ΠΡΟΓΡΑΜΜΑ ΜΕΤΑΠΤΥΧΙΑΚΩΝ ΣΠΟΥΔΩΝ
ΤΜΗΜΑΤΟΣ ΕΦΑΡΜΟΣΜΕΝΗΣ ΠΛΗΡΟΦΟΡΙΚΗΣ

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

ΘΕΜΑ:

«ΜΕΛΕΤΗ ΚΑΙ ΑΞΙΟΛΟΓΗΣΗ ΤΟΥ ΠΡΟΓΡΑΜΜΑΤΙΣΤΙΚΟΥ ΠΕΡΙΒΑΛΛΟΝΤΟΣ “ALICE” ΚΑΙ ΑΝΑΠΤΥΞΗ ΔΙΔΑΚΤΙΚΟΥ ΥΛΙΚΟΥ»

Ονοματεπώνυμο Φοιτητή: Κοκοκύρης Μιχαήλ
Εξεταστική Επιτροπή
Επιβλέπουσα: Σατρατζέμη Μαρία, Καθηγήτρια
Μέλος εξεταστικής: Ευαγγελίδης Γεώργιος, Αναπληρωτής Καθηγητής

Θεσσαλονίκη, Μάρτιος 2008

Περιεχόμενα

Περιεχόμενα……..2

Πρόλογος…….....3

Κεφάλαιο 1……..5

Εισαγωγή……..6

Ερευνητικές εργασίες εφαρμογής του περιβάλλοντος Alice..……11

Κεφάλαιο 2……...….21

Εκμάθηση προγραμματισμού με το Alice……..22

Παραγωγή εκπαιδευτικού υλικού στα ελληνικά……..……23

Κεφάλαιο 3……..…125

Υλοποίηση προγραμμάτων……..…126

Κεφάλαιο 4……..…133

Συμπεράσματα……...….134

Βιβλιογραφία…….135

Πρόλογος

Η εργασία αυτή αποτελεί μια έρευνα, μελέτη και αξιολόγηση του προγραμματιστικού περιβάλλοντος “Alice”. Το πρόγραμμα “Alice” έχει δημιουργηθεί από μια ομάδα ερευνητών του πανεπιστήμιου Carnegie Mellon το 1999 μεταξύ των οποίων είναι και οι Wanda Dann, Stephen Cooper, Randy Pausch. Η μελέτη του εκπαιδευτικού, προγραμματιστικού εργαλείου “Alice” γίνεται με διαφορετικές προσεγγίσεις σε κάθε κεφάλαιο.

Στο πρώτο κεφάλαιο παρατίθεται μια εισαγωγή στην οποία αναφέρονται ορισμένα ιστορικά στοιχεία για το πρόγραμμα Alice. Γίνεται αναφορά στην δυσκολία του αντικειμενοστραφούς προγραμματισμού, στην ανάγκη δημιουργίας που υπήρχε ώστε οι φοιτητές του Carnegie Mellon να μπορούν να μαθαίνουν εύκολα και ευχάριστα αντικειμενοστραφή προγραμματισμό. Ακόμη το κεφάλαιο 1 αναφέρεται και σε μια έρευνα στην οποία αποδεικνύεται ότι παλιότερα επέλεγαν λίγα κορίτσια μαθήματα προγραμματισμού, με την χρήση του προγράμματος Alice όμως συμμετέχουν πολύ περισσότερα.

Στο δεύτερο κεφάλαιο παράγεται το εκπαιδευτικό υλικό «Learning to program with Alice» των Wanda Dann, Stephen Cooper, Randy Pausch στα ελληνικά. Με αυτόν τον τρόπο παρουσιάζεται ακριβώς το περιβάλλον Alice, τα χαρακτηριστικά του και ο τρόπος χρήσης του. Έτσι διαφαίνεται πόσο εύκολο είναι στη χρήση του και πόσο φιλικό στον χρήστη χωρίς αυτό να σημαίνει ότι υστερεί σε δυνατότητες.

Στο τρίτο κεφάλαιο παρουσιάζονται ορισμένα παραδείγματα-προγράμματα που έχουν υλοποιηθεί με το Alice. Τα παραδείγματα είναι από απλές κινήσεις χαρακτήρων σε ένα φτωχό από εικόνες κόσμο, μέχρι και περιβάλλοντα που αγγίζουν την αίσθηση της πραγματικότητας και ο κάθε χαρακτήρας αλληλεπιδρά με τον χρήστη. Σε αυτά τα τελευταία παραδείγματα ο χρήστης έχει τη δυνατότητα με το πάτημα ενός κουμπιού ή με το κλικ του ποντικιού να προκαλέσει την κίνηση ενός χαρακτήρα.

Στο τέταρτο κεφάλαιο γίνεται μια σύνοψη της παρούσας διπλωματικής εργασίας και παρατίθενται ορισμένα συμπεράσματα που προέκυψαν από την μελέτη του προγραμματιστικού εκπαιδευτικού υλικού Alice.

ΚΕΦΑΛΑΙΟ 1
ΕΙΣΑΓΩΓΗ

Εισαγωγή

Ο αντικειμενοστραφής προγραμματισμός έγινε πολύ γνωστός στην επιστήμη των υπολογιστών. Ένα αντικειμενοστραφές πρόγραμμα θεωρείται ως μια συλλογή από αντικείμενα με συμπεριφορά, χαρακτηριστικά και ιδιότητες παρά μια λίστα εντολών. Έτσι η επαναχρησιμοποίηση ορισμένων τμημάτων κώδικα και οι φόρμες των αντικειμένων έχουν κάνει ευκολότερη την συντήρηση και την αναβάθμιση των προγραμμάτων.

Παρόλα αυτά παρουσιάστηκε ένα πρόβλημα. Οι μαθητές της μέσης εκπαίδευσης, ακόμη και οι πανεπιστημιακοί φοιτητές δεν επέλεγαν εύκολα ένα μάθημα προγραμματισμού. Οι περισσότεροι φοιτητές δεν προσελκύονταν από ένα μάθημα προγραμματισμού και αυτοί που απλά παρακολουθούσαν, για δοκιμή, ένα τέτοιο μάθημα δεν συγκινούνταν ώστε να το επιλέξουν. Οι κυριότεροι λόγοι ήταν απογοήτευση των μαθητών λόγο του επιπέδου δυσκολίας και η αμφιβολία για το αν θα καταφέρουν να αντεπεξέλθουν στις απαιτήσεις ενός τέτοιου μαθήματος. Το φαινόμενο αυτό παρατηρούταν περισσότερο σε φοιτήτριες παρά σε άνδρες φοιτητές.

Με αφορμή το παραπάνω γεγονός έγινε μια έρευνα από το Πανεπιστήμιο Carnegie Mellon και τελικά το 1999 δημιουργήθηκε ένα εκπαιδευτικό προγραμματιστικό περιβάλλον με το όνομα “Alice”.

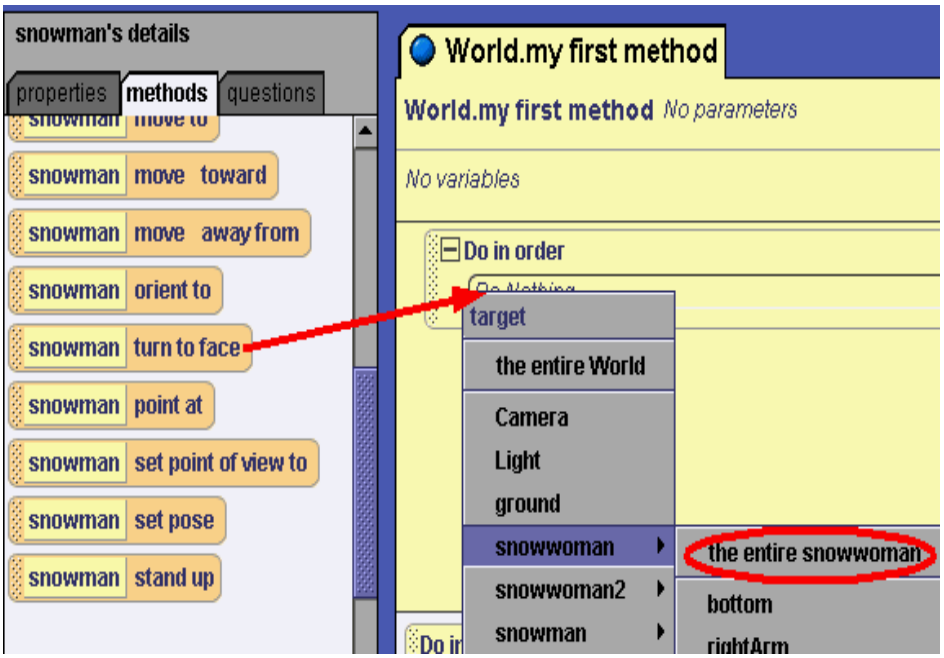
Η αρχική ιδέα του προγράμματος Alice εμφανίστηκε στις αρχές του 1991, όταν τα πρώτα συστήματα εικονικών κόσμων άρχισαν να δημιουργούνται στο Πανεπιστήμιο της Virginia. Ξεκινώντας με την επιθυμία δημιουργίας νέων τεχνικών αλληλεπίδρασης για την προσομοίωση προγραμμάτων, ανακαλύφθηκε ότι τα εργαλεία που ήταν διαθέσιμα για αυτό το σκοπό ήταν δύσκολα στη χρήση. Έτσι ξεκίνησε η ιδέα για δημιουργία νέων προγραμματιστικών εργαλείων που ήταν γρηγορότερα και ευκολότερα στη χρήση για οποιονδήποτε είχε την επιθυμία να μάθει να προγραμματίζει, ακόμη και για τους νέους μαθητές. Αυτή η προσπάθεια ξεκίνησε με πολλά διστακτικά προσχέδια με διάφορα ονόματα και τελικά κατέληξε στο σημερινό πρόγραμμα Alice.

Το Alice όπως αναφέρθηκε παραπάνω δημιουργήθηκε ουσιαστικά το 1999 από το Πανεπιστήμιο Carnegie Mellon με κύριους ερευνητές τους Wanda Dann, Stephen Cooper και Randy Pausch. Το Alice είναι ένα προγραμματιστικό περιβάλλον που χρησιμοποιείται ευρέως για εκπαιδευτικό σκοπό. Είναι ένα 3D περιβάλλον για δημιουργία εικονικών κόσμων που εμπεριέχει δυναμικές κινήσεις των χαρακτήρων και αλληλεπίδραση τους με το χρήστη. Αυτό είναι και το κύριο πλεονέκτημα του Alice. Χρησιμοποιεί 3D γραφικά ώστε να προσελκύσει τους μαθητές. Είναι πολύ πιο εύκολο και ευχάριστο για παράδειγμα να χρησιμοποιήσει κανείς μια κλάση Ανθρώπου – να της δώσει χαρακτηριστικά όπως φύλλο, χρώμα δέρματος, ύψος, χρώμα μαλλιών και ματιών και άλλα- όταν βλέπει ένα χαρακτήρα σαν αντικείμενο στον κόσμο τους προγράμματος που δημιουργεί, παρά να γράψει αμέτρητες γραμμές κώδικα χωρίς να έχει άμεσο αποτέλεσμα στην οθόνη του υπολογιστή.

Για τον παραπάνω λόγο το πρόγραμμα Alice έγινε ιδιαίτερα γνωστό και αποδεκτό σε ευρύ προγραμματιστικό κοινό. Κυρίως όμως είχε απήχηση σε

φοιτητές και παιδιά σχολείου που πλέον έβρισκαν ευχάριστο τον προγραμματισμό με αυτό το εργαλείο. Ακόμη και οι γυναίκες σπουδαστές ξεκίνησαν να επιλέγουν μαθήματα προγραμματισμού που γίνονταν με την βοήθεια του προγράμματος Alice γιατί αισθάνονται εξοικειωμένες με χαρακτήρες που υπάρχουν στην βιβλιοθήκη της Alice. Μερικοί ισχυρίζονται ότι το όνομα του προγράμματος Alice έχει δοθεί επίτηδες σε γένος θηλυκού έτσι ώστε να δημιουργηθεί ένα κλίμα «οικειότητας» και να προσεγγισθούν περισσότερες γυναίκες. Στην πραγματικότητα το σύστημα ονομάστηκε Alice προς τιμήν του Charles Lutwidge Dodson ο οποίος είχε ψευδώνυμο Lewis Caroll. Ο Lewis Caroll ήταν ο μαθηματικός που έγραψε το "Alice's Adventures in Wonderland" και το "Through the Looking Glass". Ο Caroll γνώριζε ότι το σημαντικότερο θέμα είναι να γίνουν τα πράγματα απλά και ενδιαφέροντα προς τον μαθητή.

Αναλυτικότερα στο προγραμματιστικό περιβάλλον Alice χρησιμοποιείται ένας συντάκτης μέσα στον οποίο ο χρήστης απλά επιλέγει, σύρει και αφήνει τα αντικείμενα και τις μεθόδους τους. Με αυτό τον τρόπο αποφεύγονται τα συντακτικά λάθη. Μπορεί να συμβούν λογικά λάθη αλλά εφόσον ο χρήστης μπορεί να εκτελέσει το πρόγραμμα, θα εντοπίσει οπτικά το λάθος και θα το διορθώσει εύκολα και κατόπιν δοκιμών. Έτσι προσαρμόζεται εύκολα ο κώδικας και εντοπίζονται τα αποτελέσματα των αλλαγών. Στην εικόνα παρακάτω φαίνεται ο συντάκτης σκηνών.



Στο Alice οι κλάσεις και τα αντικείμενα είναι κάτι που μπορεί κανείς να δει και να επιλέξει πριν τα προσθέσει στο πρόγραμμα του:

Η ευκολία χρήσης του προγράμματος, κατέστησε το Alice ένα κοινώς αποδεκτό πρόγραμμα το οποίο πλέον χρησιμοποιείται σε πολλά πανεπιστήμια και σχολεία. Ενδεικτικά μερικά από αυτά είναι:

- Bucknell University
- California Lutheran University
- California State University at Humboldt
- Camden County College
- Carnegie Mellon University
- Clemson University
- Colorado School of Mines
- Community College of Philadelphia
- Cornell University
- Duke University
- Georgetown College
- Haverford College
- Ithaca College
- Manor College
- Mississippi Valley State University
- Plymouth State University
- Saint Edward's University
- Saint Joseph's University
- Saint Lawrence College
- San Diego State University
- Sierra Nevada College
- Southwestern University
- Tompkins Cortland Community College
- University of Colorado
- University of Illinois
- University of Mississippi
- Virginia Tech
- Και πολλά άλλα Γυμνάσια

Η χρήση του προγράμματος Alice δεν παρέμεινε μόνο στα σχολεία, προχώρησε και σε άλλες εφαρμογές. Ένα τέτοιο παράδειγμα είναι η συνεργασία του Carnegie Mellon με την EA (Electronic Arts), εταιρία δημιουργίας και παραγωγής videogames. Έτσι το Alice συνεργάζεται με χαρακτήρες του παιχνιδιού Sims 2 της EA. Το παιχνίδι Sims 2 είναι ένα videogame που είναι γνωστό παγκοσμίως. Στις 10 Μαρτίου, λοιπόν, του 2006 στο Pittsburg το Πανεπιστήμιο Carnegie Mellon συμφώνησε μια συνεργασία με την EA, που είχε σκοπό την αναζωογόνηση της επιστήμης των υπολογιστών στην Αμερική σε όλες τις βαθμίδες εκπαίδευσης. Η EA συμφώνησε να βοηθήσει στην ανάπτυξη του Alice 3.0 παρέχοντας χαρακτήρες του παιχνιδιού The Sims –του παιχνιδιού για ηλεκτρονικούς υπολογιστές που έχει πουλήσει περισσότερο από κάθε άλλο παιχνίδι- της εταιρίας. Το περιεχόμενο του Sims θα μετατρέψει το Alice από ένα άτεχνο και άκομψο, 3D προγραμματιστικό εργαλείο σε ένα όμορφο και φιλικό προς το χρήστη προγραμματιστικό περιβάλλον. Η δημιουργία του Alice 3.0 ξεκίνησε κατευθείαν και θα διαρκέσει περίπου δύο χρόνια. Οι ειδικοί λένε πως όταν ολοκληρωθεί η μετατροπή, το νέο προγραμματιστικό περιβάλλον θα είναι σε θέση να γίνει το κύριο εργαλείο διδασκαλίας προγραμματισμού στην Αμερική. Με αυτόν τον τρόπο οι μαθητές που θα χρησιμοποιούν το Alice 3.0 θα δουλεύουν ουσιαστικά σε ένα περιβάλλον όμοιο με αυτό του Sims, εφόσον οι χαρακτήρες θα μοιάζουν και θα κινούνται όπως αυτοί του Sims και οι βιβλιοθήκη του Sims θα ενσωματωθεί στο πρόγραμμα. Με αυτό τον τρόπο ο προγραμματισμός θα γίνει ακόμη πιο διασκεδαστικός. Παρακάτω φαίνεται η διαφορά ανάμεσα στους χαρακτήρες των προγραμμάτων Alice v2.0 και Alice v3.0:

Alice v2.0

Alice v3.0

Με την χρήση του Alice v2.0 έχουν δημιουργηθεί κάποιες εφαρμογές και παιχνίδια όπως θα μελετήσουμε αναλυτικότερα στο τρίτο κεφάλαιο, αλλά με φτωχά γραφικά. Με το προγραμματιστικό εργαλείο Alice v3.0 αναμένεται να δημιουργηθούν παιχνίδια που τα γραφικά τους θα είναι άκρως ρεαλιστικά, ακόμη και παιχνίδια strategy η shoot em up.

