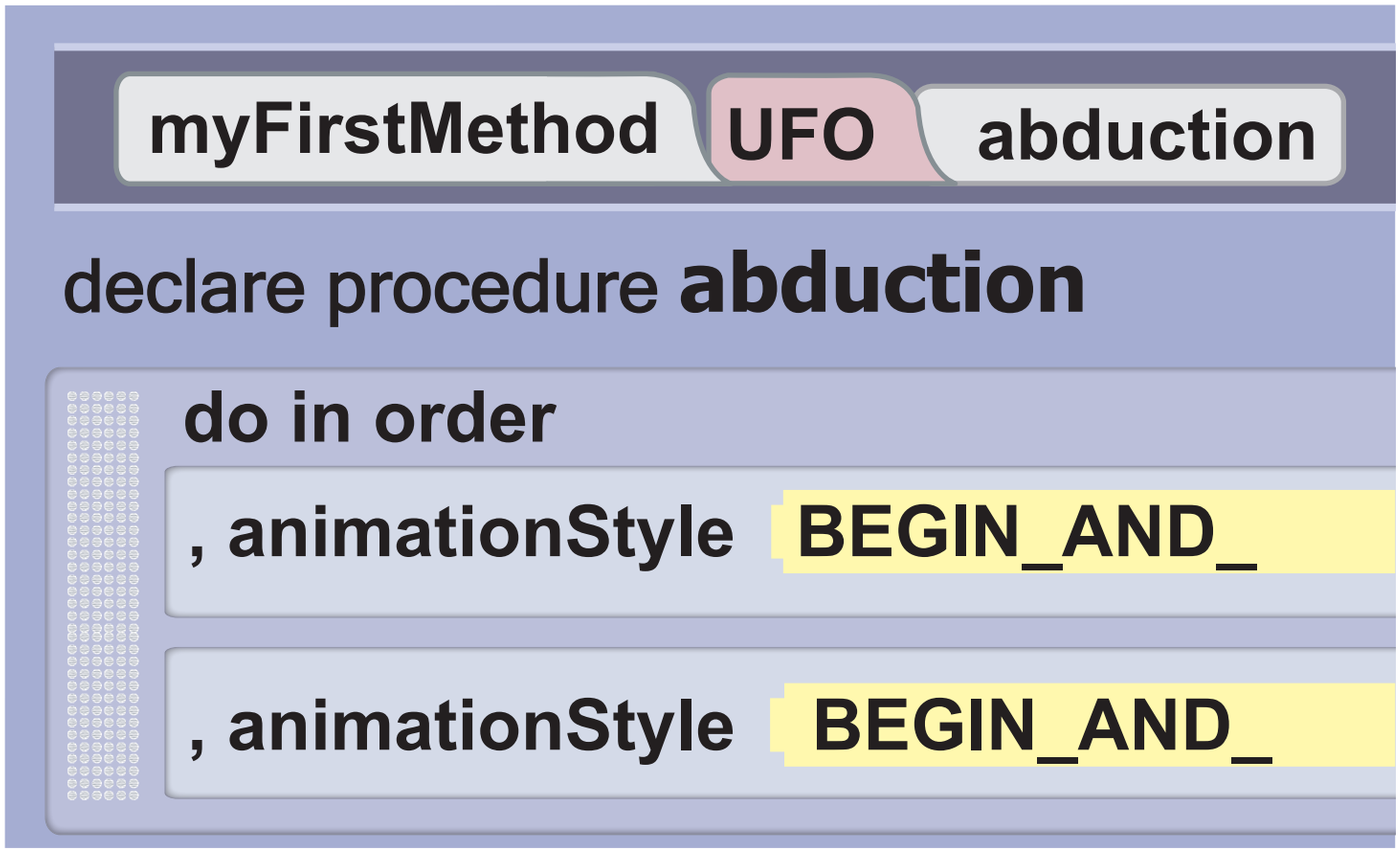
MyFirstMethod

Εικόνα 7.17. Ο κώδικας της μεθόδου

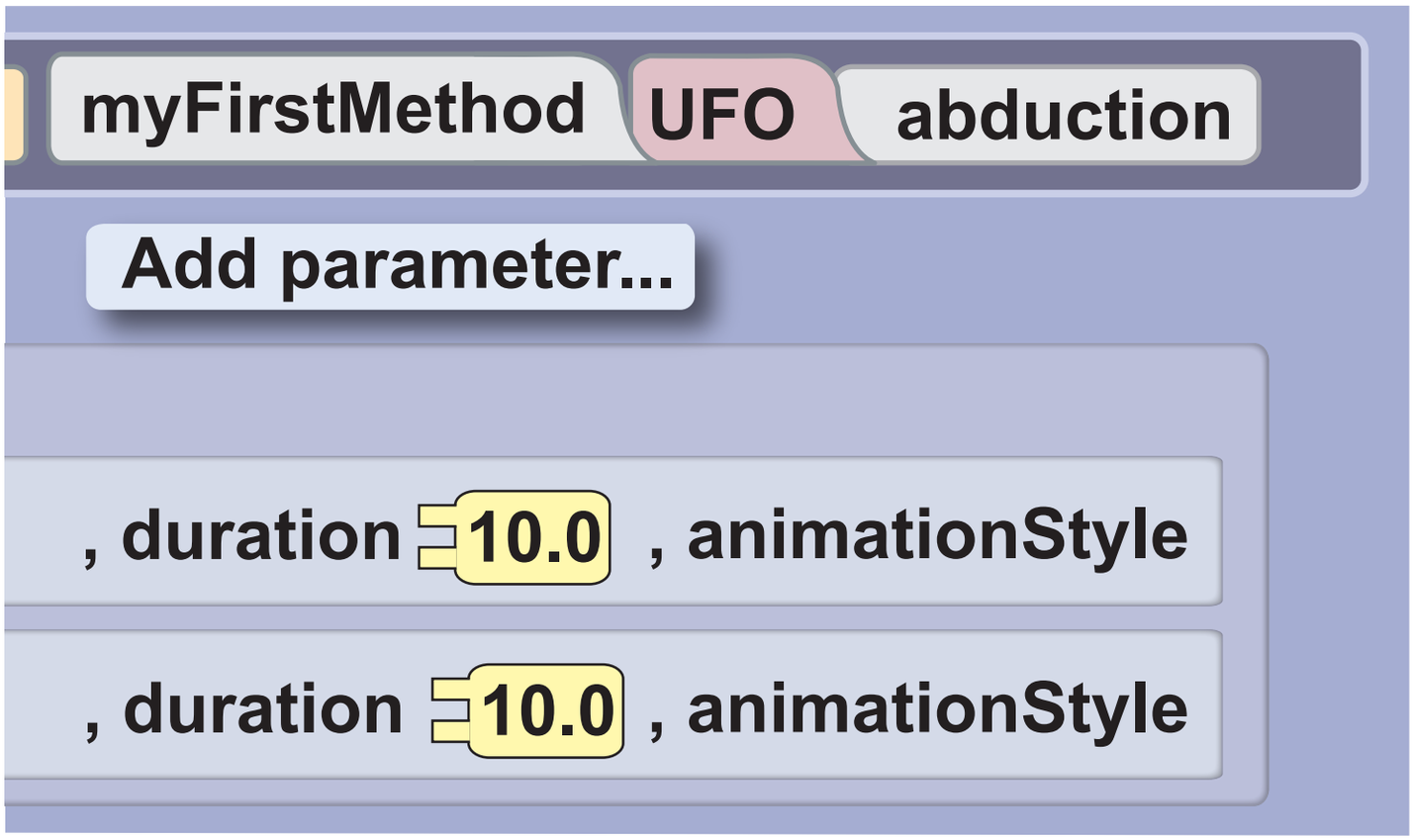