Ερευνητικές εργασίες εφαρμογής του περιβάλλοντος Alice

Το πρόγραμμα Alice έχει χρησιμοποιηθεί σε αρκετές έρευνες και δημοσιεύσεις που κατά κύριο λόγο είχαν ως θέμα την εκπαιδευτική χρήση του Alice και την ανταπόκριση των μαθητών σε αυτό. Βέβαια το Alice χρησιμοποιήθηκε και για έρευνες σχετικά με την νέα προσέγγιση του προγραμματισμού που γίνεται μέσω ενός 3D γραφικού περιβάλλοντος. Μερικές τέτοιες δημοσιεύσεις παρατίθενται στην συνέχεια:

1. Η πρώτη δημοσίευση που θα παρουσιαστεί είναι: Joel C. Adams, Department of Computer Science, Calvin College, Grand Rapids, MI, "Alice, Middle Schoolers and the Imaginary World Camps", 38th SIGCSE technical symposium on Computer science education, 307-311, 2007.
Στην έρευνα αυτή γίνεται γνωστό ότι πολλές μαθήτριες που επιλέγουν το μάθημα του προγραμματισμού αισθάνονται ότι έχουν μικρότερη σχέση με το αντικείμενο του μαθήματος απ' ότι έχουν οι συμμαθητές τους του αντίθετου φύλλου. Επίσης την στιγμή που φτάνουν στο γυμνάσιο, πολλές νεαρές γυναίκες θεωρούν ότι το μάθημα του προγραμματισμού είναι για τους "σπασίκλες" και έτσι αποφεύγουν να το επιλέξουν. Αυτή η δημοσίευση παρουσιάζει ένα καλοκαιρινό πρόγραμμα με τίτλο "Imaginary World Camps", στο οποίο χρησιμοποιείται το πρόγραμμα Alice ώστε να λυθούν αυτά τα προβλήματα πριν οι μαθητές φτάσουν στο γυμνάσιο. Τα προκαταρκτικά αποτελέσματα φάνηκαν πολύ ενθαρρυντικά.
Με σχετικές έρευνες από το U.S. Bureau του Labor Statistics επιβεβαιώνεται ότι το 60% των αγοριών και 80% των κοριτσιών της Αμερικής απέφυγαν την περίοδο 2000 με 2004 την επιλογή του προγραμματισμού ως μάθημα παρακολούθησης. Αυτό μπορεί να έχει άμεσο αντίκτυπο στην οικονομία της Αμερικής καθώς οι δουλειές που σχετίζονται με υπολογιστές θα είναι από τις πλέον αναπτυσσόμενες δουλειές τις επόμενες δεκαετίες.
Γι αυτό το λόγο αποφασίστηκε να παρουσιαστεί στις μαθήτριες ένα δελεαστικό και ευκολονόητο εκπαιδευτικό εργαλείο διδασκαλίας προγραμματισμού το οποίο θα έλυνε αρχικά το πρόβλημα της προσέγγισης του προγραμματισμού και μετέπειτα το πρόβλημα της "οικειότητας" των κοριτσιών σε σχέση με τα αγόρια.
Το πρόγραμμα που επιλέχθηκε είναι το Alice. Έτσι το καλοκαίρι του 2003 ξεκίνησε ένα πιλοτικό πρόγραμμα για τους μαθητές του Λυκείου και το 2004, 2005 και 2006 ξεκίνησαν ολοκληρωμένα προγράμματα.
Το πρώτο, πιλοτικό πρόγραμμα ξεκίνησε το καλοκαίρι του 2003 και συμμετείχαν 3 αγόρια και 1 κορίτσι. Το πρόγραμμα διήρκεσε 1 εβδομάδα, δηλαδή 5 μέρες από τις 9.30 μέχρι τις 14.30. Η δομή του προγράμματος ήταν η εξής:

Ώρα	Δραστηριότητα
9.30	Παρουσίαση: Προγραμματισμός
10.00	Άσκηση: Προγραμματισμός
10.30	Ταινίες
11.30	Δραστηριότητα στην ύπαιθρο
11.45	Μεσημεριανό
12.20	Παρουσίαση: Δημιουργία ταινιών
12.45	Ταινίες
14.20	Σνακ
14.30	Αναχώρηση μαθητών

Το πρόγραμμα δούλεψε αρκετά καλά λόγω της διαφορετικότητας των δραστηριοτήτων της ημέρας και οι μαθητές ενθουσιάστηκαν με την δημιουργία δικών τους ταινιών. Έτσι αποφασίστηκε το 2004 να ξεκινήσουν ολοκληρωμένα προγράμματα. Τα ολοκληρωμένα προγράμματα ήταν των 8 ημερών, ξεχωριστά τμήματα για αγόρια και κορίτσια των 30 ατόμων το κάθε ένα. Μοιράστηκαν στα παιδιά εγχειρίδια του Alice αλλά και το πρόγραμμα, ώστε να μπορούν να διαβάζουν μόνα τους και να εξασκούνται. Το πρόγραμμα ονομάστηκε "The Imaginary Worlds Camps (IWC)" και τα δίδακτρα ήταν μόλις 260 δολάρια. Η δημογραφική σύσταση των τμημάτων δείχνει ότι τα κορίτσια συμμετείχαν σε αυτό το πρόγραμμα από μικρότερη ηλικία, κατά μέσο όρο το 63% των κοριτσιών ήταν 10 με 11 χρονών και το 69% των αγοριών ήταν 11 με 12.
Η δομή του νέου προγράμματος ήταν:

Ημέρα	Πρωινή Δραστηριότητα	Απογευματινή Δραστηριότητα
1	Αντικείμενα και μηνύματα	Δομές σεναρίου
2	Μέθοδοι	Κίνηση κάμερας, σκηνικό
3	Μεταβλητές και συναρτήσεις	Παράμετροι
4	Λίστες	Εφέ σκηνών
5	Επαναληπτικές δομές	Αποτίμηση, αξιολόγηση
6	Δομές ελέγχου	Ήχος
7	Συμβάντα	Τελικός έλεγχος
8	Έκθεση δημιουργημάτων για τους μαθητές και τις οικογένειες τους	

Πριν την παρακολούθηση του προγράμματος "The Imaginary Worlds Camps" και μετά την ολοκλήρωση της απαντήθηκαν ερωτηματολόγια από τους μαθητές. Αυτά είναι ενδεικτικά για την ανασκόπηση των αποτελεσμάτων του προγράμματος. Στην ερώτηση ποια είναι η άποψη σας για τους υπολογιστές το 56% των κοριτσιών πριν το πρόγραμμα και το 72% μετά απάντησαν ότι τους αρέσουν πολύ. Επίσης σχετικά με τις προγραμματιστικές τους ικανότητες το 36% πριν και το 52% μετά τα μαθήματα απάντησαν πολύ καλές, ενώ αυτός ο αριθμός έφτασε το 56% το 2006. Για την εμπειρία που έζησαν στο "The

Imaginary Worlds Camps" το 2006 το 81% των κοριτσιών και το 62% των αγοριών απάντησαν πολύ καλή.

Τα αποτελέσματα της έρευνας ήταν:

- Για τους μαθητές του "The Imaginary Worlds Camps " η δημιουργία των προσωπικών τους ταινιών ήταν ένα κίνητρο και μια άκρως ευχάριστη ενασχόληση
- Το πρόγραμμα Alice απαγορεύει την ύπαρξη συντακτικών λαθών, τα οποία είναι και ο κύριος παράγοντας για τον οποίο αποφεύγουν οι αρχάριοι τον προγραμματισμό.
- Επειδή τα λάθη δεν είναι συντακτικά, αλλά λογικά μπορούν εύκολα να διορθωθούν και να προκαλέσουν γέλιο αντί για απογοήτευση.

Το γενικό συμπέρασμα ήταν ότι τόσο τα αγόρια, όσο και τα κορίτσια απόλαυσαν την χρήση του Alice και παρακινούσαν το ένα το άλλο. Γι αυτό το λόγο γίνονται προσπάθειες να δημιουργηθούν σε διάφορα σχολεία καλοκαιρινά προγράμματα με βάση το πρόγραμμα Alice. Για την διευκόλυνση αυτών των προσπαθειών τα βιβλία του προγράμματος και οι ταινίες των μαθητών είναι διαθέσιμες στην ιστοσελίδα του "Imaginary Worlds Camps".

2. Μια άλλη έρευνα σχετικά με το Alice είναι: Caitlin Kelleher, Randy Pausch, and Sara Kiesler, Human Computer Interaction Institute, Carnegie Mellon University, Pittsburgh, "Storytelling Alice motivates middle school girls to learn computer programming", SIGCHI conference on Human factors in computing systems, 1455-1464, 2007.

Στην έρευνα αυτή παρουσιάζεται το Storytelling Alice, ένα πρόγραμμα για την συγγραφή ιστοριών το οποίο χρησιμοποιείται από τους μαθητές του λυκείου και τους βοηθάει να δημιουργήσουν τα δικά τους 3D προγράμματα. Η έρευνα εστιάζει σε μαθήτριες διότι αυτές συμμετέχουν σε μικρότερα ποσοστά σε μαθήματα προγραμματισμού. Το Alice υποστηρίζει την δημιουργία ιστοριών, σκηνών και παιχνιδιών παρέχοντας: 1) ένα σύνολο υψηλού επιπέδου κινήσεων, που υποστηρίζουν οι χαρακτήρες και με αυτό τον τρόπο μπορούν να αλληλεπιδράσουν μεταξύ τους, 2) μια συλλογή 3D χαρακτήρων και σκηνικών που ανταποκρίνονται σε κάθε είδους ιδέα του προγραμματιστή, 3) ένα εγχειρίδιο που εισάγει τους χρήστες στις έννοιες του προγραμματισμού και τους βοηθάει να δημιουργούν δικά τους παραδείγματα. Οι χρήστες του Alice και κυρίως τα κορίτσια, αποδείχτηκε ότι έμαθαν πολύ εύκολα και διασκεδαστικά τις βασικές δομές του προγραμματισμού με την χρήση του συγκεκριμένου προγράμματος. Παρακινούνταν περισσότερο από χρήστες που δούλευαν με άλλα προγράμματα η με απλές εκδόσεις της Alice και όχι της έκδοσης Storytelling, αφιέρωναν 42% περισσότερο χρόνο από κάθε άλλο χρήστη στις εφαρμογές τους, διαβεβαίωναν ότι θα χρησιμοποιούσαν και στο μέλλον νέες εκδόσεις του Alice και γενικότερα απέκτησαν ενδιαφέρον για τον προγραμματισμό.

Γενικότερα η αρχική έρευνα απέδειξε ότι τα κορίτσια δεν ασχολούνται με τον προγραμματισμό λόγω έλλειψης παρότρυνσης από τους γονείς, από τα άλλα

κορίτσια και τους δασκάλους, έλλειψης ενδιαφέροντος και λιγότερες ευκαιρίες ενασχόλησης με τους υπολογιστές. Είναι δύσκολο να διορθωθούν οι πολιτιστικοί παράγοντες που επηρεάζουν τα κορίτσια, αλλά είναι πιο εύκολο να γίνει δελεαστικότερη η διαδικασία εκμάθησης προγραμματισμού, το οποίο αποτελεί βασικό κίνητρο. Γι αυτό το λόγο έπρεπε να δημιουργηθεί ένα περιβάλλον εκμάθησης προγραμματισμού το οποίο να είναι δελεαστικό ακόμη και για τις μαθήτριες του λυκείου. Έτσι επιλέχτηκε το πρόγραμμα Storytelling Alice για τους παρακάτω λόγους:

- Τα κορίτσια μπορούν να σκεφτούν σε λίγο χρόνο μια ιστορία που θα θέλουν να δημιουργήσουν.
- Οι ιστορίες έχουν φυσική ροή και είναι απίθανο να απαιτήσουν εξεζητημένες προγραμματιστικές τεχνικές, έτσι είναι κατάλληλο για τους αρχάριους μαθητές.
- Οι ιστορίες είναι ένα είδος προσωπικής έκφρασης και παρέχουν στα κορίτσια μια ευκαιρία να πειραματιστούν σε διαφορετικούς ρόλους, μια σημαντική δραστηριότητα για την εφηβεία.
- Οι φίλοι που δεν ασχολούνται με τον προγραμματισμό μπορούν άμεσα να καταλάβουν και να εκτιμήσουν ένα σχέδιο με κίνηση, το οποίο μπορεί να αποτελέσει θετική ανάδραση.

Τα τρία κίνητρα που παρέχει το Alice είναι:

- Αποτελεί εργαλείο που επιτρέπει στα παιδιά την διερεύνηση ιδεών
- Αποτελεί μέσο προσωπικής έκφρασης
- Αποτελεί κύριο λίθο για την μετέπειτα καριέρα ενός προγραμματιστή

Η ανάπτυξη του Storytelling Alice διήρκεσε πάνω από 2 χρόνια και έγινε με την βοήθεια 200 κοριτσιών. Ουσιαστικά έγινε ένα τεστ παραγωγικότητας που περιελάμβανε τετράωρη απογευματινή εργασία και εβδομαδιαία προγράμματα με ομάδες των 3 έως 20 κοριτσιών ηλικίας από 10 μέχρι 17. Κατά την διάρκεια του τεστ, τα κορίτσια δημιουργούσαν διατάξεις σκηνών ταινιών και μετέπειτα προσπαθούσαν να τις υλοποιήσουν με μια έκδοση του Storytelling Alice.

Παράλληλα με το τεστ διεξάγονταν και τετράωρα μαθήματα στα οποία γινόταν μια σύγκριση συμπεριφοράς και επιδόσεων μεταξύ των κοριτσιών που χρησιμοποιούσαν το Storytelling Alice και κάποια άλλη έκδοση του Alice. Σε αυτή την αξιολόγηση συμμετείχε ένα σύνολο από 88 κορίτσια, τα 43 έκαναν χρήση του Storytelling Alice και τα 45 του Alice. Ο μέσος όρος ηλικίας ήταν 12,6 χρονών. Οι 76 συμμετέχουσες δήλωσαν ότι θα παρακολουθήσουν δημόσιο σχολείο και οι 12 ιδιωτικό. Για παρότρυνση δόθηκε το χρηματικό ποσό των 10 δολαρίων σε κάθε κοπέλα.

Οι συμμετέχουσες είχαν 2 ώρες και ένα τέταρτο να απαντήσουν κάποιες αρχικές ερωτήσεις και να δημιουργήσουν ένα πρόγραμμα με το αντίστοιχο πρόγραμμα Alice. Μετά είχαν 30 λεπτά να δοκιμάσουν την έκδοση της Alice που δεν είχαν επιλέξει. Στο τέλος οι συμμετέχουσες ζητήθηκαν να επιλέξουν

ένα από τα δυο προγράμματα για να πάρουν σπίτι και μετά να επιλέξουν ένα από τα προγράμματα που είχε δημιουργηθεί ως το καλύτερο.

Τα αποτελέσματα ήταν πολύ θετικά αφού έδειξαν ότι αυξήθηκε το ενδιαφέρον των μαθητών χωρίς όμως να μειωθεί η εκπαιδευτική αξία του προγράμματος. Έτσι το Storytelling Alice δεν εμπόδισε τα κορίτσια να μάθουν βασικές αρχές και τεχνικές προγραμματισμού. Το ίδιο ενθαρρυντικές ήταν και οι απαντήσεις που έδωσαν τα διαγωνιζόμενα κορίτσια σε ερωτήσεις σχετικά με την ευκολία χρήσης του Alice, με το κατά πόσο είναι ψυχαγωγικό το πρόγραμμα, αν θα ενδιαφέρονταν να ξαναχρησιμοποιήσουν το Alice στο μέλλον και κατά πόσο ενδιαφέρονται πλέον για μαθήματα προγραμματισμού.

Συνοψίζοντας η έρευνα κατέληξε στο ότι τόσο οι συμμετέχουσες που χρησιμοποίησαν το Storytelling Alice όσο και το Alice έμαθαν τεχνικές προγραμματισμού. Παρόλα αυτά αυτές που χρησιμοποίησαν το Storytelling Alice ασχολήθηκαν και είχαν σκοπό να ασχοληθούν και μελλοντικά περαιτέρω με τον προγραμματισμό. Βέβαια όλες οι συμμετέχουσες βρήκαν και τα δυο προγράμματα εξίσου ψυχαγωγικά και επιμορφωτικά.

3. Προχωρώντας θα εξεταστεί μια άλλη έρευνα: Barbara Moskal, Mathematics Department, Colorado School of Mines, Golden, Deborah Lurie, Mathematics Department, Saint Joseph's University, Philadelphia and Stephen Cooper, Computer Science Department, Saint Joseph's University, Philadelphia, "Evaluating the Effectiveness of a New Instructional Approach", 35th SIGCSE technical symposium on Computer science education, 75-79, 2004.
Η δημοσίευση κάνει μια αποτίμηση ενός εκπαιδευτικού ερευνητικού έργου που είχε ως χορηγό το ίδρυμα εθνικών επιστημών (NSF). Ο πρωταρχικός στόχος αυτού του ερευνητικού σχεδίου ήταν να αναπτυχθεί και να αξιολογηθεί μια σειρά μαθημάτων πανεπιστημίου, τα οποία θα σχεδιάζονταν έτσι ώστε να βελτιώσουν την απόδοση των φοιτητών σε εισαγωγικά θέματα προγραμματισμού. Τα αποτελέσματα αυτής της έρευνας αποδεικνύουν ότι η σειρά των μαθημάτων που αναπτύχθηκε, βελτίωσε την απόδοση των μαθητών και καλλιέργησε το ενδιαφέρον τους απέναντι στην επιστήμη των υπολογιστών.

Αυτό το άρθρο αναφέρει τα αποτελέσματα μιας μελέτης που αποτέλεσε μέρος μιας αρχικής σκέψης του ιδρύματος εθνικών επιστημών. Ο αρχικός σκοπός αυτής της έρευνας ήταν να καθοριστεί εάν οι θεμελιώδεις αρχές του αντικειμενοστραφή προγραμματισμού θα μπορούσαν να διδαχθούν σε μαθητές οι οποίοι θα είχαν ελάχιστο ή ανύπαρκτο προγραμματιστικό υπόβαθρο εισάγοντας μια καινοτομική προσέγγιση που αναπτύχθηκε από τους Wanda Dann και Stephen Cooper. Στα μαθήματά τους οι Wanda Dann και Stephen Cooper παρατήρησαν ότι πολλοί μαθητές που δεν είχαν καθόλου ή είχαν μικρή επαφή με τον προγραμματισμό διέτρεχαν κίνδυνο αποτυχίας στο πρώτο μάθημα προγραμματισμού που μπορεί να ήταν δύσκολο και αυστηρό. Έτσι οι παραπάνω αποφάσισαν να αντιμετωπίσουν αυτό το πρόβλημα δημιουργώντας ένα νέο μάθημα στο οποίο θα χρησιμοποιούνταν ένα πρόγραμμα 3D, το Alice που αναπτύχθηκε στο Carnegie Mellon University. Οι Dann και Cooper επέλεξαν αυτό το εργαλείο γιατί θεώρησαν ότι το γραφικό περιβάλλον προγραμματισμού θα υποκινούσε τους φοιτητές.

Μοιράστηκαν εγχειρίδια χρήσης και νέα βιβλία. Αντικείμενο του μαθήματος Alice και της έρευνας αυτής ήταν να εξεταστεί η αποτελεσματικότητα του Alice στην εκμάθηση βασικών αρχών προγραμματισμού στους μαθητές και κατά πόσο θα τους κινούσε το ενδιαφέρον. Η έρευνα διήρκεσε 2 χρόνια και στο Saint Joseph's University (SJU) και στο Ithaca College (IC).