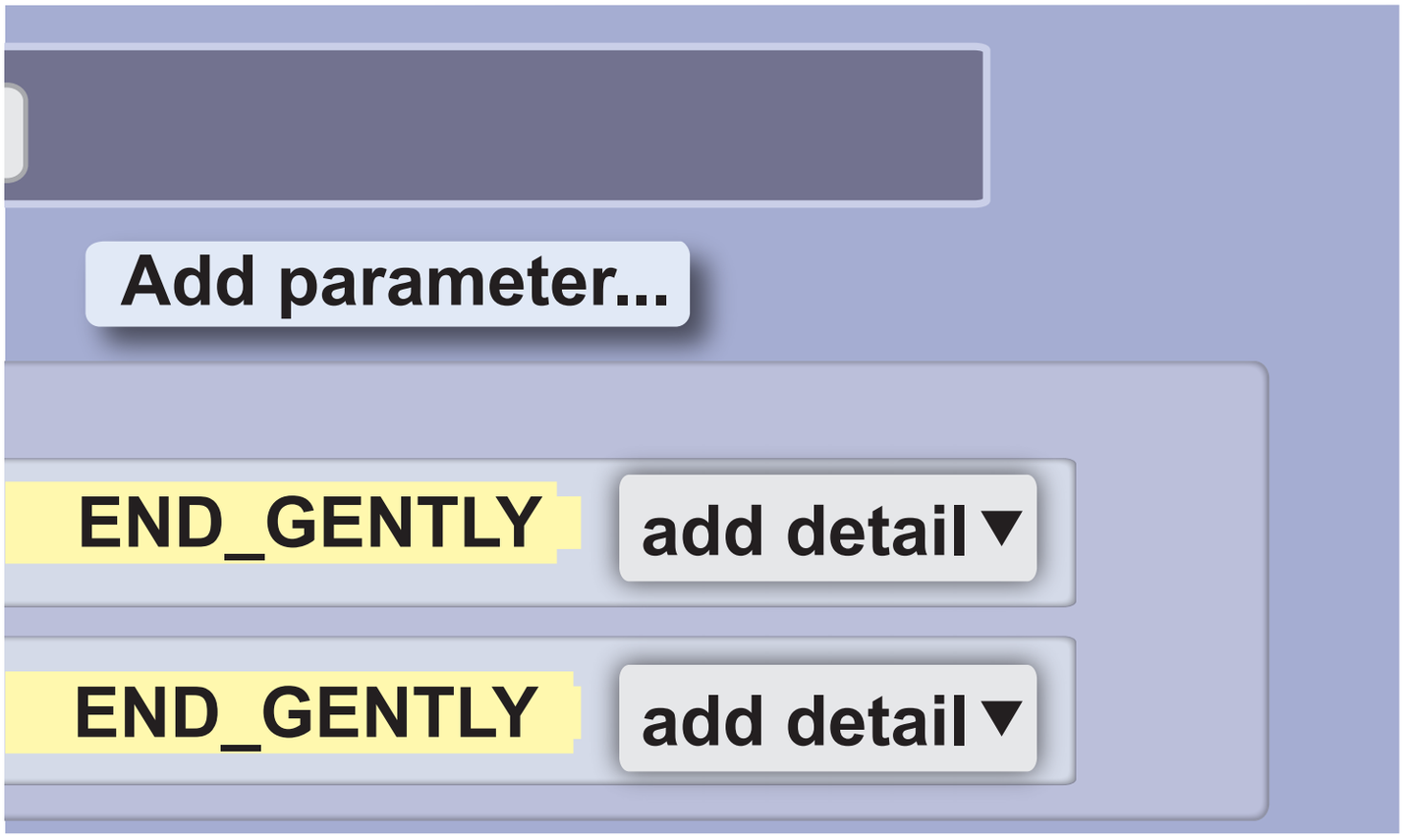
MyFirstMethod



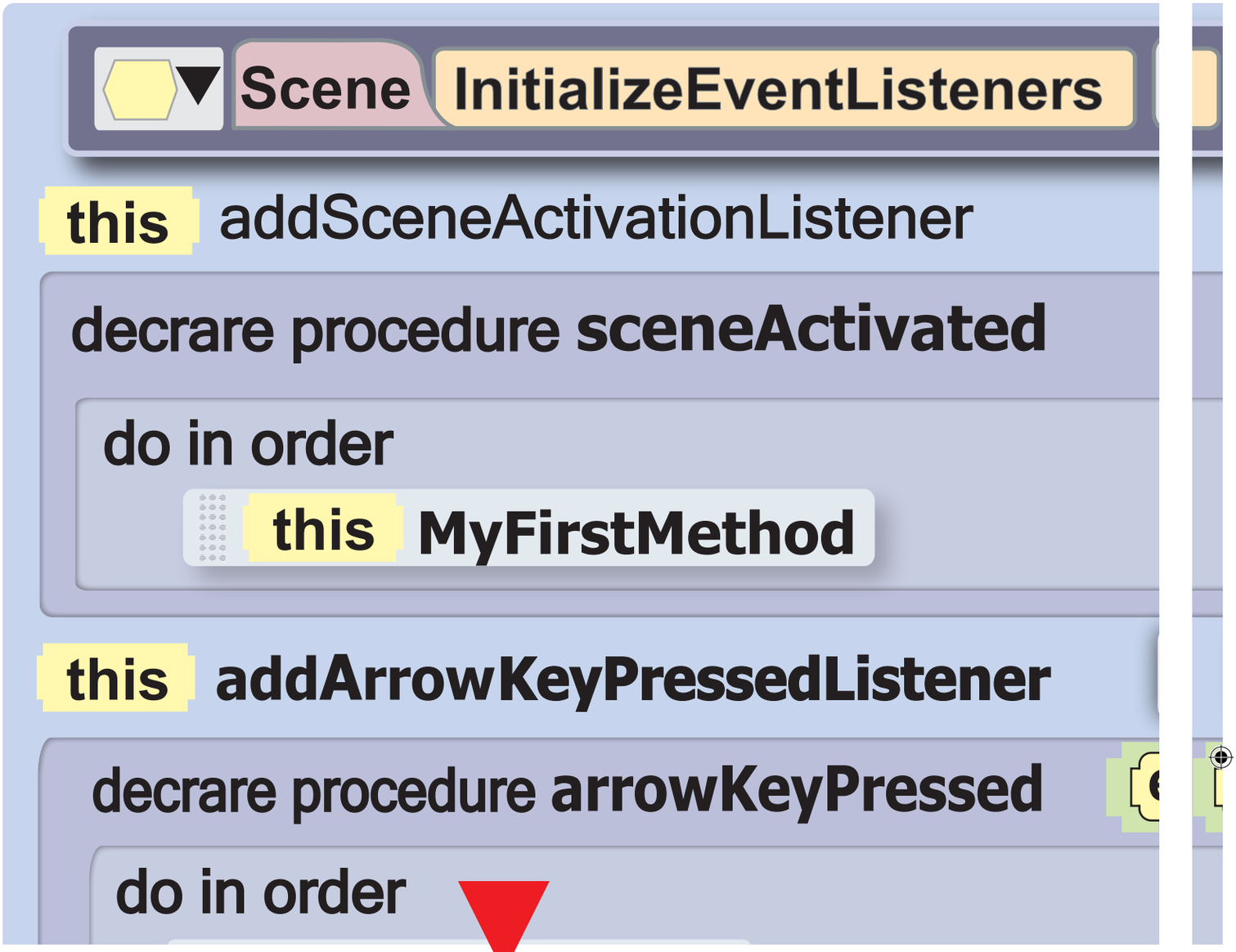