Το πρόγραμμα Alice επιλέχθηκε για τους εξής λόγους:

- Η εργασία με την βοήθεια ενός 3D περιβάλλοντος είναι ελκυστική και υποκινεί την σημερινή γενιά
- Η οπτική φύση και η άμεση ανάδραση του προγράμματος, κάνει εύκολο στους μαθητές να δουν την επίδραση μιας εντολής που έχουν δώσει. Επιπλέον κάνει το τεστ του προγράμματος ευκολότερο
- Ο συντάκτης σκηνών που λειτουργεί με σύρσιμο και επιλογή, εμποδίζει τους μαθητές από το να κάνουν συντακτικά λάθη στα οποία οι αρχάριοι είναι ευάλωτοι.
- Οι κλάσεις και τα 3D αντικείμενα στο Alice παρέχουν μια εμφανή ιδέα και περιγραφή του αντικειμένου.

Χρησιμοποιώντας το Alice, δεν παραλήφθηκε καμία από τις βασικές έννοιες του προγραμματισμού –συναρτήσεις, μέθοδοι, κλάσεις, κληρονομικότητα- που χρησιμοποιούνταν στην Java και την C++.

Στο Saint Joseph's University (SJU) αλλά και στο και στο Ithaca College (IC) τα άτομα που ήταν νέα στην επιστήμη των υπολογιστών ήταν 25 με 30. Στο τέλος του πρώτου χρόνου οι μαθητές που εκδήλωναν ενδιαφέρον για μαθήματα προγραμματισμού ήταν ελάχιστοι και ειδικά αυτοί που δεν είχαν καθόλου υπόβαθρο άγγιζαν το 0%. Οπότε οι συμμετέχοντες στην έρευνα αυτή ήταν οι πρωτοετείς φοιτητές και κατά κύριο λόγο αυτοί που δεν είχαν επαφή με προγραμματισμό και μαθηματικά. Στο IC το Alice γινόταν παράλληλα με το βασικό μάθημα προγραμματισμού, ενώ στο SJU το μάθημα του Alice έλαβε χώρα πριν το βασικό μάθημα.

Για την έρευνα χρησιμοποιήθηκαν τα εξής γκρουπ:

Treatment Group: Μαθητές χωρίς προγραμματιστική εμπειρία που συμμετείχαν στο μάθημα Alice.

Control Group1: Μαθητές χωρίς προγραμματιστική εμπειρία που δεν συμμετείχαν στο μάθημα Alice.

Control Group2: Μαθητές με προγραμματιστική εμπειρία που δεν συμμετείχαν στο μάθημα Alice. Με N εκφράζεται ο αριθμός των μαθητών σε κάθε γκρουπ. Η ανάλυση των αποτελεσμάτων έγινε με το πρόγραμμα SPSS v11.5.

Table 2. Grades (GPA) in CS1 in Each Group

Group	2001-2002		2002-2003		2-Year Total	
Treatment	**N**	**GPA**	**N**	**GPA**	**N**	**GPA**
High Risk	7	2.86	12	3.05	19	2.98
Med. Risk	2	3.65	4	2.93	6	3.17
Total	9	3.04	16	3.02	25	3.03
Control Group1	**N**	**GPA**	**N**	**GPA**	**N**	**GPA**
High Risk	10	1.32	2	0.50	12	1.18
Med. Risk	14	2.36	4	2.75	18	2.45
Total	24	1.93	6	2.00	30	1.94
Control Group2	**N**	**GPA**	**N**	**GPA**	**N**	**GPA**
Low Risk	19	2.28	3	3.33	22	2.43
Not At Risk	7	3.34	23	3.51	30	3.47
Total	26	2.57	26	3.49	52	3.03
Total of all students	**N**	**GPA**	**N**	**GPA**	**N**	**GPA**
	59	2.38	48	3.15	107	2.73

Από τον παραπάνω πίνακα γίνεται φανερό ότι ο μέσος όρος των μαθητών που δεν είχε επαφή με προγραμματισμό άλλα παρακολούθησε το μάθημα Alice είχε μέσο όρο βαθμών 3.03, ενώ αυτοί που δεν παρακολούθησαν είχαν 1.94.

Επίσης στον παρακάτω πίνακα φαίνονται και τα ποσοστά ένδειξης ενδιαφέροντος των μαθητών:

Table 3. Percentage Retention in CS1 in Each Group

Group	% retained 2001-2002	% retained 2002-2003	2-Year Total
Treatment			
High Risk	86 (6/7)	83 (10/12)	84 (16/19)
Med. Risk	100 (2/2)	100 (4/4)	100 (6/6)
Total	89 (8/9)	87 (14/16)	88 (22/25)
Control Group1			
High Risk	10 (1/10)	50 (1/2)	15 (2/12)
Med. Risk	57 (8/14)	100 (4/4)	67 (12/18)
Total	37 (9/24)	83 (5/6)	47 (14/30)
Control Group2			
Low Risk	63 (12/19)	100 (3/3)	68 (15/22)
Not At Risk	86 (6/7)	78 (18/23)	80 (24/30)
Total	69 (18/26)	81 (21/26)	75 (39/52)
Total of all students	59 (35/59)	83 (40/48)	70 (75/107)

Από αυτό φαίνεται ότι το 88% του συνόλου των μαθητών που παρακολούθησαν το Alice σε αντίθεση με το μόλις 47% αυτών που δεν παρακολούθησαν αισθάνονται σίγουροι και θα παρακολουθούσαν και επόμενα μαθήματα προγραμματισμού.

Μετά από όλα αυτά η έρευνα καταλήγει στο ότι οι μαθητές που χρησιμοποίησαν το Alice, όχι μόνο θα συμμετείχαν και σε άλλα παρόμοια μαθήματα, αλλά θα παρότρυναν και τους συμμαθητές τους. Επίσης αυτοί που συμμετείχαν στο μάθημα με το Alice τα πήγαν εξίσου καλά όσο οι μαθητές που είχαν εμπειρία στον προγραμματισμό με Java και C++. Το Alice δηλαδή ανέβασε το επίπεδο προγραμματιστικών γνώσεων των μαθητών.

4. Τέλος γίνεται η παρουσίαση της εργασίας: Stephen Cooper, Computer Science Department, Saint Joseph's University, Philadelphia, Wanda Dann, Computer Science Department, Ithaca College, NY and Randy Pausch, Computer Science Department, Carnegie Mellon University, Pittsburgh, “Teaching objects-first in Introductory Computer Science”, 34th SIGCSE technical symposium on Computer science education,191-195, 2003.

Η μελέτη σχετίζεται με την στρατηγική διδασκαλίας εισαγωγικών θεμάτων προγραμματισμού η οποία βασίζεται στα αντικείμενα. Ασχολείται με την απεικόνιση και οπτικοποίηση των αντικειμένων και την χρήση 3D περιβάλλοντος για την κίνηση τους. Η νέα μέθοδος διδασκαλίας απασχολεί πολλούς καθηγητές προγραμματισμού επειδή έχει αποδειχτεί στατιστικά αλλά και ανεπίσημα ότι η απόδοση των μαθητών εξαρτάται και επηρεάζεται άμεσα από αυτή τη νέα προσέγγιση διδασκαλίας. Επίσης γίνεται μια σύγκριση από παιδαγωγικής άποψης της νέας αυτής προσέγγισης με άλλη σχετική εργασία.

Η ACM υποστηρίζει ότι η τεχνική με βάση τα αντικείμενα είναι η πιο ενδιαφέρουσα και πιο αποτελεσματική τεχνική εκμάθησης εισαγωγικών αρχών προγραμματισμού. Αυτή η τεχνική δίνει έμφαση στις αρχές του αντικειμενοστραφούς προγραμματισμού και σχεδιασμού από το πρώτο κιόλας μάθημα. Η τεχνική ξεκινάει με τις έννοιες των αντικειμένων και της κληρονομικότητας, συνεχίζει με τις πιο παραδοσιακές δομές ελέγχου αλλά πάντα μέσα στα πλαίσια του αντικειμενοστραφούς σχεδιασμού.

Για να γίνει μια ενδιαφέρουσα προσέγγιση αυτής της τεχνικής δημιουργήθηκαν αρκετά προγραμματιστικά εργαλεία που να προωθούν την τεχνική που έχει ως βάση τα αντικείμενα. Σε αυτή τη μελέτη γίνεται λόγος για το προγραμματιστικό εργαλείο Alice που έχει 3D περιβάλλον εργασίας. Το 3D περιβάλλον εργασίας βοηθάει στην νοερή απεικόνιση των αντικειμένων και τους μαθητές να καταλάβουν και να “δουν” τις βασικές αντικειμενοστραφείς αρχές. Έτσι αν κάποιος μαθητής δημιουργήσει ένα αντικείμενο βάτραχο δεν χρειάζεται να γράψει κώδικα, παρά να προσθέσει το αντικείμενο στο πρόγραμμα που υλοποιεί τον κόσμο του. Με αυτό τον τρόπο το αποτέλεσμα θα είναι το εξής:

Μετά από 3 χρόνια χρήσης του Alice παρατηρήθηκε ότι οι μαθητές:

- Ανέπτυξαν ικανότητα σχεδιασμού προσεγμένων και αξιόλογου περιεχομένου προγραμμάτων
- Ανέπτυξαν ικανότητα δημιουργίας αξιόλογων εικονικών κόσμων. Στους κόσμους που δημιουργούσαν οι μαθητές τα πάντα ήταν αντικείμενα που εκτελούσαν κινήσεις όπως σε ταινίες και σε videogames
- Έμαθαν να τεστάρουν και να επιδιορθώνουν κώδικα. Επειδή οι μαθητές γνώριζαν ποιο κομμάτι κώδικα αναφέρεται σε κάποιο αντικείμενο, με ξεχωριστές αλλαγές και αφού έβλεπαν τι επίδραση έχει στο αντικείμενο έκαναν τις κατάλληλες επιδιορθώσεις
- Έμαθαν να δημιουργούν το πρόγραμμα τους κομμάτι κομμάτι και να κάνουν παράλληλα τεστ σωστής λειτουργίας
- Ασχολήθηκαν με την κληρονομικότητα. Οι μαθητές έγραφαν κώδικα ώστε να δημιουργήσουν μια πιο ισχυρή κλάση
- Συνεργάστηκαν. Οι μαθητές δημιουργούσαν τους δικούς τους χαρακτήρες και μετά τους συνδύαζαν ώστε να υλοποίησαν ένα κοινό κόσμο
- Απέφυγαν τα λάθη. Οι μαθητές δεν μπορούν να κάνουν λογικά λάθη αφού ο συντάκτης τους απαγορεύει να δώσουν λανθασμένη τιμή σε μια μεταβλητή η σε ένα επαναληπτικό βρόγχο

Τα αποτελέσματα αυτής της έρευνας φαίνονται στον παρακάτω πίνακα:

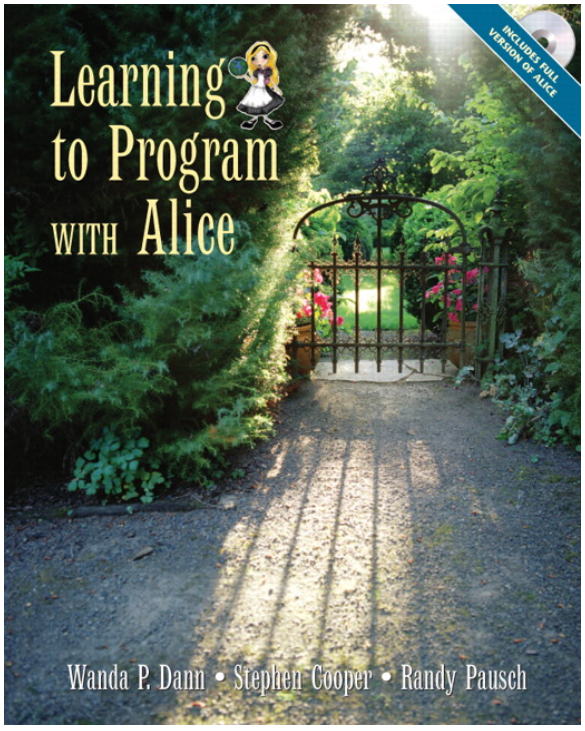
Statistics	All	Test	Control
# Students	49	11	10
Mean	2.49	2.8	1.3
Median	2.75	3	1.25
Variance	1.62	0.75	1.22

Οι μαθητές στο Ithaca College και στο Saint Joseph's University παρακολούθησαν ένα μάθημα Alice το 2001 με 2002. Επιλέχθηκαν οι 21 πιο αδύναμοι μαθητές στο αρχικό μάθημα του προγραμματισμού. Από αυτούς οι 11, όπως φαίνεται και στον πίνακα παρακολούθησαν το Alice ενώ οι 10 όχι. Τα αποτελέσματα δείχνουν ότι οι 11 μαθητές που παρακολούθησαν τα πήγαν καλύτερα όχι μόνο από τους άλλους 10, αλλά και από το σύνολο των 49 μαθητών της τάξης οι οποίοι μπορεί να είχαν ένα προγραμματιστικό υπόβαθρο. Ο δείκτης Variance αντιστοιχεί στο ποσοστό αποτυχίας των μαθητών σε μάθημα προγραμματισμού. Έτσι εφόσον είναι ο μικρότερος των τριών τιμών, αποδεικνύει ότι το μικρότερο ποσοστό αποτυχίας ήταν των μαθητών που παρακολούθησαν το Alice. Επίσης από το σύνολο των μαθητών που παρακολούθησαν το Alice το 91% παρακολούθησε και στο επόμενο εξάμηνο μάθημα προγραμματισμού. Παρόλο που αυτά τα στοιχεία είναι ενδεικτικά και άκρως πειστικά, ο αριθμός των μαθητών που συμμετείχε σε αυτή την έρευνα είναι μικρός και έτσι οι έρευνα συνεχίζεται λεπτομερέστερα και για ένα μεγαλύτερο πλήθος μαθητών έτσι ώστε να προκύψουν πιο ισχυρά στοιχεία.

ΚΕΦΑΛΑΙΟ 2
ΕΚΜΑΘΗΣΗ ΠΡΟΓΡΑΜΜΑΤΙΣΜΟΥ ΜΕ ΤΟ ALICE

Εκμάθηση προγραμματισμού με το Alice

Στο κεφάλαιο αυτό παρουσιάζεται η μετάφραση στα ελληνικά του βιβλίου “Learning to program with Alice των Wanda Dann, Stephen Cooper και Randy Pausch. Στόχος αυτού του κεφαλαίου είναι η εμβάθυνση στο πρόγραμμα Alice και η κατανόηση βασικών αρχών του αντικειμενοστραφούς προγραμματισμού. Το βιβλίο αυτό αποδεικνύει τα όσα έχουν ειπωθεί παραπάνω σχετικά με την ευκολία χρήσης του Alice και την γρήγορη και διασκεδαστική υλοποίηση προγραμμάτων. Το βιβλίο ξεκινάει με απλές προγραμματιστικές έννοιες και καταλήγει σε κλάσεις, μεθόδους, κληρονομικότητα και αλληλεπίδραση με το χρήστη, έννοιες οι οποίες είναι βασικές για τον αντικειμενοστραφή προγραμματισμό.



PEARSON PRENTICE HALL

Wanda Dann
Stephen Cooper
Randy Pausch

“Εκμάθηση προγραμματισμού με το Alice”

Περιεχόμενα

Μέρος Ι: Εισαγωγή στο Alice……..……..26

1 Ξεκίνημα με το Alice……..……..27

1-1 Εισαγωγή στο Alice……..……27

1-2 Αρχικά για το Alice……..……..31

- Μυστικά και τεχνικές 1: Ειδικά εφέ: Κείμενα και δισδιάστατες γραφικές εικόνες……..……36
- Ασκήσεις……..38
- Περίληψη……39

2 Σχεδιασμός και υλοποίηση προγράμματος ……...41

2-1 Σενάρια και σκηνές έργου……..……41

2-2 Ένα αρχικό πρόγραμμα…….……46

- Μυστικά και τεχνικές 2: Οδηγίες για προσανατολισμό και κίνηση ……..……55
- Ασκήσεις……..59
- Περίληψη……61

3 Προγραμματισμός: Συναρμολόγηση των κομματιών..……..…...63

3-1 Ολοκληρωμένες συναρτήσεις και παραστάσεις……..……64

3-2 Απλές δομές ελέγχου…….……67

- Μυστικά και τεχνικές 3: Σχεδιασμός όψης και αίσθησης……..……73
- Ασκήσεις……..74
- Περίληψη……76

Μέρος ΙΙ: Αντικειμενοστραφής προγραμματισμός και προγραμματισμός οδηγούμενος από γεγονότα……....78

4 Κλάσεις, αντικείμενα, μέθοδοι και παράμετροι……...79

4-1 Μέθοδοι επιπέδου κλάσης και κληρονομικότητα……..81

- Μυστικά και τεχνικές 4: Ορατά και αόρατα αντικείμενα……92
- Ασκήσεις……..94
- Περίληψη……95

5 Αλληλεπίδραση: Γεγονότα και χειρισμός γεγονότων…..…....97
5-1 Αλληλεπιδρών προγραμματισμός……...97
- Μυστικά και τεχνικές 5: Γεγονότα…..……102
- Ασκήσεις……..…..103
- Περίληψη……...…105

Παραρτήματα.……107
Παράρτημα Α: Χρήση του Alice……..…..108
Μέρος 1: Εκτέλεση οπτικού κόσμου στο Alice……..…....108
Μέρος 2:Χρήση popup μενού για την δημιουργία μιας αρχικής σκηνής…….……..……..…..117
Παράρτημα Β: Διαχείριση του Interface του Alice……...…...121
Ευρετήριο……..……xxx

Μέρος Ι:
Εισαγωγή στο Alice

Κεφάλαιο 1 Ξεκίνημα με το Alice

1-1 Εισαγωγή στο Alice

Γιατί να μάθει κανείς προγραμματισμό;

Υποθέτουμε ότι διαβάζετε αυτό το βιβλίο επειδή είτε α) θέλετε να μάθετε για τον προγραμματισμό ή β) παρακολουθείτε κάποιο μάθημα στο οποίο απαιτείται να μάθετε προγραμματισμό υπολογιστών. Σε κάθε περίπτωση, ας ξεκινήσουμε συζητώντας γιατί μπορεί να είναι χρήσιμη σε εσάς το να μάθετε πώς να γράφετε προγράμματα σε ηλεκτρονικό υπολογιστή.