Εικόνα 7.18. Ο κώδικας της procedure-

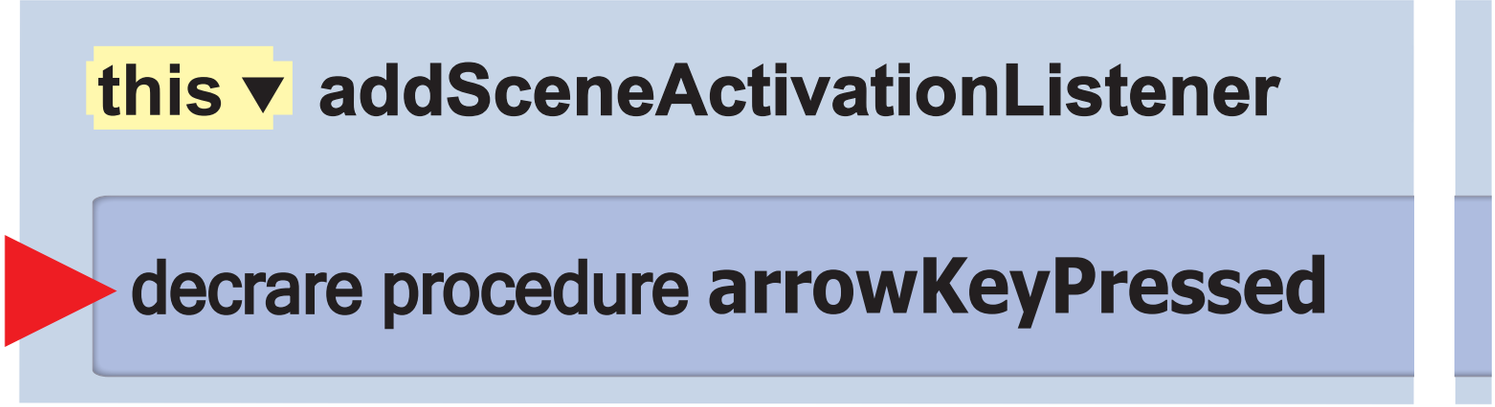


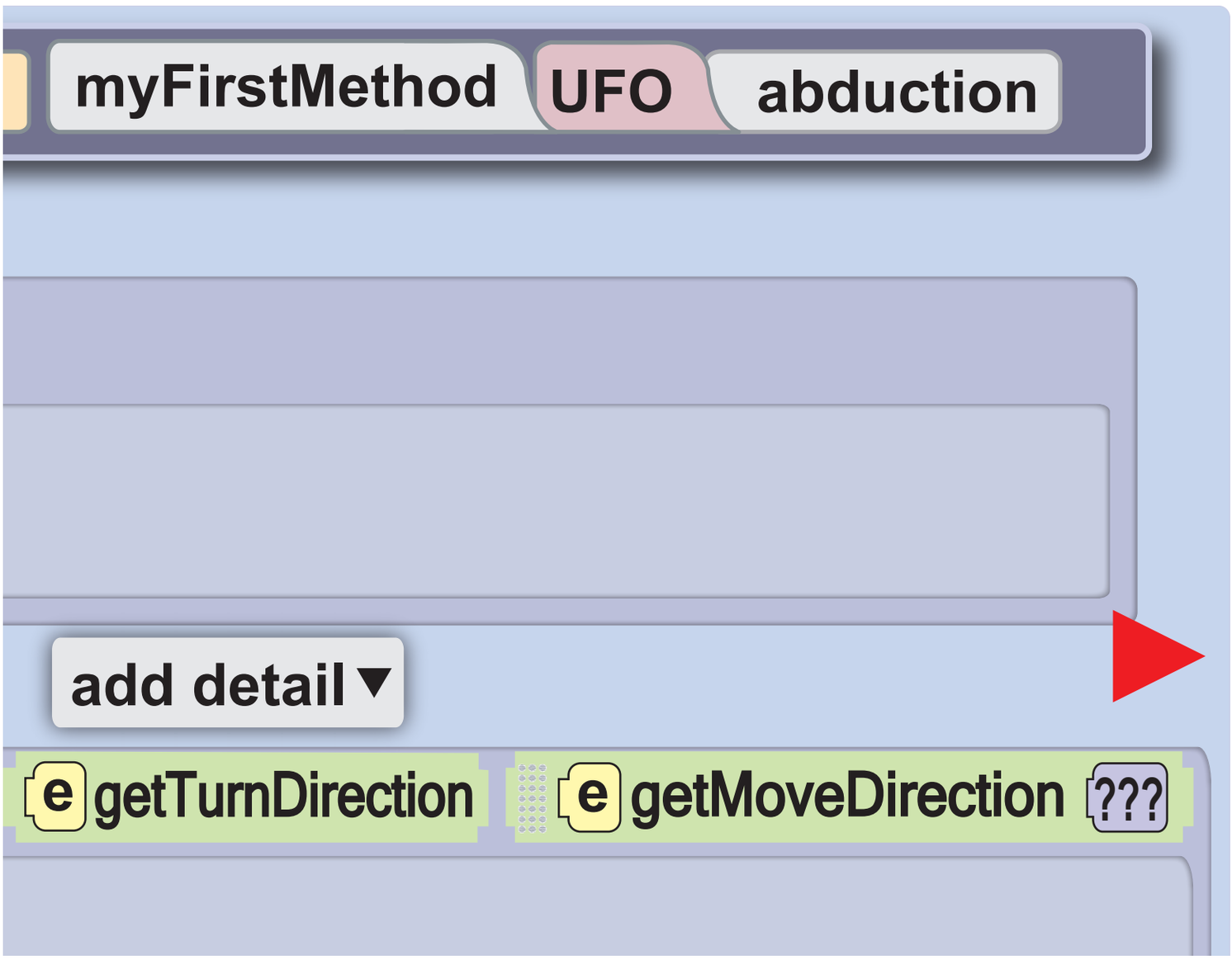


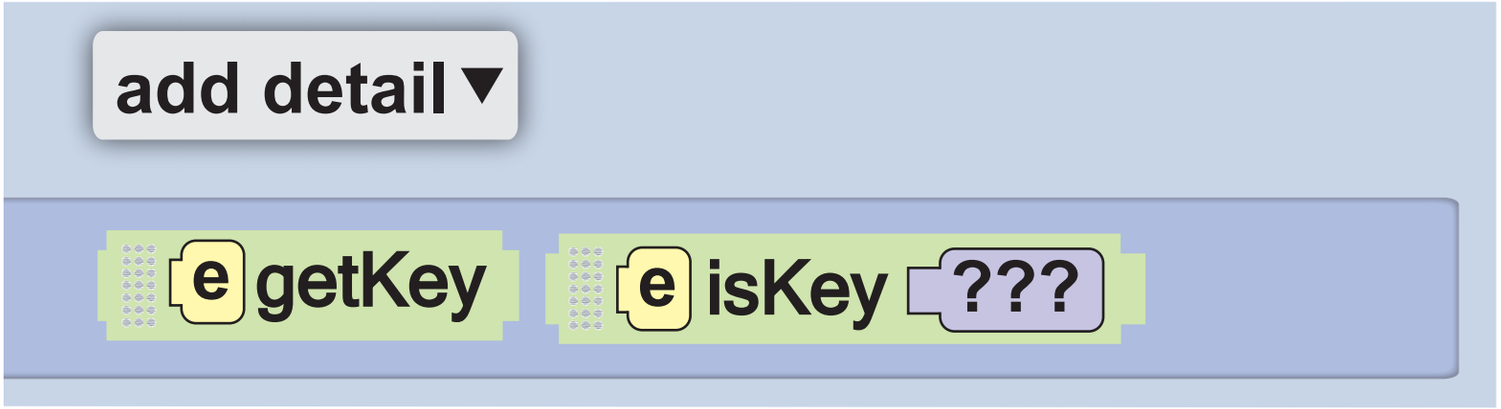
διαδικασίας που αφορά στην απαγωγή



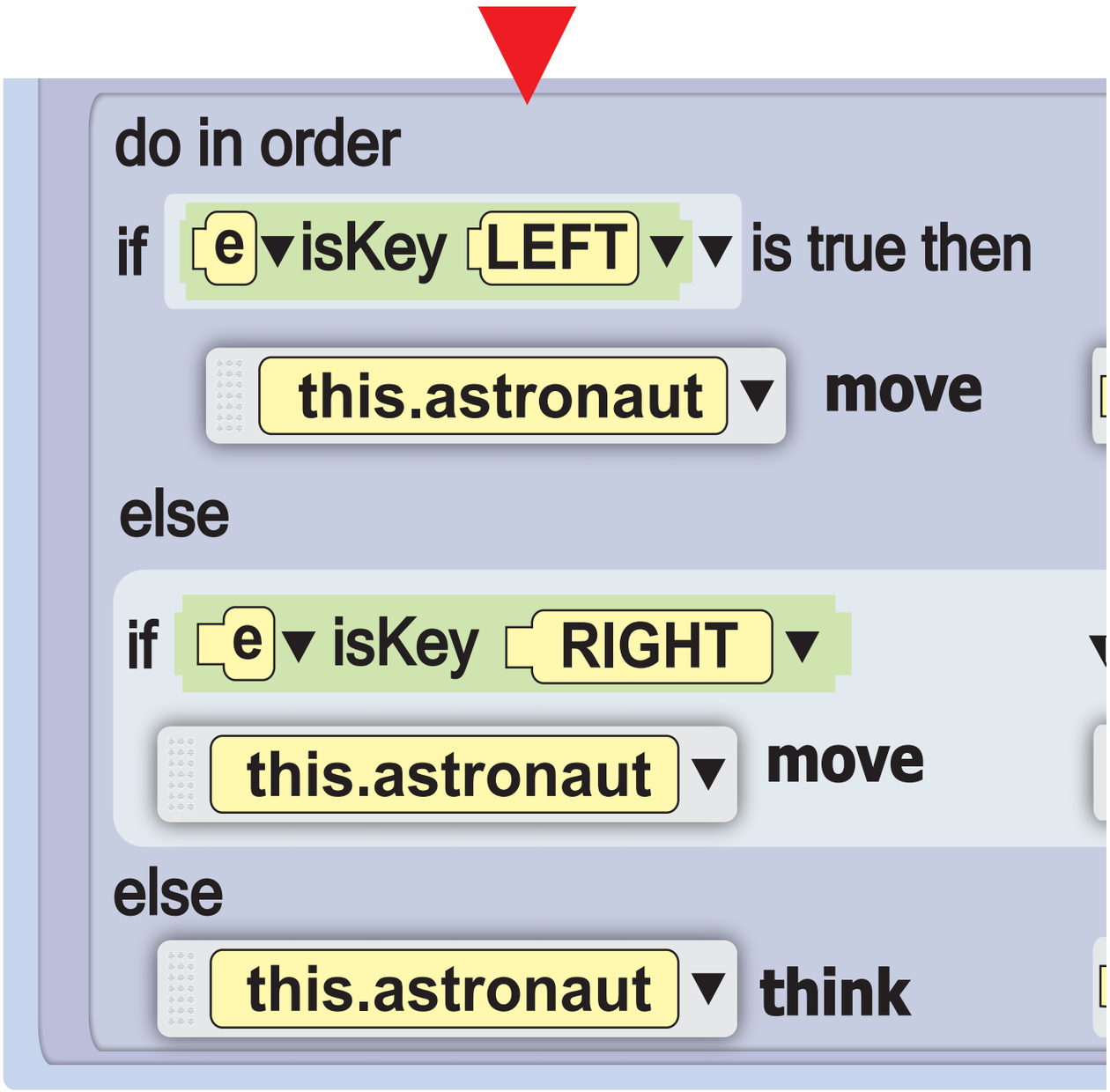
(Συνέχεια εικόνας)