Πρώτα ας ξεκαθαρίσουμε κάτι: το να μάθετε να προγραμματίζετε δεν σημαίνει ότι μεταμορφώνεστε σε "κολλημένους κομπιουτεράκηδες". Γνωρίζουμε ότι υπάρχουν πολλές απόψεις σχετικά με το τι είναι προγραμματισμός υπολογιστών και με το τι είδους άνθρωποι γράφουν προγράμματα. Αλλά υποσχόμαστε ότι δεν θα επιθυμήσετε να φορέσετε ένα κάλυμμα προστασίας για την τσέπη και τα λεφτά σας ή θα ξεκινήσετε να μιλάτε με μια δυσνόητη προγραμματιστική γλώσσα. Ειλικρινά. Αυτό το βιβλίο χρησιμοποιεί ένα σύστημα που καλείται Alice, το οποίο βοηθάει στην υλοποίηση προγραμμάτων με ένα τρόπο πολύ διαφορετικό από τους συνηθισμένους. Δεν γράφετε κώδικα σε μια μηχανή με σκοπό να κάνει υπολογισμούς, αλλά γίνεστε ο σκηνοθέτης ενός έργου, όπου τα αντικείμενα που βλέπετε στην οθόνη ενεργούν σύμφωνα με το σενάριο που έχετε δημιουργήσει. Αλλά ας μην πάμε μακριά, ας γυρίσουμε στο γιατί θέλετε να δημιουργήσετε προγράμματα στον ηλεκτρονικό υπολογιστή.

Υπάρχουν πολλοί λόγοι για να θέλετε να μάθετε να προγραμματίζετε. Για πολλούς ο προγραμματισμός είναι τρόπος διασκέδασης. Αλλά για τους περισσότερους ο προγραμματισμός είναι εργασία. Έχουν κάτι σημαντικό να κάνουν, και ο υπολογιστής αποτελεί το κατάλληλο εργαλείο. Στην πραγματικότητα οι εφαρμογές των υπολογιστών είναι τόσο διεισδυτικές στην κοινωνία μας, ώστε αν για παράδειγμα γεννιόσασταν αύριο θα είχατε επαφή με αυτές από την ημέρα της γέννησης μέχρι την τελευταία σας ημέρα. Πολλά νοσοκομεία δένουν ένα μικροτσίπ στον αστράγαλο των νεογέννητων για να γνωρίζουν που βρίσκονται για όλη τους τη ζωή. Την τελευταία σας ημέρα, είναι πιθανόν να έχετε ένα μόνιτορ ενός υπολογιστή που θα δείχνει το καρδιακό σήμα όταν θα βρίσκεστε στο κρεβάτι. Στο μεταξύ μπορείτε να ζήσετε πιο πολύ και πιο υγιείς, εξαιτίας των προόδων που έγιναν με την βοήθεια υπολογιστών στην ιατρική έρευνα, στην τεχνολογία, στα ρυθμιζόμενα από υπολογιστές φρένα και αερόσακους στα αυτοκίνητα και στην μοντελοποίηση των φαρμάκων από υπολογιστές που βοηθούν στην αντιμετώπιση ασθενειών όπως το AIDS. Οι προγραμματιστές υπολογιστών βοηθούν στο να γίνουν αληθινές όλες αυτές οι τεχνολογικές πρόοδοι.

Οι υπολογιστές και τα προγράμματα που γράφονται έχουν ανατρέψει την βιομηχανία της ψυχαγωγίας. Τα παιχνίδια στους υπολογιστές έχουν γίνει άκρως δημοφιλή. Οι Pew Internet και American Life Project ανέφεραν (το 2003) ότι περίπου 70% των φοιτητών παίζουν διαδικτυακά παιχνίδια τουλάχιστον μια φορά την εβδομάδα. Τα γραφικά των ταινιών Star Wars γίνονται μόνο μέσω υπολογιστών. Παρεμπιπτόντως ένας προπτυχιακός

φοιτητής, συγγραφέας της Alice, αποφοίτησε και πήγε για δουλειά στην ILM (Βιομηχανικό Φως και Μαγεία), και εκεί έκανε τα ειδικά εφέ για το Star Wars.

Οι υπολογιστές μας βοηθούν στην επικοινωνία, υποστηρίζοντας σύνθετα κινητά δίκτυα τηλεφωνίας, βοηθούν στην θαλάσσια έρευνα παρακολουθώντας τα ίχνη των αποδημητικών ζώων και μας επιτρέπουν να εξερευνούμε το διάστημα. Κανένα από αυτά δεν θα ήταν δυνατό χωρίς τους υπολογιστές.

Φυσικά πολλοί από τους ανθρώπους που γράφουν προγράμματα είναι επαγγελματίες που πέρασαν πολλά χρόνια μελετώντας. Αλλά ακόμη και οι άνθρωποι που δεν σκοπεύουν να γίνουν επαγγελματίες μπορούν να επωφεληθούν ακόμη και από ένα απλό μάθημα προγραμματισμού. Οι σύγχρονες εφαρμογές, όπως τα προγράμματα λογιστικού φύλλου και ο επεξεργαστής κειμένου, δίνουν στον τελικό χρήστη την ευκαιρία να κερδίσουν χρόνο και προσπάθεια χρησιμοποιώντας μακροεντολές ή άλλα χαρακτηριστικά του προγραμματισμού τα οποία υποχρεώνουν τον υπολογιστή να κάνει κάτι χρονοβόρο και βαρετό, αντί να το κάνει ο χρήστης. Επίσης αν έχετε μια μικρή εμπειρία στον προγραμματισμό μπορείτε να γίνετε το “χρήσιμο άτομο” στο γραφείο που δουλεύετε.

Κυριότερα ένα μάθημα προγραμματισμού μπορεί να καλλιεργήσει ένα νέο τρόπο σκέψης, όπως και ένα μάθημα ζωγραφικής μπορεί να βοηθήσει στο πως μπορεί κανείς να δει τον κόσμο με διαφορετικό μάτι. Το να μάθετε να σκέφτεστε με νέους τρόπους είναι πάντα πολύ χρήσιμο. Πολλοί από εμάς αναζητούμε τρόπους για την βελτίωση μας στην λύση προβλημάτων. Με τον όρο λύση προβλημάτων εννοούμε να δίνει κανείς την σωστή απάντηση σε ένα πρόβλημα ή να εκτελεί μια αποστολή. Ο προγραμματισμός υπολογιστών είναι ένα είδος επίλυσης προβλημάτων. Έτσι μαθαίνοντας να προγραμματίζετε ένα υπολογιστή θα σας βοηθήσει να αναπτύξετε ένα νέο τρόπο σκέψης-καθιστώντας σας ικανό να βρίσκετε απαντήσεις σε ερωτήσεις και να βρίσκετε πώς να επιλύετε προβλήματα.

Πως θα μάθετε να προγραμματίζετε με αυτό το βιβλίο και με το Alice

Αυτό το βιβλίο και το σύστημα Alice θα σας διδάξει πώς να κάνετε ένα πρόγραμμα σε υπολογιστή, αλλά με μια θεμελιώδη διαφορά και με ένα διασκεδαστικό τρόπο. Δουλέψαμε πολύ σκληρά για να σας διευκολύνουμε στην εκμάθηση προγραμματισμού. Πολλά αρχικά μαθήματα προγραμματισμού ξεκινούν με υπολογισμούς και εκτυπώσεις μέσων όρων και αθροισμάτων. Πολλές φορές οι μαθητές αποθαρρύνονται γιατί πρέπει να μάθουν πολλές τεχνικές λεπτομέρειες μέχρι να λειτουργήσουν όλα σωστά. Εμείς πιστέψαμε ότι υπάρχει ευκολότερος τρόπος.

Το βιβλίο αυτό χρησιμοποιεί μια εντελώς διαφορετική προσέγγιση η οποία επιτεύχθηκε προσφάτως με την αύξηση της δύναμης των υπολογιστών και την δημιουργία νέου λογισμικού το οποίο χρησιμοποιεί αυτές τις αυξημένες δυνατότητες, κυρίως τα τριών διαστάσεων γραφικά. Το σύστημα Alice το οποίο παρέχεται δωρεάν ως δημόσια παροχή από το πανεπιστήμιο Carnegie Mellon, παρέχει μια εντελώς νέα προσέγγιση στην εκμάθηση προγραμματισμού. Αρχικά αναπτύχθηκε στα πλαίσια ενός ερευνητικού project

για την εικονική πραγματικότητα. Το πρόγραμμα Alice σας επιτρέπει να είστε ο σκηνοθέτης μιας ταινίας, ή ο δημιουργός ενός video game, όπου τα 3D αντικείμενα κινούνται τριγύρω σε ένα εικονικό κόσμο στην οθόνη ανάλογα με τις κατευθύνσεις που τους δίνετε. Δεν χρησιμοποιείται δύσκολες προγραμματιστικές εντολές παρά απλές καθημερινές Αγγλικές λέξεις όπως: “προχώρησε μπροστά” ή ”στρίψε δεξιά ”. Το καλύτερο από όλα είναι ότι δεν μπορείτε να κάνετε λάθη! Βέβαια, πάντα μπορείτε να κάνετε κάποια λάθη όπως για παράδειγμα να πείτε το αντικείμενο να μετακινηθεί μπροστά ενώ εσείς θέλατε πίσω. Δεν μπορείτε όμως να κάνετε λάθη που δεν κατανοεί ο υπολογιστής, όταν πληκτρολογήσετε κάτι λάθος, τα οποία έχουν ως αποτέλεσμα να μην τρέχει το πρόγραμμά σας καθόλου.

Εάν ο όρος προγραμματιστής υπολογιστών σας φέρνει στο μυαλό ένα “ταλαίπωρο” άνθρωπο, πάνω από ένα πληκτρολόγιο σε ένα σκοτεινό δωμάτιο-μην ανησυχείτε! Δεν θα ακουμπήσετε σχεδόν ποτέ το πληκτρολόγιο όσο χρησιμοποιείται το Alice. Θα δημιουργήσετε προγράμματα επιλέγοντας λέξεις και αντικείμενα και μεταφέροντας τα στην οθόνη με το ποντίκι. Τότε μόλις θα πατήσετε το κουμπί “Play” (κυκλωμένο στην εικόνα 1-1-1), τα αντικείμενα στον 3D κόσμο της οθόνης σας θα ζωντανέψουν και θα ακολουθήσουν το σενάριο που τους έχετε γράψει! Έτσι το να είστε προγραμματιστής υπολογιστών χρησιμοποιώντας το Alice είναι σαν να είστε σκηνοθέτης ταινίας ή χορογράφος-ο καθένας που δίνει οδηγίες στους ανθρώπους για το τι να κάνουν με ένα ακριβές και περιορισμένο λεξιλόγιο.

ΕΙΚΟΝΑ 1-1-1. Ένας 3D κόσμος της Alice.

Αφού μάθατε πώς να χρησιμοποιείτε το Alice, θα καταλάβετε όλες τις θεμελιώδεις ιδέες του προγραμματισμού. Τότε θα είστε έτοιμοι να χρησιμοποιήσετε μια από τις συνηθισμένες γλώσσες προγραμματισμού στις οποίες πρέπει να πληκτρολογήσετε και να βάλετε όλα τα κόμματα και τις τελείες στην σωστή θέση. Θα γνωρίζετε πως να προγραμματίζετε και το μόνο που θα πρέπει να μάθετε είναι οι συγκεκριμένοι κανόνες σύνταξης των γλωσσών όπως Java, C++, C# ή όποια άλλη.

Τα βασικά του προγραμματισμού υπολογιστών

Ένα πρόγραμμα στον υπολογιστή είναι τίποτα περισσότερο από μία σειρά οδηγιών οι οποίες καθοδηγούν τον υπολογιστή τι να κάνει. Φυσικά υπάρχουν πολλοί τρόποι με τους οποίους μπορείτε να πείτε στον υπολογιστή να κάνει κάτι, οπότε παίζει ρόλο το πώς θα το κάνετε. Οι προγραμματιστές υπολογιστών συχνά χρησιμοποιούν όρους όπως “κομψό” για να περιγράψουν καλογραμμένα προγράμματα. Συνιστούμε να βλέπετε ένα πρόγραμμα υπολογιστή όχι μόνο ως τον τρόπο που θα πείτε στον υπολογιστή τι να κάνει άλλα και ως ένα τρόπο να πείτε σε έναν άλλο άνθρωπο το τι θέλετε να κάνει ο υπολογιστής.

Ένα πρόγραμμα υπολογιστή δεν είναι μόνο ο τρόπος για να πείτε στον υπολογιστή τι να κάνει

Ένα πρόγραμμα υπολογιστή είναι και ο τρόπος να πείτε σε έναν άλλο άνθρωπο τι θέλετε να κάνει ο υπολογιστής.

Αυτό κάνει πολύ ευκολότερο το να κριθεί κάτι ως κομψό. Ένα πρόγραμμα είναι κομψό εάν άλλοι άνθρωποι μπορούν εύκολα να καταλάβουν και να εκτιμήσουν τις προθέσεις του αρχικού προγραμματιστή. Γι αυτό το λόγο, ένα βασικό θέμα στην γραφή ενός προγράμματος είναι να εμπεριέχει τεκμηρίωση (σχόλια στο πρόγραμμα, μια ιστοσελίδα για παραπομπή ή ένα συνοδευτικό έγγραφο) η οποία βοηθάει κάποιο άλλο άτομο να καταλάβει τι προσπαθείτε να κάνετε.

Το κλειδί στον προγραμματισμό υπολογιστών είναι να καταλάβετε τις βασικές θεμελιώδεις ιδέες. Ο προγραμματισμός είναι πολύ εύκολος. Όλα τα προγράμματα αποτελούνται από απλά σχέδια:

Μία λίστα οδηγιών: Για παράδειγμα, “Χτυπήστε τα αυγά, αναμίξτε με την ζάχαρη, χύστε το μίγμα σε ένα ταψί και ψήστε στους 375 βαθμούς για 45 λεπτά.” Οι επιστήμονες υπολογιστών το ονομάζουν διαδοχική επεξεργασία.

If (υποθέσεις): Για παράδειγμα, “Εάν βρέχει, πάρε μια ομπρέλα.” Οι επιστήμονες υπολογιστών το ονομάζουν εκτέλεση υπό συνθήκη (εκτέλεση υπό όρους).

Επαναληπτική συμπεριφορά: Για παράδειγμα, “Χτύπα το πόδι πέντε φορές” ή “Όσο υπάρχουν κουλουράκια στο πιάτο, συνέχισε να τρως.” Οι επιστήμονες υπολογιστών το ονομάζουν επανάληψη.

Να χωρίζετε τα πράγματα σε μικρότερα κομμάτια: Για παράδειγμα, “Ο τρόπος που θα καθαρίσεις το σπίτι είναι πρώτα να καθαρίσεις την κουζίνα, μετά το μπάνιο, μετά καθάρισε κάθε ένα από τα τρία δωμάτια.” Οι επιστήμονες υπολογιστών το ονομάζουν αποσύνθεση προβλήματος, ή βηματική διύλιση ή από πάνω προς τα κάτω σχεδίαση, αλλά είναι πραγματικά μια παλιά φιλοσοφική προσέγγιση που καλείται υπεραπλούστευση. Όπως και να το ονομάσετε σημαίνει ότι μια πολύπλοκη διεργασία, χωρίζεται σε μικρότερες και απλούστερες διεργασίες. Το αποτέλεσμα της εκτέλεσης των απλών διεργασιών είναι η εκτέλεση και της σύνθετης διεργασίας.

Υπολογισμός του αποτελέσματος: Εδώ εκτελούμε μια ακολουθία βημάτων ώστε να επιτύχουμε ένα αποτέλεσμα που είναι απάντηση σε μια ερώτηση. Για παράδειγμα: “Κοίταξε στον τηλεφωνικό κατάλογο και βρες τον αριθμό της Ρεβέκα Σμιθ, ” ή “Βάλε το μωρό στην ζυγαριά και πες μου πόσα κιλά είναι.” Στην πραγματικότητα κάθε μια από αυτές τις ενέργειες εμπεριέχει μια ερώτηση: “Ποιο είναι το τηλέφωνο της Ρεβέκας;” ή “Πόσο ζυγίζει το μωρό;” Μια ερώτηση είναι γνωστή ως συνάρτηση στον προγραμματισμό υπολογιστών. Κάνοντας μια ερώτηση για να υπολογιστεί το αποτέλεσμα (η απάντηση) είναι γνωστό ως κλήση συνάρτησης.

Ο προγραμματισμός υπολογιστών είναι η χρήση όλων αυτών των ιδεών σε διάφορους συνδυασμούς. Αυτό που μπορεί να κάνει τα πράγματα δύσκολα είναι η πολυπλοκότητα. Η αλήθεια είναι ότι οι περισσότεροι υπολογιστές "αναγνωρίζουν" μόνο, περίπου 100 διαφορετικές οδηγίες. Τα εκατομμύρια προγράμματα που υπάρχουν στους υπολογιστές χρησιμοποιούν αυτές τις ίδιες 100 οδηγίες με διαφορετική σειρά και συνδυασμούς. Οπότε που υπάρχει η πολυπλοκότητα; Σκεφτείτε το ως εξής: Σε ένα παιχνίδι σκάκι, υπάρχουν μόνο έξι είδη πιονιών και κάθε κομμάτι κάνει μια απλή κίνηση. Το σκάκι είναι ένα σύνθετο παιχνίδι εξαιτίας όλων των πιθανών συνδυασμών κινήσεων.

Διαφορετικά, γράφοντας ένα πρόγραμμα είναι σαν να βάζετε σε ένα θεατρικό έργο 200 ηθοποιούς, 500 κοστούμια και 5 καμήλες μαζί σε μια συγκεκριμένη σκηνή. Τα πράγματα μπορούν να δυσκολέψουν μόνο με την επίβλεψη όλων αυτών των αντικειμένων. Αυτό το βιβλίο θα σας μάθει κάποια "κόλπα" για να διαχειριστείτε την πολυπλοκότητα και να σχεδιάσετε πως θα γράψετε τα προγράμματα πριν προσπαθήσετε να τα κάνουν να δουλέψουν. Στην πραγματικότητα, το να μάθετε πώς να βάζετε σε σειρά μια ακολουθία οδηγιών για να φέρετε σε πέρας μια διεργασία (πώς να σχεδιάσετε ένα πρόγραμμα) είναι το πολυτιμότερο μέρος στην εκμάθηση προγραμματισμού. Μπορεί να έχετε ακούσει τον όρο αντικειμενοστραφής προγραμματισμός. Αυτό το βιβλίο και το σύστημα Alice είναι βασισμένα στην χρήση αντικειμένων. Σε ένα πρόγραμμα γραμμένο με Alice, τα αντικείμενα είναι πράγματα τα οποία μπορείτε να δείτε.

Γιατί καλείται Alice;

Πρώτα απ' όλα, το Alice δεν είναι αρκτικόλεξο. Δεν είναι A.L.I.C.E. Η ομάδα ονόμασε το σύστημα Alice στην μνήμη του Charles Lutwidge Dodson, έναν άγγλο μαθηματικό που υπέγραφε με το όνομα Lewis Carroll. Ο Carroll έγραψε το έργο "Οι περιπέτειες της Alice στην χώρα των θαυμάτων". Όπως οι άνθρωποι που δημιούργησαν το Alice έτσι και ο Lewis Carroll ήταν ικανός να επιλύσει πολύπλοκα μαθηματικά, αλλά γνώριζε ότι το σημαντικότερο ήταν να κάνει τα πράγματα απλά και συναρπαστικά σε ένα μαθητή.

Με τον ίδιο τρόπο με τον οποίο η Alice ήταν διστακτική όταν αρχικά πέρασε μέσα από το καθρέφτη, με τον ίδιο τρόπο μπορεί να έχετε κάποιες αμφιβολίες σχετικά με την εκμάθηση ενός προγράμματος. Παρακαλώ προχωρήστε και σας υποσχόμαστε ότι η εκμάθηση του προγραμματισμού θα είναι ευκολότερη απ' ότι νομίζετε.

1-2 Αρχικά για το Alice

Ο προγραμματισμός με το σύστημα Alice σημαίνει ότι θα δημιουργήσετε πραγματικούς κόσμους με διάφορα αντικείμενα στον υπολογιστή σας. Μετά θα γράψετε προγράμματα για να κατευθύνετε τα δικά σας κινούμενα σχέδια. Θα ξεκινήσουμε με μια περίληψη του λογισμικού Alice και με μια επισκόπηση του περιβάλλοντος εργασίας για να σας βοηθήσουμε να ξεκινήσετε. Αυτό το κεφάλαιο συνεργάζεται με τις αρχικές ασκήσεις στο παράρτημα Α. Προτείνουμε να διαβάσετε αυτό το κεφάλαιο και παράλληλα να επιλύετε τις ασκήσεις σε ένα υπολογιστή.

Αρχικά: Ένας εικονικός κόσμος

Τα video games και οι προσομοιώσεις μπορούν να είναι είτε δυο, είτε τριών διαστάσεων (2D, 3D). Μπορεί να έχετε χρησιμοποιήσει ένα 2D προσομοιωτή γραφικών σε κάποιο μάθημα οδήγησης. Ένα μέρος της εκπαίδευσης των πιλότων είναι η χρήση προσομοιωτών πτήσης. Το πλεονέκτημα των προσομοιώσεων είναι εμφανές-όταν ένας αρχάριος πιλότος καταστρέψει ένα πολεμικό αεροπλάνο, ούτε ο πιλότος άλλα ούτε το αεροπλάνο δεν βρίσκεται σε κίνδυνο. Ένα video game ή μια προσομοίωση υλοποιημένα σε 3D καλείται εικονικός κόσμος. Χρησιμοποιώντας ένα εικονικό κόσμο, ο προσομοιωτής γίνεται ρεαλιστικότερος και αποτελεσματικότερος.

Για να καταλάβετε τη διαφορά μεταξύ 2D και 3D, συγκρίνετε τις εικόνες 1-2-1 και 1-2-2. Η εικόνα 1-2-1 δείχνει μια ταινία που έχει δημιουργηθεί με ένα 2D προσομοιωτή. Η δομή είναι δυο διαστάσεων γιατί η ταινία έχει μήκος και πλάτος αλλά δεν έχει βάθος. Η εικόνα 1-2-2 δείχνει πλάνα του λαγού και της χελώνας από μια μπροστινή και μια πισινή κάμερα. Ο λαγός και η χελώνα είναι αντικείμενα σε ένα 3D εικονικό κόσμο που έχει πλάτος, μήκος και βάθος, οπότε η κάμερα μπορεί να κινηματογραφήσει από διάφορες οπτικές γωνίες, πράγμα το οποίο δίνει μια αίσθηση πραγματικότητας στα αντικείμενα.

ΕΙΚΟΝΑ 1-2-1. 2D προσομοίωση μπροστά και πίσω όψη

ΕΙΚΟΝΑ 1-2-2. 3D κόσμος με μπροστά και πίσω όψη του λαγού και της χελώνας

Ένας πραγματικός κόσμος στο λογισμικό Alice ξεκινάει με μια φόρμα αρχικής σκηνής. Οι φόρμες επιλογής παρουσιάζονται στο αρχικό παράθυρο μόλις ξεκινήσει το Alice. Οι φόρμες φαίνονται στην εικόνα 1-2-3, όπου έχουμε επιλέξει μια αρχική σκηνή που αποτελείται από ένα γαλάζιο ουρανό και από καταπράσινο γρασίδι.

ΕΙΚΟΝΑ 1-2-3. Επιλέγοντας μια φόρμα για αρχική σκηνή

Αρχικά: Αντικείμενα και 3D μοντέλα

Ένα από τα ευχάριστα σημεία στην χρήση του Alice είναι η φαντασία που απαιτείται για την δημιουργία νέων κόσμων. Ξεκινάμε με μια απλή σκηνή και προσθέτουμε αντικείμενα. Στον κόσμο της εικόνας 1-2-2, τα αντικείμενα είναι ένα δέντρο, ένας φράχτης, μια χελώνα και ένας λαγός. Μερικά αντικείμενα αποτελούν το σταθερό σκηνικό (δέντρα, σπίτια, ουρανός και άλλα). Άλλα αντικείμενα (άνθρωποι, ζώα, διαστημόπλοια) παίζουν τον ρόλο των ηθοποιών στο σκηνικό σας (κινούνται τριγύρω, μιλάνε).

Για να διευκολυνθεί ο χρήστης στην δημιουργία ενός νέου κόσμου με διάφορα αντικείμενα το Alice παρέχει ένα μεγάλο αριθμό 3D μοντέλων. Ένα

3D μοντέλο είναι σαν ένα μηχανολογικό σχέδιο ενός σπιτιού. Το σχέδιο παρέχει ένα μοντέλο του σπιτιού το οποίο αναλύει την όψη του, την τοποθεσία και το μέγεθος των δωματίων και κάποιες πληροφορίες που θα ακολουθήσει ο εργολάβος για το χτίσιμο του σπιτιού. Παρόμοια ένα μοντέλο του Alice δίνει οδηγίες για το πώς θα δημιουργηθεί ένα νέο αντικείμενο στη σκηνή. Το 3D μοντέλο παρέχει πληροφορίες για το πώς θα σχεδιαστεί το αντικείμενο, τι χρώμα θα έχει, από τι κομμάτια θα αποτελείται, το μέγεθος του και πολλές άλλες λεπτομέρειες.

Με την εγκατάσταση του λογισμικού στον υπολογιστή σας, εγκαθίσταται και μια τοπική βιβλιοθήκη που περιέχει διάφορα έτοιμα 3D μοντέλα. Πρόσθετα μοντέλα μπορεί να βρεθούν στην ιστοσελίδα (http://www.alce.org) και στο cd που παρέχεται μαζί με το βιβλίο. Τα μοντέλα που παρέχονται στην βιβλιοθήκη είναι εύκολο να διαχειριστούν μέσω του διαχειριστή σκηνών, που φαίνεται στην εικόνα 1-2-4. Εάν έχετε το cd μέσα στο cd-rom, ένας επιπλέον φάκελος της βιβλιοθήκης του cd θα εμφανιστεί στον φάκελο. Τα παραδείγματα και οι ασκήσεις το βιβλίου χρησιμοποιούν μοντέλα και της τοπικής αλλά και της web βιβλιοθήκης. Εάν θέλετε να χρησιμοποιήσετε ένα 3D μοντέλο που δεν εμφανίζεται στην τοπική βιβλιοθήκη, μπορείτε να την βρείτε είτε στο cd, είτε στο web.

Η Alice δεν είναι ένα πρόγραμμα δημιουργίας 3D γραφικών. Γι αυτό το λόγο παρέχονται βιβλιοθήκες με έτοιμα μοντέλα. Παρόλα αυτά είναι απίθανο να καλυφθούν οι απαιτήσεις όλων των χρηστών. Έτσι μπορείτε να φτιάξετε τον δικό σας χαρακτήρα με την βοήθεια των εργαλείων hebuilder και shebuilder που βρίσκονται στο φάκελο των ανθρώπων της τοπικής βιβλιοθήκης. Λεπτομέρειες για την χρήση αυτών των εργαλείων βρίσκονται στο παράρτημα Β.

ΕΙΚΟΝΑ 1-2-4. Διαχειριστής σκηνών με τους φακέλους της τοπικής και της web βιβλιοθήκης

Αρχικά: Τρεις και έξι κατευθύνσεις

Τα αντικείμενα στο κόσμο του Alice είναι τριών διαστάσεων. Κάθε αντικείμενο έχει μήκος, πλάτος και βάθος όπως φαίνεται στην εικόνα 1-2-5. (Σε αυτό τον κόσμο, υπάρχει ένας αστροναύτης (από την βιβλιοθήκη των ανθρώπων) στην φόρμα του διαστήματος). Το ύψος μετριέται κατά μήκος μιας νοητής γραμμής που είναι κατακόρυφη από πάνω προς τα κάτω, το πλάτος κατά μήκος μιας νοητής γραμμής που είναι οριζόντια από αριστερά προς τα δεξιά και το βάθος κατά μήκος μιας νοητής γραμμής από μπρος προς τα πίσω.

ΕΙΚΟΝΑ 1-2-5. Τρεις διαστάσεις

Όσο αναφορά αυτές τις τρεις διαστάσεις το αντικείμενο γνωρίζει ποιος δρόμος είναι πάνω ή κάτω σε σχέση με τον εαυτό του. Επίσης το αντικείμενο γνωρίζει το αριστερά και το δεξιά, το μπρος και το πίσω όπως φαίνεται στην εικόνα 1-2-6. Αυτό ισοδυναμεί με έξι πιθανές κατευθύνσεις στις οποίες μπορεί να κινηθεί ένα αντικείμενο. Αυτό είναι, το αντικείμενο έχει έξι βαθμούς

ελευθερίας για να κινηθεί σε ένα κόσμο. Είναι σημαντικό να προσέξετε ότι οι κατευθύνσεις είναι δεξιά και αριστερά σε σχέση με τον αστροναύτη, όχι με την θέση της κάμερας. Καλούμε έξι βαθμούς ελευθερίας (πιθανές κατευθύνσεις κίνησης) τον προσανατολισμό του αντικειμένου. Όταν κάνετε κλικ με το ποντίκι σε ένα αντικείμενο, εμφανίζεται ένα κίτρινο περίβλημα, όπως φαίνεται στην εικόνα 1-2-6. Το κίτρινο περίβλημα αυτό τονίζει το αντικείμενο.

ΕΙΚΟΝΑ 1-2-6. Προσανατολισμός αντικειμένου: έξι βαθμοί ελευθερίας

Αρχικά: Το κέντρο ενός αντικειμένου

Κάθε αντικείμενο στο πρόγραμμα Alice έχει ένα μοναδικό κέντρο. Το κεντρικό αυτό σημείο δεν υπολογίζεται. Είναι ένα χαρακτηριστικό κάθε αντικειμένου το οποίο ορίζεται από τον γραφίστα όταν δημιουργείται το 3D μοντέλο. Συνήθως το κεντρικό σημείο ενός μοντέλου βρίσκεται στο κέντρο του κίτρινου περιβλήματος του-η κοντά στο κέντρο βάρους. Το κεντρικό σημείο παρέχει πληροφορία για τον άξονα περιστροφής. Έτσι μερικά αντικείμενα όπως ένας τροχός ή ένα πουλί θα περιστρέφεται γύρω από το κεντρικό σημείο του. Η εικόνα 1-2-7 δείχνει το κέντρο ενός αντικειμένου-πουλιού. Χρησιμοποιήσαμε μια προβολή του πουλιού ως πλαίσιο από γραμμές για να δείξουμε ότι το κέντρο του βρίσκεται μέσα στο σώμα του.

ΕΙΚΟΝΑ 1-2-7. Κέντρο στο κέντρο της μάζας

Δεν έχουν όλα τα αντικείμενα το κέντρο στο κέντρο της μάζας του. Αυτά τα οποία γενικότερα κάθονται ή στέκονται στο έδαφος ή σε ένα τραπέζι, έχουν το κέντρο τους στη βάση του κίτρινου περιβλήματός του. Για τα αντικείμενα-ανθρώπους, το κεντρικό σημείο βρίσκεται ανάμεσα στα πόδια τους, όπως φαίνεται στην εικόνα 1-2-8. Αυτό γίνεται γιατί τα πόδια των ανθρώπων βρίσκονται στο έδαφος και η απόσταση του ανθρώπου από το έδαφος είναι μηδέν μέτρα.

Άλλα είδη αντικειμένων που δεν έχουν το κέντρο τους στο κέντρο της μάζας τους είναι αυτά που “κρατούνται” όταν τα χρησιμοποιεί κανείς, για παράδειγμα ένα ρόπαλο του baseball. Το κέντρο ενός ροπάλου του baseball βρίσκεται εκεί που κρατείται, όπως φαίνεται στην εικόνα 1-2-9. Το κέντρο βρίσκεται στην λαβή έτσι ώστε όταν περιστρέφεται, θα κινείται γύρω από το σημείο.

Αρχικά: Απόσταση

Η απόσταση ενός αντικειμένου από ένα άλλο μετριέται από το κέντρο του. Για παράδειγμα, η απόσταση του πουλιού από το έδαφος στην εικόνα 1-2-10 μετριέται από το κεντρικό σημείο του πουλιού.

ΕΙΚΟΝΑ 1-2-8. Το κέντρο ενός αντικειμένου που στέκεται στο έδαφος

ΕΙΚΟΝΑ 1-2-9. Το κεντρικό σημείο ενός αντικειμένου που κρατείται

ΕΙΚΟΝΑ 1-2-10. Η απόσταση προς τα κάτω, προς το έδαφος μετριέται από το κέντρο

Αρχικά: Θέση

Το κέντρο ενός αντικειμένου είναι το σημείο που δείχνει την θέση του στον κόσμο. Το σύστημα Alice ορίζει αυτόματα το κέντρο του εδάφους ως το κέντρο του κόσμου. Στην εικόνα 1-2-11, τρεις άξονες συντεταγμένων τοποθετούνται στο κέντρο του εδάφους. Στην λίστα ιδιοτήτων του εδάφους (βρίσκεται στον κατάλογο λεπτομερειών, κάτω αριστερά στην εικόνα 1-2-11), μπορείτε να δείτε ότι το κέντρο του εδάφους βρίσκεται στο σημείο (0,0,0) των αξόνων συντεταγμένων.

Όπως το έδαφος έτσι και κάθε αντικείμενο τοποθετείται σε σχέση με το κέντρο του κόσμου. Το πουλί στην εικόνα 1-2-12 βρίσκεται στην θέση (-3,41, 1,59, 6,15). Αυτό σημαίνει ότι το κέντρο του πουλιού βρίσκεται 3,41 μέτρα αριστερά, 1,59 μέτρα πάνω και 6,15 μέτρα εμπρός από το κέντρο του κόσμου.

ΕΙΚΟΝΑ 1-2-11. Το κέντρο του εδάφους τοποθετείται στο κέντρο του κόσμου

ΕΙΚΟΝΑ 1-2-12. Η θέση του πουλιού σε σχέση με το κέντρο του κόσμου

Αρχικά: Κίνηση

Στο Alice θα δημιουργήσετε εικονικούς κόσμους και θα δώσετε ζωή κινώντας τα αντικείμενα στον κόσμο με τον ίδιο τρόπο που κινούνται τα αντικείμενα σε ένα εξομοιωτή πτήσης ή σε ένα videogame. Θα χρησιμοποιήσετε πολλές τεχνικές κίνησης ίδιες με αυτές που χρησιμοποιούνται στα κινούμενα σχέδια στα φιλμ της Disney και της Pixar. Η ζωή των αντικειμένων είναι μια οφθαλμαπάτη, ένα φανταστικό θέαμα. Για να δημιουργηθεί αυτή η οφθαλμαπάτη, ο παραγωγός του φιλμ συνεργάζεται με τον ηθοποιό για να δημιουργηθεί μια ακολουθία καλλιτεχνικών σκηνών (ζωγραφιές και εικόνες), όπου η κάθε μια έχει μια μικρή διαφορά από την προηγούμενη. Η σκηνή σχεδιάζεται με αντικείμενα και μετά ξανασχεδιάζεται με τα αντικείμενα σε μια ελαφρά αλλαγμένη θέση. Η σκηνή σχεδιάζεται ξανά και τα αντικείμενα κινούνται λίγο ακόμη, λίγο ακόμη, λίγο ακόμη, λίγο ακόμη κτλ. Η εικόνα 1-2-13 απεικονίζει μια ακολουθία σκηνών στο σύστημα Alice.

Στην παραγωγή κινουμένων σχεδίων, τα πλαίσια σκηνών αποτυπώνονται σε μια ακολουθία πάνω σε ένα καρούλι φιλμ ή φωτογραφίζονται από μια ψηφιακή κάμερα. Το φιλμ παίζεται σε ένα προτζέκτορα ή μια οθόνη, εμφανίζοντας γρήγορα πολλές εικόνες που υπάρχουν στην ακολουθία δημιουργώντας έτσι την ψευδαίσθηση της κίνησης.

Το λογισμικό Alice δημιουργεί ένα παρόμοιο εφέ στην οθόνη του υπολογιστή. Δεν χρειάζεται να ανησυχείτε αν δεν είστε τέλειος καλλιτέχνης. Το Alice δημιουργεί από μόνο του την ακολουθία των σκηνών. Εσείς απλά είστε ο σκηνοθέτης και δίνετε οδηγίες για το τι να κάνει το κάθε αντικείμενο. Το Alice δημιουργεί (αποδίδει) την κίνηση.

ΕΙΚΟΝΑ 1-2-13. Μια ακολουθία σκηνών που δημιουργεί την κίνηση

Ξεκίνημα με το Alice

Σας συνιστούμε να πειραματιστείτε με το Alice, όσο θα εξερευνούσατε το καινούριο σας κινητό τηλέφωνο. Το βγάζετε από το κουτί και δοκιμάζετε όλα τα χαρακτηριστικά του. Με τον ίδιο τρόπο μπορείτε να μάθετε πώς να χειρίζεστε το λογισμικό Alice.

Το παράρτημα Α σας παρέχει ένα αρχικό εγχειρίδιο χρήσης που αποτελείται από ασκήσεις με λεπτομερείς πληροφορίες για το πώς θα ξεκινήσετε ένα νέο κόσμο, που θα βρείτε τις βιβλιοθήκες για 3D μοντέλα, πως θα αλλάξετε το χρώμα του εδάφους, πως θα προσθέσετε αντικείμενα σε ένα κόσμο και πως θα τα τοποθετήσετε κατάλληλα στην σκηνή. Εάν δεν το έχετε κάνει, πηγαίνετε στις ασκήσεις τώρα!