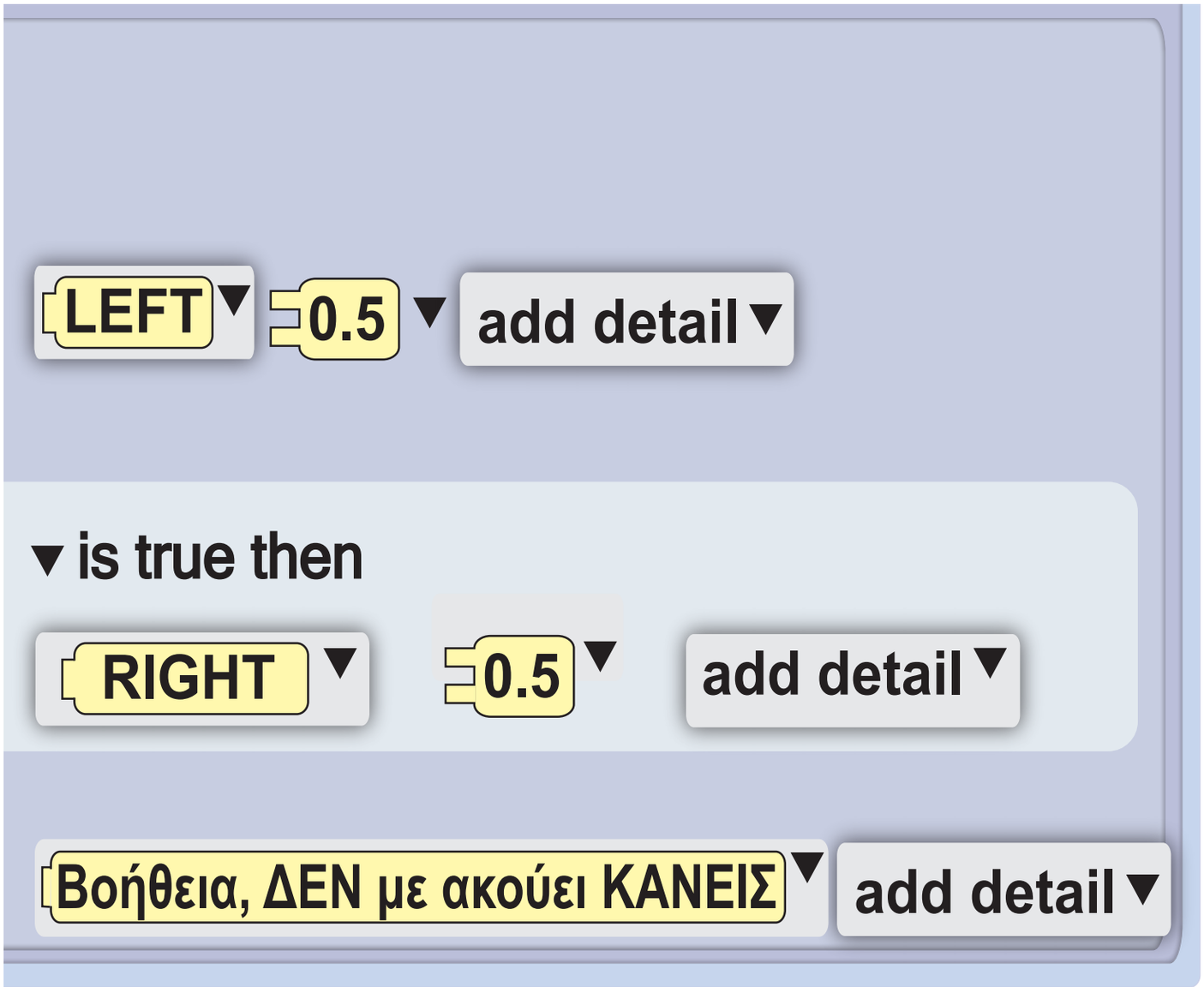




(Συνέχεια εικόνας)



Εικόνα 7.19. κώδικας στη μέθοδο του αστροναύτη από τον χρήστη



EventListeners για τον χειρισμό

Ερωτήσεις - Δραστηριότητες:

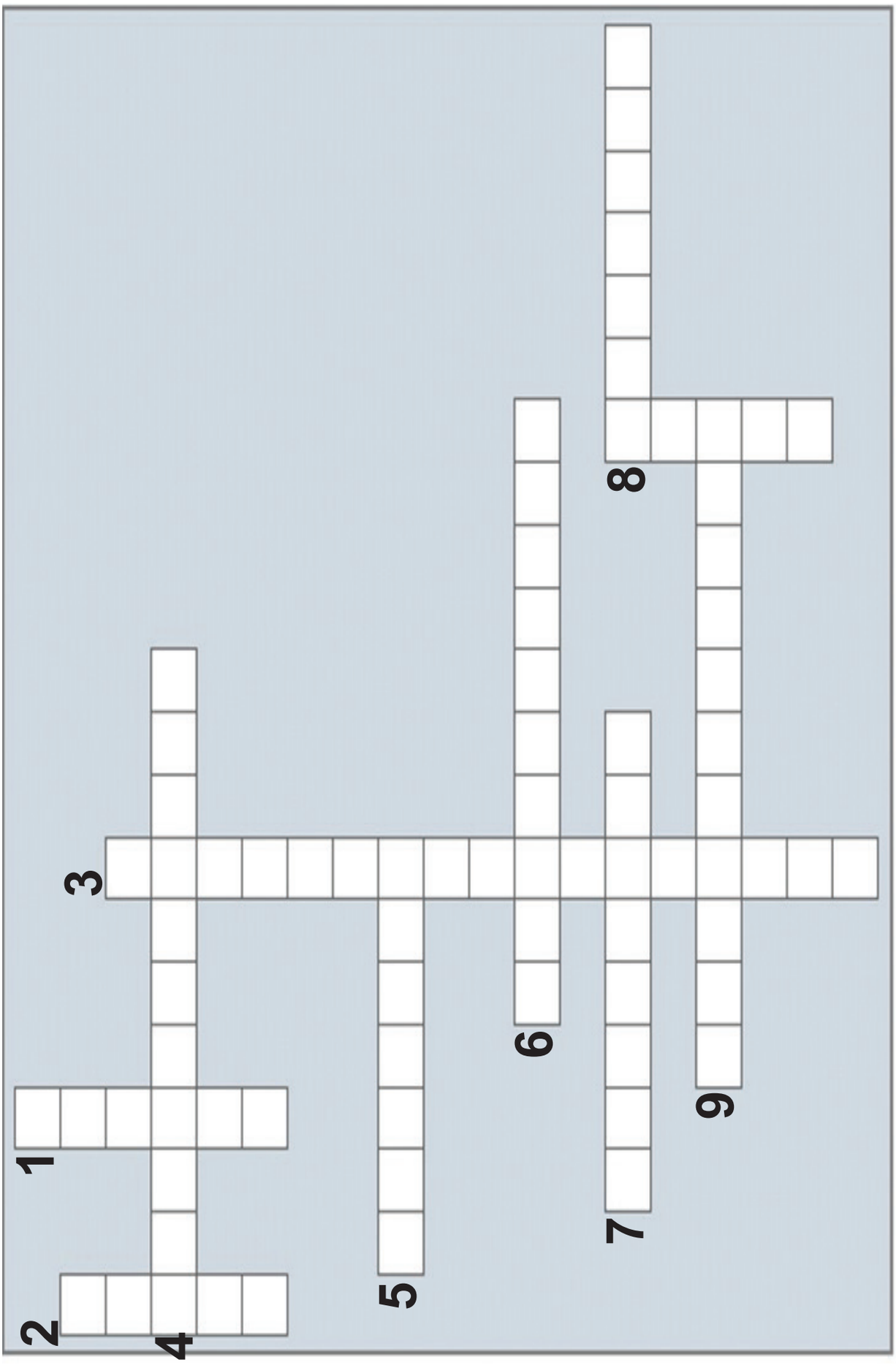
1. Εμπλουτίστε την εφαρμογή με καλύτερες κινήσεις των αντικειμένων, όπως π.χ. όταν μετακινείται ο αστροναύτης δεξιά ή αριστερά να στρίβει και αναλόγως.

2. Επεκτείνατε την εφαρμογή, εισάγοντας έναν ακόμη αστροναύτη, και εμπλουτίστε την αλληλεπίδραση μεταξύ όλων των αντικειμένων.

3. Αφού πρώτα περιηγηθείτε σε έτοιμα προγράμματα που έχουν δημιουργηθεί με το Alice, συζητήστε σε ομάδες μια δική σας ιστορία που νομίζετε ότι είναι εφικτό να υλοποιηθεί στο συγκεκριμένο περιβάλλον. Συμβουλευτείτε τον καθηγητή σας και υλοποιήστε την.

Ασκήσεις Αυτοαξιολόγησης

1. Λύστε το σταυρόλεξο (συμπληρώστε το με κεφαλαία γράμματα της ελληνικής ή αγγλικής ορολογίας, κατά περίπτωση).



Οριζόντια

4. Οπτικό περιβάλλον προγραμματισμού με πλακίδια (blocks) που χρησιμοποιούμε για την ανάπτυξη εφαρμογών για φορητές συσκευές (έξυπνα κινητά, tablets) με λειτουργικό σύστημα Android.

5. Έτσι ονομάζονται οι διαδικασίες που ορίζουν τις συμπεριφορές των αντικειμένων στον αντικειμενοστρεφή προγραμματισμό.

6. Κατά τη φάση της _____ του κύκλου ζωής μιας εφαρμογής γίνονται όλες οι απαραίτητες προσαρμογές, αναβαθμίσεις και διορθώσεις της εφαρμογής, προκειμένου αυτή να συνεχίσει να χρησιμοποιείται απρόσκοπτα και αποδοτικά.

7. Κατά τη φάση της _____ του κύκλου ζωής μιας εφαρμογής

καταγράφονται τα δεδομένα και τα ζητούμενα, οι προδιαγραφές και οι απαιτήσεις των μελλοντικών χρηστών της εφαρμογής.

8. Με το App Inventor αναπτύσσουμε εφαρμογές για φορητές συσκευές με Λειτουργικό σύστημα ____

9. Ορισμένοι προγραμματιστικοί παρέχουν τρισδιάστατη (3D) απεικόνιση, π.χ. Kodu, Yenka.

Κατακόρυφα

1. Χαρακτηρίζεται έτσι το περιβάλλον προγραμματισμού, όπου ο προγραμματιστής δεν πληκτρολογεί εντολές, αλλά επιλέγει και τοποθετεί κατάλληλα γραφικά στοιχεία (πλακίδια – blocks) με τη διαδικασία “σύρε και άφησε”.

2. Είναι το πρότυπο, καλούπι που χρησιμοποιούμε, για να

δημιουργήσουμε αντικείμενα στον αντικειμενοστρεφή προγραμματισμό.

3. Στο εκπαιδευτικό περιβάλλον Alice αναπτύσσουμε εφαρμογές με _____ προγραμματισμό.

8. 3D περιβάλλον προγραμματισμού για την ανάπτυξη εικονικών κόσμων με δυναμικές κινήσεις χαρακτήρων και αλληλεπίδραση με τον χρήστη.

2. Να χαρακτηρίσετε ως σωστή (Σ) ή λάθος (Λ) καθεμιά από τις παρακάτω προτάσεις:

Προτάσεις	Σ/Λ
1. Οι προγραμματιστικοί μικρόκοσμοι διαθέτουν ένα περιορισμένο ρεπερτόριο εντολών με απλή σύνταξη και απλές δομές δεδομένων, και διευκολύνουν τη δημιουργία παιχνιδιών με τη διαχείριση ενός κεντρικού ήρωα.	
2. Η γλώσσα Logo και τα Logo-like περιβάλλοντα ανήκουν στα επαγγελματικά προγραμματιστικά περιβάλλοντα.	

3. Τα συντακτικά λάθη στον προγραμματισμό εντοπίζονται μετά την εκτέλεση του προγράμματος, όταν παρατηρούμε ότι τα αποτελέσματα δεν είναι τα επιθυμητά.	
4. Στο προγραμματιστικό περιβάλλον App Inventor έχουμε τη δυνατότητα εκτέλεσης της εφαρμογής μας στον προσομοιωτή (Emulator) φορητής συσκευής.	
5. Το Alice είναι ένα ελεύθερα διαθέσιμο και δισδιάστατο (2D) περιβάλλον προγραμματισμού.	
6. Στο αντικειμενοστρεφές προγραμματιστικό περιβάλλον Alice έχουμε επίσης προγραμματισμό οδηγούμενο από γεγονότα.	