Μυστικά και τεχνικές 1

Ειδικά εφέ: Κείμενα και δυδιάστατες γραφικές εικόνες

Παρακάτω θα αναλύσουμε όλες τις δυνατότητες του Alice μέσω παραδειγμάτων. Στο τέλος κάθε κεφαλαίου, στο τμήμα Μυστικά και Τεχνικές θα παρέχονται ευχάριστοι τρόποι για να δημιουργήσετε κόσμους και κινήσεις αντικειμένων με ειδικά εφέ. Αν και αυτά τα τμήματα του βιβλίου δεν είναι απαραίτητα στην εκμάθηση θεμελιωδών προγραμματιστικών αρχών, είναι σημαντικά γιατί μερικές από τις αυτές τις τεχνικές χρησιμοποιούνται στους κόσμους των παραδειγμάτων. Τα Μυστικά και οι Τεχνικές αποτελούν ένα οδηγό γι' αυτούς που θέλουν να μάθουν περισσότερα για την κίνηση των σχεδίων. Πρόσθετες λεπτομέρειες για την χρήση του Alice παρέχονται στο παράρτημα Β, όπου μπορείτε να μάθετε πώς να ψάχνετε στις βιβλιοθήκες και πώς να δημοσιεύετε ένα κόσμο στο διαδίκτυο. Τα τμήματα Μυστικά και Τεχνικές μαζί με το παράρτημα Α και Β μπορούν να θεωρηθούν ως ένα μικρό εγχειρίδιο χρήσης του Alice.

Ένα σημαντικό θέμα της κίνησης των αντικειμένων είναι η μετάδοση πληροφορίας στο άτομο που βλέπει την κίνηση (τον χρήστη). Το κείμενο, ο ήχος, και οι γραφικές εικόνες σας βοηθούν να επικοινωνήσετε. Τα επόμενα τμήματα σας βοηθούν να προσθέσετε κείμενο και γραφικές εικόνες στον κόσμο σας.

3D Κείμενα

Για να προσθέσετε ένα 3D κείμενο σε ένα κόσμο, κάντε κλικ στην επιλογή "Δημιουργία 3D κειμένου" στην τοπική βιβλιοθήκη, όπως φαίνεται στην εικόνα Τ-1-1.

ΕΙΚΟΝΑ Τ-1-1. Κείμενο 3D στην τοπική βιβλιοθήκη

Ένα πλαίσιο κειμένου εμφανίζεται, όπως στην εικόνα Τ-1-2. Εκεί πληκτρολογείτε ένα κείμενο σε όποια μορφή θέλετε (για παράδειγμα έντονα, γράμματα, πλάγια γραφή).

Όταν πατήσετε το OK, το λογισμικό προσθέτει ένα αντικείμενο κειμένου στον κόσμο και μια καταχώρηση του αντικειμένου στο δένδρο των αντικειμένων. Το όνομα του αντικειμένου είναι το ίδιο με το όνομα του κειμένου που πληκτρολογήσατε, όπως φαίνεται στην εικόνα Τ-1-3.

Το αντικείμενο μπορεί να τοποθετηθεί χρησιμοποιώντας το ποντίκι με τον ίδιο τρόπο που γίνεται για κάθε άλλο αντικείμενο. Για να τροποποιήσετε το κείμενο στο αντικείμενο, κάντε κλικ στο κείμενο στην λίστα ιδιοτήτων του καταλόγου λεπτομερειών. Τότε εισάγετε ένα νέο κείμενο στο πλαίσιο κειμένου, όπως φαίνεται στην εικόνα Τ-1-4.

Προσέξτε ότι τροποποιώντας το κείμενο, δεν αλλάζει το όνομα του αντικειμένου. Το όνομα είναι ίδιο όπως ήταν αρχικά, όπως φαίνεται στην εικόνα Τ-1-5.

ΕΙΚΟΝΑ Τ-1-2. Ένα πλαίσιο κειμένου

ΕΙΚΟΝΑ Τ-1-3. Το κείμενο-αντικείμενο προστίθεται στο σκηνικό και στο δέντρο αντικειμένων

ΕΙΚΟΝΑ Τ-1-4. Τροποποίηση του κειμένου

ΕΙΚΟΝΑ Τ-1-5. Το όνομα του αντικειμένου παραμένει αμετάβλητο

Γραφικές εικόνες (Billboards)

Παρόλο που το Alice είναι ένα 3D σύστημα, είναι πιθανό να δημιουργηθούν επίπεδες 2D εικόνες σε ένα σκηνικό. Οι επίπεδες 2D εικόνες μπορούν να δημιουργηθούν σε ένα οποιοδήποτε εργαλείο ζωγραφικής και να σωθούν σε GIF, JPG ή TIF. Για να προσθέσετε μια 2D εικόνα (το Alice την ονομάζει billboard) στον κόσμο σας, επιλέξτε από το μενού File το Make Billboard όπως φαίνεται στην εικόνα Τ-1-6. Στο πλαίσιο διαλόγου, επιλέξτε την αποθηκευμένη εικόνα και μετά κάντε κλικ στο κουμπί εισαγωγή.

ΕΙΚΟΝΑ Τ-1-6. Εισαγωγή ενός billboard

Το Alice θα προσθέσει την επίπεδη εικόνα στον κόσμο. Το billboard στην εικόνα Τ-1-7 υλοποιεί μια από τις χρήσεις των billboard – παρέχοντας

πληροφορίες στον χρήστη για το πώς να παίξει ένα παιχνίδι. Σε αυτό το παράδειγμα, η εικόνα παρέχει πληροφορίες για την κίνηση ενός αντικειμένου με χρήση του πληκτρολογίου. (Παραδείγματα με χρήση των πλήκτρων του πληκτρολογίου χρησιμοποιούνται στο κεφάλαιο 5).

ΕΙΚΟΝΑ Τ-1-7. Ένα billboard που δίνει πληροφορίες

Ασκήσεις

Οι ασκήσεις παρακάτω χρησιμοποιούν το λογισμικό Alice και επαληθεύουν ότι έχετε μάθει όλα τα παραπάνω. Ο σκοπός σε κάθε άσκηση είναι να δημιουργηθεί μια αρχική σκηνή ενός κόσμου. Το Alice θα σε προτρέπει περιοδικά να σώζετε τον κόσμο σας. (Πληροφορίες για το πώς θα σώζετε τον κόσμο σας παρέχονται στο παράρτημα Α στις αρχικές ασκήσεις).Ένα όνομα για τον κόσμο σας προτείνεται σύμφωνα με το όνομα της άσκησης. Για παράδειγμα, ο κόσμος στην άσκηση 1 μπορεί να ονομαστεί Νησί.α2w.

Τα αντικείμενα σε κάθε κόσμο δημιουργούνται από τα 3D μοντέλα στην βιβλιοθήκη της Alice. Τα περισσότερα μοντέλα βρίσκονται στην τοπική βιβλιοθήκη που εγκαθίσταται μαζί με το πρόγραμμα Alice. Εάν το μοντέλο δεν βρίσκεται στην τοπική βιβλιοθήκη, ψάξτε το μοντέλο στην βιβλιοθήκη του cd ή στην βιβλιοθήκη Web (www.alice.org).

1. Νησί

Δημιουργείστε ένα σκηνικό νησί. Ξεκινήστε επιλέγοντας μια φόρμα που να περιέχει θάλασσα. Εναλλακτικά επιλέξτε τον κόσμο με το γρασίδι και αλλάξτε το χρώμα σε μπλε. Προσθέστε ένα νησί (από την βιβλιοθήκη περιβάλλοντος). Χρησιμοποιήστε το διαχειριστή σκηνών για να τοποθετήσετε το νησί λίγο αριστερά από το κέντρο του κόσμου. Μετά τοποθετήστε ένα χρυσόψαρο στον κόσμο σας. Μπορεί να διαπιστώσετε ότι το χρυσόψαρο είναι αόρατο επειδή τοποθετήθηκε πίσω από το νησί ή απλά ότι βρίσκεται σε λάθος σημείο. Χρησιμοποιείστε το διαχειριστή σκηνών για να τοποθετήσετε το χρυσόψαρο να κολυμπάει μέσα στη θάλασσα στα δεξιά του νησιού. Χρησιμοποιήστε τις ρυθμίσεις της κάμερας για να βάλετε το νησί και το χρυσόψαρο μέσα στην εμβέλεια της κάμερας.

ΕΙΚΟΝΑ

2. Χειμώνας

Τοποθετήστε δυο χιονάνθρωπους (βιβλιοθήκη ανθρώπων) σε μια χιονισμένη σκηνή. Χρησιμοποιήστε μια σκηνή με χιόνι στον αρχικό κόσμο. Μετά, δημιουργήστε χιονάνθρωπους τον ένα πάνω στον άλλο κάθετα.

ΕΙΚΟΝΑ

3. Σύνολο χιονάνθρωπων

Φτιάξτε ένα τοίχο τεσσάρων χιονάνθρωπων τον ένα πάνω στον άλλο, βάζοντας τους οριζόντια. (Χρησιμοποιήστε μεθόδους, το ποντίκι και το κίτρινο πλαίσιο του αντικειμένου). Οι τέσσερις χιονάνθρωποι θα είναι κάπως έτσι:

ΕΙΚΟΝΑ

4. Πάρτι με τσάι

Για να τιμήσετε τον Lewis Caroll, δημιουργήστε ένα πάρτι με τσάι για την Alice και τον λευκό λαγό. Πέρα από την Alice και το λαγό, το πάρτι πρέπει να περιέχει ένα τραπέζι (τραπέζι δείπνου στον φάκελο με τα έπιπλα στο cd ή στην βιβλιοθήκη web) και τρεις καρέκλες (έπιπλα), μια τσαγιέρα, μια τοστιέρα και ένα πιάτο (κουζίνα). Χρησιμοποιήστε τις πληροφορίες των μεθόδων, το ποντίκι και το πλαίσιο του αντικειμένου για να τοποθετήσετε σωστά τα αντικείμενα όπως την τσαγιέρα και το δοχείο γάλατος στο τραπέζι. Η εικόνα παρακάτω δίνει ένα παράδειγμα.

ΕΙΚΟΝΑ

5. Στρατιώτες στο κατάστρωμα

Προσθέστε ένα μεταφορέα (μεταφορικά μέσα) και τέσσερις στρατιώτες (άνθρωποι) σε ένα νέο κόσμο. Στοιχείστε τους στρατιώτες για μια επίσημη τελετή-δυο στρατιώτες σε κάθε μια άκρη του καταστρώματος, όπως φαίνεται παρακάτω.

ΕΙΚΟΝΑ

Κάντε δεξί κλικ σε κάθε στρατιώτη και χρησιμοποιήστε τις μεθόδους (από το popup μενού) για να κινήσετε τα χέρια των στρατιωτών, για να χαιρετίσει ο ένας τον άλλο. Η, χρησιμοποιήστε το ποντίκι στο διαχειριστή σκηνών για να κινήσετε τα χέρια στην κατάλληλη θέση. (Χρησιμοποιήστε το “επιρροή των υπομερών του σώματος” για να μπορέσετε να κινήσετε μόνο τα χέρια τους). Ανυψώστε το αριστερό χέρι του κάθε στρατιώτη (σχεδόν 45 μοίρες γωνία σε ένα οριζόντιο πλάνο). Το αποτέλεσμα πρέπει να είναι μια σκηνή, όπου και οι τέσσερις στρατιώτες χαιρετούν. Αυτό δεν είναι μια κίνηση-το μόνο που προσπαθείτε να κάνετε είναι να δημιουργήσετε το σκηνικό.

Υπόδειξη: Εάν τσεκάρετε το “επιρροή των υπομερών του σώματος”, ώστε να επιτρέψετε στα υπομέρη ενός αντικειμένου να κινηθούν, θυμηθείτε να το ξετσεκάρετε πριν να χρησιμοποιήσετε το ποντίκι για κάποιο άλλο σκοπό.

Περίληψη

Στο τέλος του κάθε κεφαλαίου, θα παραθέτουμε μια περίληψη και μια λίστα σημαντικών αρχών. Ο σκοπός της περίληψης είναι η συγκέντρωση των

σημαντικών πληροφοριών και των ιδεών που παρουσιάζονται στο κεφάλαιο. Ο σκοπός της λίστας των σημαντικότερων αρχών είναι να σας παρέχει ένα γρήγορο οδηγό.

Σε αυτό το κεφάλαιο, ένα πρόγραμμα υπολογιστή παρουσιάστηκε ως μια ακολουθία πληροφοριών η οποία καθοδηγεί τον υπολογιστή τι να κάνει. Ένα πρόγραμμα υπολογιστή είναι επίσης ένας τρόπος να δώσετε σε ένα άλλον άνθρωπο να καταλάβει το τι θέλετε να κάνει ο υπολογιστής. Το σημαντικότερο θέμα του προγραμματισμού υπολογιστών είναι το να μάθετε να σκέφτεστε πως θα βάλετε σε σειρά μια ακολουθία από πληροφορίες για να εκτελέσετε μια διεργασία.

Το Alice είναι ένα λογισμικό 3D το οποίο μπορεί να χρησιμοποιηθεί για να μάθετε πώς να σχεδιάζετε και να γράφετε ένα πρόγραμμα. Η Alice σας επιτρέπει να δημιουργήσετε κινούμενα σχέδια σε ένα 3D εικονικό κόσμο. Τα αντικείμενα είναι τριών διαστάσεων, έχοντας βάθος, ύψος και βάθος. Κάθε αντικείμενο έχει ένα προσανατολισμό που παρέχει μια αίσθηση κίνησης. Αυτό σημαίνει ότι ένα αντικείμενο “γνωρίζει” ποιος δρόμος είναι πάνω, κάτω, αριστερά, δεξιά, μπροστά και πίσω σε σχέση με τον εαυτό τους.

Σημαντικά θέματα σε αυτό το κεφάλαιο

- Ένα πρόγραμμα υπολογιστή είναι μια ακολουθία πληροφοριών που επεξηγεί στον υπολογιστή τι να κάνει. Είναι επίσης μια ακολουθία πληροφοριών που επεξηγεί σε ένα άλλο άνθρωπο το τι θέλετε να κάνει ο υπολογιστής.
- Η εκμάθηση προγραμματισμού είναι ουσιαστικά το να μάθετε να σκέφτεστε πώς να οργανώνετε μια ακολουθία πληροφοριών για να εκτελέσετε μια διεργασία.
- Στο Alice, η κίνηση των 3D αντικειμένων γίνεται σε ένα εικονικό κόσμο.
- Το Alice παρέχει ένα μεγάλο αριθμό 3D μοντέλων. Τα 3D μοντέλα είναι διαθέσιμα στο cd που συνοδεύει αυτό το βιβλίο, όπως επίσης και στο site του Alice στο http://www.alice.org.
- Ένα αντικείμενο στο Alice έχει έξι βαθμούς ελευθερίας για να κινηθεί τριγύρω σε ένα εικονικό κόσμο. Καλούμε τους έξι βαθμούς ελευθερίας (πιθανές κατευθύνσεις κίνησης) προσανατολισμό του αντικειμένου.
- Ένα αντικείμενο στο Alice έχει ένα μοναδικό κέντρο που ορίζεται από ένα γραφίστα όταν δημιουργείται το 3D αντικείμενο. Το κέντρο του εδάφους σε ένα κόσμο του Alice βρίσκεται στην θέση (0,0,0).

Κεφάλαιο 2 Σχεδιασμός και υλοποίηση προγράμματος

Σε αυτό το κεφάλαιο ξεκινάμε με μια εισαγωγή στον προγραμματισμό. Ένα πρόγραμμα είναι ένα σύνολο πληροφοριών που επεξηγεί στον υπολογιστή τι να κάνει. Κάθε πληροφορία είναι μια πράξη που πρέπει να εκτελεστεί. Το να γράψετε ένα πρόγραμμα για κίνηση 3D αντικειμένων σε ένα εικονικό κόσμο σχετίζεται με τα αντικείμενα και τις πράξεις που μπορούν να εκτελέσουν. Από μια πλευρά το γράψιμο ενός προγράμματος είναι σαν να λύνετε προβλήματα στα μαθηματικά. Πρώτα διαβάζουμε το πρόβλημα (μια περιγραφή της κατάστασης) και αποφασίζουμε πως θα το λύσουμε (τι βήματα θα ακολουθήσουμε). Μετά επιλύουμε το πρόβλημα / γράφουμε μια λύση και τεστάρουμε την απάντηση για να δούμε αν είναι σωστή. Παρόμοια, στο γράψιμο ενός προγράμματος που περιέχει κίνηση αντικειμένων, πρώτα διαβάζουμε το σενάριο (μια περιγραφή ιστορίας, παιχνιδιού-συχνά καλείται δήλωση προβλήματος) και αποφασίζουμε πως θα δώσουμε στα αντικείμενα ζωή (σχεδιασμός μιας σειράς σκηνών). Μετά γράφουμε τον κώδικα του προγράμματος (υλοποίηση) και το τεστάρουμε τρέχοντας το πρόγραμμα μας.

Πρέπει να πείτε ακριβώς τι εννοείτε όταν γράφεται ένα πρόγραμμα. Ο καλύτερος τρόπος να γράψετε ένα πρόγραμμα είναι να ξεκινήσετε διαβάζοντας ένα σενάριο / περιγραφή ιστορίας, παιχνιδιού και μετά να σχεδιάσετε μια λίστα ενεργειών για το πρόγραμμα.

Το τμήμα 2-1 ξεκινάει με σενάρια και σειρές σκηνών σαν μια μεθοδολογία για τον σχεδιασμό προγραμμάτων. Οι οπτικές σειρές σκηνών επιλέχθηκαν επειδή αποτελούν το σχεδιαστικό εργαλείο που χρησιμοποιείται από επαγγελματίες κατασκευαστές κινουμένων σχεδίων. Οι σειρές σκηνών που βασίζονται σε κείμενο, επιλέχθηκαν επειδή παρέχουν μια αλγοριθμική δομή (βήμα προς βήμα). Οι γραμμές του κειμένου σε μια σειρά σκηνών που βασίζονται σε κείμενο είναι παρόμοιες με τον ψευδοκώδικα-μια απλή έκδοση των οδηγιών που θα γίνουν τελικά κώδικας προγράμματος.

Το τμήμα 2-2 παρουσιάζει τα βασικά για την δημιουργία ενός απλού προγράμματος στο Alice. Η ιδέα είναι να χρησιμοποιηθεί μια σειρά σκηνών σαν οδηγός για το γράψιμο ενός προγράμματος σε ένα συντάκτη του Alice που βασίζεται στο ποντίκι. Μπορούμε να εστιάσουμε σε μια βήμα προς βήμα λύση επειδή το Alice θα φροντίσει αυτόματα για όλες τις λεπτομέρειες του “συντακτικού” (δομή και στίξη των εντολών). Σε μια κίνηση, μερικές ενέργειες πρέπει να γίνουν διαδοχικά και μερικές ταυτοχρόνως. Αυτό σημαίνει ότι ο κώδικας του προγράμματος πρέπει να είναι δομημένος ώστε να λέει στο λογισμικό ποιες ενέργειες να γίνουν διαδοχικά και ποιες να γίνουν ταυτόχρονα.