3. Αντιστοιχίστε τα δεδομένα μεταξύ των δύο στηλών. Η αντιστοίχιση είναι ένα προς πολλά (σε κάθε επιλογή της αριστερής στήλης αντιστοιχούν 2-3 από τη δεξιά στήλη).

(Α) Το περιβάλλον App Inventor υποστηρίζει (Β) Μεταφραστικό πρόγραμμα (Γ) Τρισδιάστατη απεικόνιση (Δ) Κάθε γλώσσα προγραμματισμού διαθέτει υποχρεωτικά (Ε) Ο κύκλος ζωής εφαρμογών περιλαμβάνει	1- Σχεδίαση εφαρμογής 2- Διερμηνευτής 3- Λεξιλόγιο 4- Υλοποίηση εφαρμογής 5- Δομή ελέγχου (if) 6- Μεταγλωττιστής 7- Kodu 8- Δομή επανάληψης (While) 9- Αλφάβητο 10- Alice 11- Συντακτικό 12- Μεταβλητές (Variables)

Θέματα για Συζήτηση

1. Να συγκρίνετε τη διαδικασία κατασκευής ενός μεγάλου τεχνικού έργου (π.χ. μιας γέφυρας, ενός κτιρίου) με τα βήματα του κύκλου ζωής εφαρμογών και να καταγράψετε τις ομοιότητες και τις διαφορές.
2. Υποθέστε ότι θέλετε να αναπτύξετε μια εφαρμογή παιχνιδιού. Να περιγράψετε τις ενέργειες που θα κάνατε σε κάθε βήμα του κύκλου ζωής εφαρμογών.
3. Αφού ανατρέξετε στο κεφάλαιο 6 και παρατηρήσετε τις εικόνες 6.1, 6.2 και 6.3, να περιγράψετε ποια γλώσσα προγραμματισμού θα επιλέγατε, για να γράψετε το πρώτο σας πρόγραμμα.
4. Να καταγράψετε τις δυνατότητες

και τα χαρακτηριστικά που θα θέλατε να έχει το προγραμματιστικό περιβάλλον που θα διαλέγατε, για να αναπτύξετε μια εφαρμογή.

5. Γιατί πιστεύετε ότι το Android είναι ένα δημοφιλές Λειτουργικό Σύστημα για φορητές συσκευές;
6. Θεωρείτε ότι είναι σημαντικό, όταν αναπτύσσετε μια εφαρμογή στο περιβάλλον App Inventor ή Alice, να ακολουθείτε τα βήματα του κύκλου ζωής εφαρμογών;

ΛΕΞΙΛΟΓΙΟ ΒΑΣΙΚΩΝ ΟΡΩΝ

Α

Αντικειμενοστρεφής προγραμματισμός: Μοντέλο προγραμματισμού στο οποίο τα δεδομένα και οι εντολές των προγραμμάτων οργανώνονται σε αντικείμενα.

Γ

Γλώσσα προγραμματισμού: Τεχνητή γλώσσα για τη δημιουργία προγραμμάτων.

Δ

Δίκτυο υπολογιστών: Είναι ένα σύνολο από υπολογιστές που είναι συνδεδεμένοι μεταξύ τους μέσω κάποιου μέσου μετάδοσης, ώστε να μπορούν να ανταλλάσσουν δεδομένα και να μοιράζονται διάφορες περιφερειακές συσκευές (π.χ. εκτυπωτές).

Ε

Εγγενείς εφαρμογές: Εφαρμογές που έχουν μεταφραστεί για μια συγκεκριμένη πλατφόρμα ή Λειτουργικό Σύστημα εγκαθίστανται και εκτελούνται σε αυτό και μπορούν να αλληλεπιδρούν εύκολα με λειτουργίες του συστήματος καθώς και με το υλικό του υπολογιστή στον οποίο είναι εγκατεστημένες.

Ελεύθερο Λογισμικό/Λογισμικό Ανοικτού Κώδικα (ΕΛ/ΛΑΚ): Το λογισμικό που μπορεί να χρησιμοποιηθεί, αντιγραφεί, μελετηθεί, τροποποιηθεί και αναδιανεμηθεί χωρίς περιορισμό.

Εμφώλευση ετικετών HTML: Η εισαγωγή μιας ετικέτας HTML μέσα σε μία άλλη, με τρόπο ώστε η ετικέτα που εισάγεται «εσωτερικά» (μέσα στην «εξωτερική») να εμφανίζεται

ολόκληρη και να ολοκληρώνεται πριν από την εξωτερική. Η εμφώλευση πρέπει να ακολουθεί συντακτικούς και σημασιολογικούς κανόνες που αφορούν στο ποιες ετικέτες μπορούν να εμφωλευτούν.

Ενσωμάτωση περιεχομένου: Η εισαγωγή περιεχομένου σε ένα έγγραφο HTML από διαφορετική πηγή με τρόπο ώστε το περιεχόμενο αυτό να περιλαμβάνεται εντός του εγγράφου και να περιέχεται σε αυτό σαν να αποτελεί κομμάτι του.

Εφαρμογές Διαδικτύου: Εφαρμογές που παρέχονται μέσω Διαδικτύου και εκτελούνται από το πρόγραμμα πλοήγησης (φυλλομετρητή) χωρίς να απαιτείται εγκατάσταση εξειδικευμένου λογισμικού.

Εφαρμογή νέφους: Εφαρμογή που εκτελείται στο «υπολογιστικό

νέφος» και παρέχεται στον χρήστη μέσω διαδικτύου.

Η

Ηλεκτρονική Μάθηση: Μάθηση που επιτυγχάνεται με χρήση των Τεχνολογιών Πληροφορίας και Επικοινωνιών (ΤΠΕ).

Ηλεκτρονικό ψάρεμα: Ένας τρόπος εξαπάτησης των χρηστών υπολογιστών με στόχο να τους κάνει να αποκαλύψουν προσωπικές πληροφορίες ή οικονομικά στοιχεία, μέσω ενός παραπλανητικού μηνύματος ηλεκτρονικού ταχυδρομείου ή ενός παραπλανητικού δικτυακού τόπου.

Κ

Κακόβουλο λογισμικό: Το λογισμικό το οποίο εκ προθέσεως διαθέτει τις απαιτούμενες εντολές για να βλάψει ένα υπολογιστικό σύστημα.

Κβαντικός Υπολογιστής: Μια υπολογιστική συσκευή που εκμεταλλεύεται χαρακτηριστικές ιδιότητες της κβαντομηχανικής για την επεξεργασία δεδομένων και την εκτέλεση υπολογισμών.

Κεντρική Μονάδα Επεξεργασίας ή ΚΜΕ (Central Processing Unit – CPU): Το μέρος του υλικού (hardware) που εκτελεί τις εντολές ενός προγράμματος υπολογιστή χρησιμοποιώντας βασικές αριθμητικές και λογικές πράξεις, καθώς και λειτουργίες εισόδου-εξόδου.

Κλάση: Πρότυπο που χρησιμοποιείται για τη δημιουργία ενός αντικειμένου στον αντικειμενοστρεφή προγραμματισμό.

Κληρονομικότητα: Η διεργασία μέσω της οποίας μια κλάση μπορεί να αποκτήσει τις ιδιότητες και μεθόδους μιας άλλης κλάσης στον

αντικειμενοστρεφή προγραμματισμό.

Κοινωνικό Δίκτυο: Ηλεκτρονική πλατφόρμα εικονικής κοινότητας που παρέχει στα μέλη της δυνατότητες διασύνδεσης και αλληλεπίδρασης.

Κύκλος ζωής εφαρμογών: Μια συστηματική διαδικασία με βήματα-φάσεις για την ανάπτυξη εφαρμογών.

Λ

Λειτουργικό Σύστημα ή ΛΣ (Operating System ή OS): Το λογισμικό του υπολογιστή που είναι υπεύθυνο για τη διαχείριση και τον συντονισμό των εργασιών, καθώς και την κατανομή των διαθέσιμων πόρων.

Λογισμικό (software): Το σύνολο των προγραμμάτων του υπολογιστή.

Λογισμικό ασφαλείας: Λογισμικό για την προστασία του υπολογιστή από κακόβουλα προγράμματα

Μ

Μαζικά Ανοικτά Διαδικτυακά Μαθήματα (Massive Open Online Courses – MOOCs): Δωρεάν ηλεκτρονικά μαθήματα σε μεγάλες ομάδες μαθητών- χρηστών διαδικτύου, τα οποία προσφέρονται από τη συνεργασία κορυφαίων πανεπιστημίων μεταξύ τους και με παρόχους υπηρεσιών διαδικτύου.

Μέθοδος: Διαδικασία που ορίζει συμπεριφορά για μια κλάση στον αντικειμενοστρεφή προγραμματισμό.

Μηνύματα spam/Ενοχλητική Αλληλογραφία: Ανεπιθύμητα μηνύματα ηλεκτρονικού ταχυδρομείου που αποστέλλονται μαζικά, κυρίως για προώθηση προϊόντων.

Ο

Ολοκληρωμένο περιβάλλον ανάπτυξης εφαρμογών: Εφαρμογή λογισμικού που περιλαμβάνει όλα

τα απαραίτητα εργαλεία που χρειάζονται οι προγραμματιστές για την ανάπτυξη εφαρμογών.

Οπτικός προγραμματισμός: **Ανάπτυξη προγραμμάτων με άμεσο χειρισμό αντικειμένων με γραφική αναπαράσταση.**

Π

Πειρατεία λογισμικού: **Η παράνομη αντιγραφή και χρήση προγραμμάτων χωρίς την άδεια του δημιουργού τους.**

Πνευματικά δικαιώματα: **Τα δικαιώματα που αποκτά κάποιος πάνω σε ένα πρωτότυπο πνευματικό δημιούργημα (π.χ. μουσική, συγγραφικό έργο, λογισμικό κ.ά.).**

Πρόβλημα: **Κάθε ζήτημα που τίθεται προς επίλυση, κάθε κατάσταση που μας απασχολεί και πρέπει να αντιμετωπιστεί.**

Πρόγραμμα: Μια σειρά από εντολές (οδηγίες) που κατευθύνουν με κάθε λεπτομέρεια τον υπολογιστή, για να εκτελέσει μία συγκεκριμένη εργασία και να επιλύσει ένα πρόβλημα.

Προσωπικά δεδομένα: Οι πληροφορίες που μας χαρακτηρίζουν όπως, για παράδειγμα, το όνομά μας, η διεύθυνσή μας, οι φωτογραφίες μας, οι απόψεις μας κ.ά.

Πρωτόκολλο επικοινωνίας: Το σύνολο των κανόνων και διαδικασιών που οφείλουν να εφαρμόζουν οι υπολογιστές και τα περιφερειακά, για να είναι δυνατή η επικοινωνία μεταξύ τους.

Σ

Σύστημα Διαχείρισης Μάθησης (Learning Management System – LMS): Πλατφόρμα λογισμικού για την υποστήριξη της ηλεκτρονικής μάθησης.

Τ

Τοπικό Δίκτυο υπολογιστών: Το δίκτυο υπολογιστών στο οποίο οι υπολογιστές και τα περιφερειακά που το απαρτίζουν εκτείνονται σε μικρή απόσταση.

Υ

Υλικό υπολογιστή (hardware): Τα φυσικά μέρη που μπορούμε να δούμε και να αγγίξουμε σε έναν υπολογιστή.

Υπολογιστικό νέφος: Το σύνολο των υποδομών δικτύου, εξυπηρετητών και λογισμικού, που παρέχονται μέσω δικτύου και στις οποίες οι χρήστες εκτελούν εργασίες αντί του προσωπικού τους υπολογιστή.

ΠΕΡΙΕΧΟΜΕΝΑ

ΕΝΟΤΗΤΑ 2: ΠΡΟΓΡΑΜΜΑΤΙΣΤΙΚΑ ΠΕΡΙΒΑΛΛΟΝΤΑ–ΔΗΜΙΟΥΡΓΙΑ ΕΦΑΡΜΟΓΩΝ

Βάσει του ν. 3966/2011 τα διδακτικά βιβλία του Δημοτικού, του Γυμνασίου, του Λυκείου, των ΕΠΑ.Λ. και των ΕΠΑ.Σ. τυπώνονται από το ΙΤΥΕ - ΔΙΟΦΑΝΤΟΣ και διανέμονται δωρεάν στα Δημόσια Σχολεία. Τα βιβλία μπορεί να διατίθενται προς πώληση, όταν φέρουν στη δεξιά κάτω γωνία του εμπροσθόφυλλου ένδειξη «ΔΙΑΤΙΘΕΤΑΙ ΜΕ ΤΙΜΗ ΠΩΛΗΣΗΣ». Κάθε αντίτυπο που διατίθεται προς πώληση και δεν φέρει την παραπάνω ένδειξη θεωρείται κλεψίτυπο και ο παραβάτης διώκεται σύμφωνα με τις διατάξεις του άρθρου 7 του νόμου 1129 της 15/21 Μαρτίου 1946 (ΦΕΚ 1946,108, Α').