2-1 Σενάρια και σκηνές έργου

Η δημιουργία ενός προγράμματος που κινεί τα αντικείμενα σε ένα οπτικό κόσμο είναι μια διαδικασία τεσσάρων βημάτων: Διαβάστε το σενάριο (περιγραφή ενός προβλήματος ή διεργασίας), σχεδιασμός (σχέδιο), υλοποίηση (γράψτε το πρόγραμμα) και τεστ (δείτε αν λειτουργεί). Αυτό το τμήμα παρουσιάζει τα πρώτα δύο βήματα.

Το διάβασμα του σεναρίου και ο σχεδιασμός του πλάνου δράσης είναι σημαντικά βήματα στην κατασκευή ενός προγράμματος. Ένα σχέδιο είναι ένα "πλάνο" στρατηγικής. Τα προγράμματα που παρουσιάζονται στα πρώτα κεφάλαια του βιβλίου είναι αδρά περιγραφόμενα, γιατί πιστεύουμε ότι είναι φρόνιμο να ξεκινήσετε από νωρίς να σχεδιάζετε όμορφα σχέδια. Μετέπειτα όταν τα προγράμματα δυσκολεύουν, ο χρόνος που θα αφιερωθεί για τον σχεδιασμό προγραμμάτων θα είναι πολύ περισσότερος.

Διαβάστε το σενάριο

Πριν να συζητήσουμε πώς να δημιουργήσουμε ένα σχέδιο, πρέπει να γνωρίζουμε τι είδους πρόβλημα πρόκειται να λυθεί ή τι είδους διεργασία πρέπει να εκτελεστεί. Ένα σενάριο είναι μια δήλωση προβλήματος (ή διεργασίας) η οποία περιγράφει την συνολική κίνηση σε σχέση με το τι πρόβλημα πρέπει να λυθεί ή τι μάθημα πρέπει να διδαχθεί. (Πολλοί επιστήμονες πληροφορικής χρησιμοποιούν τον όρο προδιαγραφές απαιτήσεων. Στο Alice, ο όρος σενάριο είναι ευκολότερο να σχετισθεί με την σκηνή του κόσμου, τα αντικείμενα και τις πράξεις). Τα κινούμενα σχέδια και τα φιλμ ξεκινούν με ένα σενάριο που δημιουργείται από επαγγελματίες συγγραφείς, που καλείται "ιστορία". Όπως χρησιμοποιείται εδώ, σε σχέση με το αρχικό νόημα, μια ιστορία μπορεί να είναι η διδασκαλία ενός μαθήματος, ένα παιχνίδι ή ένα κινούμενο σχέδιο.

Σε ένα κόσμο του Alice, ένα σενάριο παρέχει όλες τις απαραίτητες λεπτομέρειες για το στήσιμο του αρχικού σκηνικού και μετά ο σχεδιασμός μιας ακολουθίας πληροφοριών για την επίτευξη της κίνησης. Αυτό είναι, ένα σενάριο παρέχει απαντήσεις στις παρακάτω ερωτήσεις:

1. Τι ιστορία έχει ειπωθεί;
2. Τι αντικείμενα χρειάζονται; Μερικά αντικείμενα θα παίξουν ηγετικό ρόλο στην ιστορία και μερικά άλλα θα χρησιμοποιηθούν απλά για το πίσω μέρος του σκηνικού.
3. Τι ενέργειες πρέπει να γίνουν; Οι ενέργειες στην ιστορία θα αποτελέσουν τελικά τις πληροφορίες του προγράμματος.

Παράδειγμα σεναρίου

Ας θεωρήσουμε ένα σενάριο: Μετά από ένα ταξίδι στο διάστημα, ένα διαστημόπλοιο επανδρωμένο με ρομπότ έχει προσγειωθεί στο φεγγάρι. Το ρομπότ έχει βγει από το διαστημόπλοιο και έχει στήσει μια κάμερα έτσι ώστε οι επιστήμονες που βρίσκονται στο κέντρο της NASA στο Houston να δουν το ιστορικό γεγονός. Στο σκηνικό μας μπορούμε να δούμε το ρομπότ, το διαστημόπλοιο, το σεληνιακό έδαφος και μερικούς βράχους. Ξαφνικά ένας εξωγήινος κρυφοκοιτάζει πίσω από ένα βράχο το ρομπότ. Το ρομπότ εκπλήσσεται και γυρνάει το κεφάλι του ακριβώς προς τον εξωγήινο. Το ρομπότ περπατάει προς τον εξωγήινο για να ρίξει μια πιο κοντινή ματιά, με αποτέλεσμα ο εξωγήινος να κρυφτεί πίσω από τους βράχους. Τελικά το ρομπότ κοιτάζει στην κάμερα, κάνει σήμα κινδύνου, και λέει "Houston έχουμε πρόβλημα!"

Από αυτό το σενάριο έχουμε απαντήσεις στις ερωτήσεις:

- Τι ιστορία έχει ειπωθεί; Αυτό το σενάριο περιγράφει μια χιουμοριστική ιστορία για μια συνάντηση ενός ρομπότ και ενός εξωγήινου στο φεγγάρι.
- Τι αντικείμενα χρησιμοποιούνται; Τα αντικείμενα είναι το ρομπότ, το διαστημόπλοιο και ο εξωγήινος. Το σκηνικό είναι η επιφάνεια του φεγγαριού σε ένα διαστημικό κόσμο.
- Τι ενέργειες θα γίνουν; Οι ενέργειες είναι το κρυφοκοίταγμα πίσω από το βράχο, το γύρισμα του κεφαλιού του ρομπότ και η κίνηση του προς τον εξωγήινο, το κρύψιμο του εξωγήινου πίσω από το βράχο και η αποστολή μηνύματος από το ρομπότ στην γη.

Σχεδιασμός

Μια διάταξη σκηνών είναι η σχεδιαστική προσέγγιση που θα χρησιμοποιήσουμε για να επιλύσουμε το πρόβλημα ή για να σχεδιάσουμε μια λίστα ενεργειών ώστε να εκτελεστεί η διεργασία που καθορίζεται στο σενάριο. Στην Disney και στην Pixar και σε άλλα μεγάλα στούντιο που κάνουν animation, οι εργαζόμενοι αποσυνθέτουν ένα μεγάλο και μακρύ σενάριο σε ακολουθίες πολλών μικρών σεναρίων. Για κάθε σενάριο, δημιουργείται μια διάταξη σκηνών ώστε να αναπαρασταθεί η ακολουθία των σκηνών. Η διάταξη των σκηνών μπορεί να αποτελείται από δεκάδες σκίτσα, που ζωγραφίζονται από καλλιτέχνες ή δημιουργούνται από ανιματερ υπολογιστών που χρησιμοποιούν ειδικό λογισμικό. Στην εικόνα 2-1-1 παρουσιάζονται σκίτσα διαφόρων σκηνών από το “παιχνίδι του Geri”, ένα μικρό φιλμ από την Pixar που έχει γραφτεί από τον Jan Pinkava. Το φιλμ κέρδισε όσκαρ για την καλύτερη ταινία με animation. Σε αυτό το φιλμ, ο χαρακτήρας παίζει σκάκι με τον εαυτό του.

Η προσέγγιση της αποσύνθεσης ενός προβλήματος ή μιας διεργασίας σε υποπροβλήματα ή υποδιεργασίες δεν χρησιμοποιείται μόνο από τους προγραμματιστές υπολογιστών και τους ανιματέρ. Οι συγγραφείς θεατρικών έργων, διασπούν τα έργα τους σε ξεχωριστές πράξεις και τις πράξεις σε ξεχωριστές σκηνές! Οι μηχανικοί διαλύουν τα πολύπλοκα συστήματα (π.χ. αεροπλάνα) ή τις συσκευές (π.χ. μικροκυκλώματα) στα κομμάτια που τα συνθέτουν για να κάνουν το πρόβλημα πιο διαχειρίσιμο.

ΕΙΚΟΝΑ 2-1-1. Διατάξεις σκηνών του έργου “το παιχνίδι του Geri”, Pixar

Οπτικές διατάξεις σκηνών έργου

Μια οπτική διάταξη σκηνών “σπάει” ένα σενάριο σε μια ακολουθία σκηνών με μετάβαση μεταξύ των σκηνών. Κάθε σκίτσο αναπαριστά ένα στιγμιότυπο μιας σκηνής (κατάσταση) κατά την διάρκεια της κίνησης. Κάθε στιγμιότυπο σχετίζεται με αντικείμενα σε συγκεκριμένες θέσεις, χρώμα, μέγεθος και πόζα. Όταν συμβαίνουν μια ή περισσότερες μεταβολές στο κινούμενο σχέδιο, οδηγούμαστε στην επόμενη σκηνή.

Τα στιγμιότυπα αριθμίζονται σε μια ακολουθία και παρέχουν κατάλληλες πληροφορίες. Για μικρού μήκους κινούμενα σχέδια, η ταξινόμηση μπορεί να αναπαρασταθεί σε μια μεγάλη σελίδα φύλλου. Για πιο πολύπλοκα σχέδια, ένα ξεχωριστό φύλλο ζωγραφικής μπορεί να χρησιμοποιηθεί για κάθε σκηνή, επιτρέποντας στον σχεδιαστή να παραλείψει ή να αλλάξει την σειρά των σκηνών χωρίς να ξεκινήσει από την αρχή.

Για να δημιουργήσουμε μια οπτική διάταξη σκηνών δανειζόμαστε μια τεχνική επαγγελματιών ανιματερ-μια ακολουθία από ζωγραφισμένες στο χέρι σκηνές. Μια φόρμα διατάξεων σκηνών φαίνεται στην εικόνα 2-1-2. Κάθε στιγμιότυπο παρέχει πληροφορίες για τον αριθμό σκηνής και περιέχει ένα σκίτσο ή μια εικόνα που δείχνει που βρίσκονται τα αντικείμενα στη σκηνή. Η περιγραφή εξηγεί τι ενέργεια γίνεται. Εάν στα κινούμενα σχέδια είναι κατάλληλος ένας ήχος, η περιγραφή θα περιέχει μια λίστα ήχων που θα παίζει κατά τη διάρκεια της σκηνής. Επίσης μπορεί να χρησιμοποιηθεί και κείμενο. Ο ήχος και το κείμενο χρησιμοποιούνται μόνο εάν χρειάζεται.

ΕΙΚΟΝΑ 2-1-2. Φόρμα διατάξεων σκηνών

Για εμάς εδώ δεν θα δώσουμε βάση σε σκίτσα άκρως καλλιτεχνικά. Απλοί κύκλοι, τετράγωνα και γραμμές μπορούν να χρησιμοποιηθούν για να αναπαραστήσουν αντικείμενα που εμφανίζονται στην σκηνή. Εάν είναι απαραίτητο, τα σχήματα μπορούν να ονομαστούν με το όνομα ή τον κωδικό του χρώματος του αντικειμένου.

Για να υλοποιήσετε μια διάταξη σκηνών, θα χρησιμοποιηθεί το σενάριο με το ρομπότ. Η εικόνα 2-1-3 δείχνει μια απλή σκηνή όπου ο εξωγήινος κρυφοκοιτάζει πίσω από τον βράχο.

ΕΙΚΟΝΑ 2-1-3. Σκηνές με σκίτσα ζωγραφισμένα με το χέρι

Απλά σκίτσα έχουν χρησιμοποιηθεί για να δημιουργηθεί το διαστημόπλοιο, το ρομπότ και ο εξωγήινος. Καφέ γραμμές έχουν χρησιμοποιηθεί για να σχεδιαστούν οι βράχοι μπροστά από τον εξωγήινο. Οι γκρι γραμμές αναπαριστούν την επιφάνεια του φεγγαριού. Χρησιμοποιώντας απλές φιγούρες, οι σκηνές είναι εύκολο να δημιουργηθούν.

Για τις εικόνες αυτού το βιβλίου, χρησιμοποιούμε τον συντάκτη σκηνών του Alice για να προσθέσουμε αντικείμενα σε ένα κόσμο και για να κανονίσουμε τα αντικείμενα σε διάφορες πόζες. Όταν δημιουργηθεί μια επιτυχημένη σκηνή, φωτογραφίζεται και αντιγράφεται σε ένα έγγραφο. Η εικόνα 2-1-4 παρουσιάζει μια σειρά εικόνων για την αρχή του σκηνικού του σεναρίου με το ρομπότ. (Χρησιμοποιήσαμε το spiderRobot και το alienOnWheels από τον SciFi φάκελο της βιβλιοθήκης). Οι φωτογραφίες της οθόνης για μια διάταξη σκηνών είναι ομορφότερες από τα απλά σκίτσα που ζωγραφίζονται στο χέρι αλλά χρειάζονται περισσότερο χρόνο για να δημιουργηθούν και να οργανωθούν.

ΕΙΚΟΝΑ 2-1-4. Φωτογραφίες οθόνης διάταξης σκηνών

Σειρές σκηνών που βασίζονται σε κείμενο

Ενώ οι επαγγελματίες ανιματέρ χρησιμοποιούν διατάξεις σκηνών με εικόνες, δεν έχουν όλοι την υπομονή να φτιάξουν δεκάδες σκίτσα. Μια σκηνή που βασίζεται σε κείμενο είναι μια εναλλακτική λύση. Είναι σαν μια "λίστα υποχρεώσεων " και μας επιτρέπει να σχεδιάσουμε μια δομή για τον κώδικα του προγράμματος. Για να επωφεληθούμε από τα πλεονεκτήματα που προσφέρουν η εικόνα και το κείμενο και οι δύο μέθοδοι δημιουργίας σκηνών θα χρησιμοποιηθούν σε αυτό το βιβλίο.

Παράδειγμα σειρών σκηνών που βασίζονται σε κείμενο

Η σειρά σκηνών που βασίζεται σε κείμενο για το παράδειγμα του ρομπότ φαίνεται παρακάτω. Παρατηρείστε ότι μια σειρά σκηνών που βασίζεται σε κείμενο μπορεί να συνοψίζει πολλές σκηνές από μια διάταξη σκηνών με εικόνες. Για παράδειγμα η σειρά των σκηνών που φαίνεται παρακάτω συνοψίζει τις σκηνές 1,2 και 3 της οπτικής διάταξης σκηνών της εικόνας 2-1-4 και αναπαριστά μόνο μερικές ενέργειες. Θα ολοκληρωθεί στο επόμενο τμήμα.

Do the following steps in order
- alien moves up
- alien says "Slithy toves?"
- robot's head turns around
- robot turns to look at alien
- *Do the following steps together*
  - robot moves toward the alien
  - robot legs walk
  - etc.

Κάνε τα ακόλουθα βήματα στην σειρά
- ο εξωγήινος κινείται
- ο εξωγήινος λέει "σύντροφος;"
- το κεφάλι του ρομπότ γυρίζει
- το ρομπότ γυρίζει και κοιτάει τον εξωγήινο
- *Κάνε τα ακόλουθα βήματα παράλληλα*
  - το ρομπότ κινείται προς τον εξωγήινο
  - τα πόδια του ρομπότ περπατούν
  - κτλ

Οι γραμμές του κειμένου σε μια σειρά σκηνών που βασίζεται σε κείμενο παρέχουν μια διατεταγμένη λίστα πράξεων. Οι γραμμές γράφονται σε ένα περιγραφικό σχήμα και οι εσοχές κάνουν τις σκηνές εύκολες στο διάβασμα.

Παρατηρήστε ότι δυο γραμμές είναι σε πλάγια γραφή. Αυτές οι γραμμές οργανώνουν τις ενέργειες – μερικές ενέργειες πρέπει να γίνουν στην σειρά (μια κάθε φορά) και άλλες πρέπει να γίνουν παράλληλα (την ίδια στιγμή). Οι τέσσερις πρώτες πράξεις γίνονται στη σειρά (ο εξωγήινος κινείται, μιλάει, το κεφάλι του ρομπότ κινείται, το ρομπότ κινείται και κοιτάζει προς τον εξωγήινο). Η επόμενη πράξη, όπου το ρομπότ κινείται προς τον εξωγήινο για να ρίξει μια πιο κοντινή ματιά, είναι στην πραγματικότητα μια σύνθεση πράξεων που γίνονται ταυτόχρονα (το ρομπότ κινείται μπροστά την ίδια στιγμή που τα πόδια του ρομπότ προσομοιώνουν ένα περπάτημα).

Στην τεχνολογία των υπολογιστών, μια σειρά σκηνών που βασίζονται σε κείμενο ονομάζεται αλγόριθμος-μια λίστα ενεργειών που εκτελούν μια διεργασία ή λύνουν ένα πρόβλημα. Οι ενέργειες σε μιας σειρά σκηνών που βασίζονται σε κείμενο μοιάζουν πολύ με τον κώδικα ενός προγράμματος και έτσι συχνά καλούνται ψευδοκώδικας.

Αποτιμήστε και επανεξετάστε

Εφόσον σχεδιαστεί μια διάταξη σκηνών, είναι καλή ιδέα να ρίξετε μια αντικειμενική ματιά για να αποφασίσετε αν θα αλλαχθεί κάτι. Αποτιμήστε τις σκηνές απαντώντας τις παρακάτω ερωτήσεις:

- Η πράξη “ρέει” από σκηνή σε σκηνή, όσο η ιστορία εκτυλίσσεται;
- Χρειάζονται να γίνουν αλλαγές για να ταιριάξει κάποια σκηνή με την επόμενη;
- Μήπως παραβλέψατε κάποιο σημαντικό μέρος της ιστορίας;
- Υπάρχει κάτι που θα έπρεπε να αλλάξει στην ιστορία;

Σημαντική ιδέα είναι ότι η διάταξη σκηνών δεν είναι η τελική. Πρέπει να είμαστε πρόθυμοι να ανασκοπήσουμε τα σχέδια μας και να τα τροποποιήσουμε. Κατά τον σχεδιασμό ενός προγράμματος, ακολουθούμε την ίδια τακτική με αυτή που ακολουθεί ένας ζωγράφος που έχει μια ιδέα για ένα πίνακα. Ο ζωγράφος συχνά κάνει ένα αρχικό σκίτσο για να σχεδιάσει το πώς θα μοιάζει ο πίνακας. Αυτός είναι ένας τρόπος που μετασχηματίζει την συνοπτική ιδέα σε ένα περισσότερο πραγματικό θέαμα. Μετά ο ζωγράφος κοιτάζει ξανά την αρχική έκδοση και μπορεί να την αλλάξει αρκετές φορές πριν να βάλει χρώματα στον καμβά. Έτσι και ένας μηχανικός που σχεδιάζει μια γέφυρα ή ένα αεροπλάνο (ή κάτι άλλο) δοκιμάζει πολλές διαφοροποιημένες φάσεις σχεδίων μέχρι να παραχθεί το τελικό προϊόν. Όλοι οι δημιουργικοί άνθρωποι περνούν από αυτούς τους κύκλους αλλαγής σχεδίασης του έργου.

2-2 Ένα πρώτο πρόγραμμα

Στο τμήμα 2-1, μάθατε πώς να διαβάζετε ένα σενάριο και να σχεδιάζετε ένα κινούμενο σχέδιο για να εκτελέσετε μια διεργασία ή να παίξετε ένα παιχνίδι. Τώρα είστε έτοιμοι να μάθετε πως μπορεί να γραφεί ένα πρόγραμμα με κινούμενα σχέδια. Αυτό το βήμα ονομάζεται υλοποίηση. Σας συνιστούμε να

διαβάσετε αυτό το τμήμα όσο κάθεστε μπροστά σε ένα υπολογιστή. Ξεκινήστε το Alice και επαναλάβετε τα βήματα που παρουσιάζονται στα παραδείγματα.

Τι είναι ένα πρόγραμμα;

Όπως ξέρετε, ένα πρόγραμμα είναι μια λίστα εντολών (ενεργειών) για να επιτευχθεί μια διεργασία. Μπορείτε να φανταστείτε ένα πρόγραμμα στο Alice όπως ένα σενάριο σε ένα θεατρικό έργο. Ένα θεατρικό κείμενο διηγείται μια ιστορία περιγράφοντας τις πράξεις που θα γίνουν και τα λόγια που θα εκφωνήσουν οι ηθοποιοί στην σκηνή. Παρόμοια ένα πρόγραμμα στο Alice καθορίζει τις πράξεις που θα γίνουν, τους ήχους και τα κείμενα που θα χρησιμοποιηθούν από τα αντικείμενα σε ένα εικονικό κόσμο.

Δημιουργία μιας αρχικής στιγμής

Ένα αρχαίο γνωμικό λέει ότι “Το μακρύ ταξίδι ξεκινάει με ένα απλό βήμα”. Ας ξεκινήσουμε το ταξίδι μας υλοποιώντας την πρώτη κίνηση του ρομπότ που περιγράφεται στην ενότητα 2-1. Θυμηθείτε ότι το επανδρωμένο με ρομπότ διαστημόπλοιο μόλις έχει φτάσει στο φεγγάρι. Το ρομπότ μόλις συνάντησε απρόσμενα ένα εξωγήινο που εμφανίστηκε πίσω από τους βράχους. Το ρομπότ κινείται προς τον εξωγήινο για να ελέγξει και μετά στέλνει μήνυμα στη γη: “Houston, έχουμε πρόβλημα!”

Το πρώτο βήμα στην υλοποίηση του προγράμματος κινουμένων σχεδίων είναι να δημιουργηθεί μια αρχική σκηνή. Επιλέγεται ένα σκηνικό διαστήματος και μετά ένα ρομπότ, ένας εξωγήινος και ένα διαστημόπλοιο (από τον φάκελο SciFi της βιβλιοθήκης) και προστίθενται στον κόσμο. Βράχια (από τον φάκελο της φύσης στο cd ή στην web βιβλιοθήκη) προστίθενται και τοποθετούνται μπροστά από τον εξωγήινο για να τον κρύψουν. Η αρχική σκηνή φαίνεται στην εικόνα 2-2-1.

ΕΙΚΟΝΑ 2-2-1. Αρχική σκηνή

Κόσμοι στο cd

Το cd που συνοδεύει αυτό το βιβλίο περιέχει όλους τους κόσμους των παραδειγμάτων. Οι κόσμοι έχουν όλα τα αντικείμενα τοποθετημένα στην αρχική σκηνή. Οι κόσμοι του cd δεν έχουν τον κώδικα του προγράμματος-ο κώδικας παρέχεται σε κάθε κεφάλαιο. Σας προτείνουμε να καθίσετε σε ένα υπολογιστή, να βάλετε το cd, να φορτώσετε τον κόσμο και να αναδομήσετε το πρόγραμμα όσο διαβάζετε το κεφάλαιο. Αυτό θα σας βοηθήσει να μάθετε πώς να γράφετε προγράμματα και επίσης θα σας βοηθήσει να φτιάξετε τα δικά σας κινούμενα σχέδια.

Συντάκτης κώδικα προγράμματος

Όταν δημιουργηθεί η αρχική σκηνή, πρέπει να γραφούν οι πληροφορίες που φτιάχνουν τον κώδικα. Το Alice παρέχει ένα συντάκτη κώδικα-το μεγάλο κίτρινο πλαίσιο στην κάτω δεξιά πλευρά του κύριου παραθύρου του Alice, όπως φαίνεται στην εικόνα 2-2-2. Οι εντολές για ένα πρόγραμμα εισάγονται

στον συντάκτη. (Από δω και στο εξής, θα αναφερόμαστε στον συντάκτη κώδικα προγράμματος με τον όρο “συντάκτης”).

ΕΙΚΟΝΑ 2-2-2. Συντάκτης κώδικα προγράμματος (μεγάλο, κίτρινο πλαίσιο)

World.my first method (η πρώτη μέθοδος του κόσμου)

Όπως φαίνεται στην εικόνα 2-2-2, η ετικέτα της περιοχής σύνταξης είναι *World.my first method.* Μια μέθοδος είναι ένα τμήμα ενός κώδικα (ένα μικρό σύνολο πληροφοριών) που ορίζει πώς να εκτελεστεί μια συγκεκριμένη διεργασία. Το Alice αυτόματα χρησιμοποιεί το όνομα *World.my first method* για το πρώτη σύνταξη. Στην πραγματικότητα μπορεί να χρησιμοποιηθεί οποιοδήποτε όνομα για μια μέθοδο. Θα χρησιμοποιήσουμε το όνομα *World.my first method* γι’ αυτό το παράδειγμα. Το σενάριο με το ρομπότ είναι εύκολο αρκετά και μπορεί να προγραμματιστεί μόνο με μια μέθοδο, την *World.my first method .*Όταν πατηθεί το κουμπί Play, το Alice θα εκτελέσει την *World.my first method* εκτελώντας τις οδηγίες που γράφουμε εκεί.

Τι πληροφορίες χρειάζονται;

Ας ρίξουμε μια ματιά στην προηγούμενη διάταξη σκηνών.

Κάνε τα ακόλουθα βήματα στην σειρά
ο εξωγήινος κινείται
ο εξωγήινος λέει “σύντροφος;”
το κεφάλι του ρομπότ γυρίζει
το ρομπότ γυρίζει και κοιτάει τον εξωγήινο
Κάνε τα ακόλουθα βήματα παράλληλα
το ρομπότ κινείται προς τον εξωγήινο
τα πόδια του ρομπότ περπατούν
κτλ

Πραγματικά, αυτή η διάταξη σκηνών είναι ελλιπής επειδή δεν ολοκληρώσαμε την ιστορία. Το σενάριο περιέγραψε μια ακολουθία ενεργειών: (α) Ο εξωγήινος εμφανίζεται πίσω από τους βράχους, (β) ο εξωγήινος λέει “σύντροφος;” (γ) το κεφάλι του ρομπότ γυρίζει, (δ) το ρομπότ γυρίζει και κοιτάει τον εξωγήινο, (ε) το ρομπότ κινείται προς τον εξωγήινο για να ρίξει μια κοντινή ματιά, (στ) ο εξωγήινος κρύβεται πίσω από τους βράχους, (ζ) το ρομπότ κοιτάζει στην κάμερα, (η) το ρομπότ λέει “Houston, έχουμε πρόβλημα!”. Ας ολοκληρώσουμε το σκηνικό προσθέτοντας τις εναπομείναντες ενέργειες, όπως φαίνεται παρακάτω.

Do the following steps in order
alien moves up
alien says “Slithy toves?”
robot’s head turns around
robot turns to look at alien
Do together
robot moves toward the alien
robot legs walk
alien moves down
robot turns to look at the camera
robot’s head turns red (to signal danger)
robot says “Houston, we have a problem!”

Κάνε τα ακόλουθα βήματα στην σειρά
ο εξωγήινος κινείται
ο εξωγήινος λέει “σύντροφος;”
το κεφάλι του ρομπότ γυρίζει
το ρομπότ γυρίζει και κοιτάει τον εξωγήινο
Κάνε τα ακόλουθα βήματα παράλληλα
το ρομπότ κινείται προς τον εξωγήινο
τα πόδια του ρομπότ περπατούν
ο εξωγήινος κρύβεται
το ρομπότ γυρίζει και κοιτάει στην κάμερα
το κεφάλι του ρομπότ γίνεται κόκκινο (για να σημάνει συναγερμό)
το ρομπότ λέει “Houston, έχουμε πρόβλημα!”

Μετάφραση της διάταξης σκηνών σε κώδικα

Για να μεταγλωττίσετε την διάταξη σκηνών σε κώδικα, ξεκινήστε με το πρώτο βήμα της διάταξης και μεταφράστε την σε μια εντολή. Μετά μεταφράστε το δεύτερο βήμα, μετά το τρίτο, μετά το τέταρτο κοκ μέχρι να μεταφραστούν όλες οι σκηνές. Οι εντολές που χρησιμοποιούνται στον κώδικα προγράμματος είναι οι ίδιες μέθοδοι που μάθατε στις αρχικές ασκήσεις του παραρτήματος Α. Για να δείτε τις διαθέσιμες μεθόδους του αντικειμένου alienOnWheels, πρώτα κάντε κλικ στο αντικείμενο alienOnWheels στο δέντρο αντικειμένων (object tree) και μετά κάντε κλικ στην καρτέλα των μεθόδων στην περιοχή των λεπτομερειών, όπως βλέπετε στην εικόνα 2-2-3.

Στο παράδειγμα, θέλουμε να μεταγλωττίσουμε τις διατάξεις σκηνών σε κώδικα προγράμματος. Ξεκινάμε με το πρώτο βήμα, κάνοντας το

alienOnWheels να ξεπροβάλλει πίσω από τους βράχους. Μια μέθοδος του alienOnWheels είναι η *move* (κίνηση)-μπορούμε να χρησιμοποιήσουμε αυτή τη μέθοδο για να κάνουμε το αντικείμενο alienOnWheels να κινηθεί προς τα πάνω. Το επόμενο βήμα είναι να πει ο alienOnWheels "σύντροφος;" Το alienOnWheels έχει μια say μέθοδο που μπορεί να χρησιμοποιηθεί γι' αυτό το σκοπό. Με ένα παρόμοιο τρόπο, κάθε πράξη στην διάταξη των σκηνών θα μεταγλωττίζεται σε πληροφορίες, χρησιμοποιώντας ενσωματωμένες μεθόδους των αντικειμένων του κόσμου.

ΕΙΚΟΝΑ 2-2-3. Ενσωματωμένες μέθοδοι για την συγγραφή ενός προγράμματος

Διαδοχικές κατά ταυτόχρονων ενεργειών

Από τις διατάξεις σκηνών, είναι ξεκάθαρο ότι οι πρώτες τέσσερις ενέργειες πρέπει να γίνουν η μια μετά την άλλη, διαδοχικά. Μπορούμε να πούμε στο Alice να κάνει αυτές τις πράξεις στην σειρά. Οι άλλες ενέργειες γίνονται ταυτόχρονα. Για παράδειγμα το ρομπότ κινείται μπροστά την ίδια στιγμή που τα πόδια του κινούνται. Πρέπει να δώσουμε διαταγή στο Alice να τα κάνει αυτά ταυτόχρονα. Το "κάνε διαδοχικά" και "κάνε ταυτόχρονα" είναι λέξεις της γλώσσας του Alice. Μπορούμε να τα ονομάσουμε δηλώσεις ελέγχου, επειδή τις χρησιμοποιούμε για να πούμε στο Alice πώς να εκτελέσει τις εντολές σε ένα πρόγραμμα.

Do in order (κάνε διαδοχικά)

Για να πείτε στην Alice να εκτελέσει πληροφορίες στην σειρά, βάζετε τη φόρμα *do in order (κάνε διαδοχικά)* στον συντάκτη, όπως φαίνεται στην εικόνα 2-2-4.

ΕΙΚΟΝΑ 2-2-4. Βάζοντας τη φόρμα *do in order* στον συντάκτη

Οι τέσσερις πρώτες εντολές μπορούν να τοποθετηθούν στο μπλοκ *do in order*. Αρχικά το alienOnWheels επιλέγεται στο δέντρο των αντικειμένων (object tree). Μετά η μέθοδος *move* του alienOnWheels επιλέγεται και τοποθετείται μέσα στο μπλοκ *do in order* , όπως φαίνεται στην εικόνα 2-2-5. Η μέθοδος *move* απαιτεί ορίσματα -ποια κατεύθυνση και πόσο μακριά πρέπει να κινηθεί το alienOnWheels. (Ένα όρισμα είναι μια πληροφορία που πρέπει να δοθεί, έτσι ώστε το Alice να μπορεί να εκτελέσει την ενέργεια). Σε αυτό το παράδειγμα, το alienOnWheels είναι κρυμμένο πίσω από τους βράχους και θέλουμε να κινηθεί προς τα πάνω, οπότε η κατεύθυνση είναι επάνω. Οι βράχοι δεν είναι πολύ ψηλοί, έτσι θα προσπαθήσουμε μια απόσταση ενός μέτρου. (Εάν αυτή η απόσταση δεν είναι η επιθυμητή μπορούμε να την διορθώσουμε αργότερα). Το όνομα της μεθόδου και τα ορίσματα της συνθέτουν την εντολή.

Η τελική εντολή φαίνεται στην εικόνα 2-2-6.

Η δεύτερη εντολή είναι να πει το alienOnWheels “σύντροφος;” Επιλέξτε το alienOnWheels από το δέντρο των αντικειμένων και τοποθετήστε την μέθοδο *say* μέσα στην φόρμα. Θα εμφανιστεί μια καρτέλα προεπιλεγμένων λέξεων, εσείς επιλέξτε το *άλλο,* όπως φαίνεται στην εικόνα 2-2-7. Ένα πλαίσιο κειμένου θα εμφανιστεί, όπου μπορείτε να γράψετε τις λέξεις που θέλετε να πει το alienOnWheels. Γράψτε “σύντροφος;” όπως φαίνεται στην εικόνα 2-2-8 και μετά OK.

Οι δυο πρώτες εντολές φαίνονται στην εικόνα 2-2-9. Όταν τρέξει αυτό το πρόγραμμα (δοκιμάστε το αποτέλεσμα αυτών των δυο εντολών πατώντας το κουμπί Play), το alienOnWheels θα ξεπροβάλλει πίσω από τους βράχους και μετά θα πει “σύντροφος;”

ΕΙΚΟΝΑ 2-2-5. Προσθήκη μιας εντολής κίνησης

ΕΙΚΟΝΑ 2-2-6. Η εντολή κίνησης (*move*) ολοκληρωμένη

ΕΙΚΟΝΑ 2-2-7. Προσθήκη μιας εντολής *say*

ΕΙΚΟΝΑ 2-2-8. Εισαγωγή μιας φράσης

ΕΙΚΟΝΑ 2-2-9. Οι πρώτες δυο εντολές

Για την τρίτη εντολή, το κεφάλι του spiderRobot πρέπει να γυρίσει μια πλήρη περιστροφή (για να δηλώσουμε θαυμασμό) Πως μπορούμε να γυρίσουμε το κεφάλι του spiderRobot; Πηγαίνετε στο δέντρο των αντικειμένων και πατήστε το + που βρίσκεται αριστερά από το αντικείμενο spiderRobot. Έτσι εμφανίζονται τα υπομέρη του αντικειμένου. Πατήστε το + δίπλα στο λαιμό του spiderRobot. Μετά κάνοντας κλικ στο κεφάλι του spiderRobot στο δέντρο των αντικειμένων toy Alice μας επιτρέπει να έχουμε πρόσβαση στις μεθόδους του κεφαλιού. Επιλέξτε και σύρετε την μέθοδο *turn* (γυρίζω) μέσα στον συντάκτη και επιλέξτε το δεξιά ως κατεύθυνση και 1 περιστροφή, όπως φαίνεται στην εικόνα 2-2-10.

ΕΙΚΟΝΑ 2-2-10. Προθήκη μιας εντολής *turn* για το κεφάλι του spiderRobot

Στην τέταρτη εντολή, το spiderRobot θα γυρίσει για να αντικρίσει το alienOnWheels. Η μέθοδος *turn to face* (γυρίζω για να αντικρίσω) του spiderRobot επιλέγεται και ως όρισμα επιλέγεται το alienOnWheels, όπως φαίνεται στην εικόνα 2-2-11. Ο κώδικας του προγράμματος με τις τέσσερις πρώτες εντολές ολοκληρωμένες φαίνεται στην εικόνα 2-2-12.

ΕΙΚΟΝΑ 2-2-11. Προσθήκη μιας εντολής *turn to face*

ΕΙΚΟΝΑ 2-2-12. *Do in order* με τις πρώτες τέσσερις εντολές

Do together (κάνε ταυτόχρονα)

Το επόμενο βήμα στη διάταξη σκηνών απαιτεί δυο πράγματα να γίνουν ταυτόχρονα: το spiderRobot να κινηθεί προς το alienOnWheels, την ίδια στιγμή που θα κινούνται τα πόδια του. Ένα *do together* μπαίνει μετά και μέσα στο *do in order*, όπως φαίνεται στην εικόνα 2-2-13. Παρατηρήστε την οριζόντια γραμμή (πράσινη) στην εικόνα 2-2-13. Η πράσινη γραμμή υποδεικνύει που θα μπει η εντολή *do together.*

ΕΙΚΟΝΑ 2-2-13. Προσθήκη μιας *do together* (μέσα στην *do in order*)

Το αποτέλεσμα αυτής της τροποποίησης, που φαίνεται στην εικόνα 2-2-14, είναι ότι το μπλοκ *do together* είναι ενσωματωμένο μέσα στο μπλοκ *do in order.* Η ενσωμάτωση σημαίνει ότι μια εντολή ενός προγράμματος είναι μέσα σε μια άλλη. Σημειώστε ότι η ενσωμάτωση του μπλοκ *do together* μέσα στο *do in order,* είναι η καλύτερη λύση να δημιουργήσετε αυτό το αποτέλεσμα. Η ενσωμάτωση δεν είναι αναγκαστική. Αυτά τα μπλοκ κώδικα μπορούν να δουλέψουν μαζί ή και ξεχωριστά με πολλούς διαφορετικούς τρόπους.

ΕΙΚΟΝΑ 2-2-14. *Do together* μέσα στην *do in order*

Τώρα μπορούν οι μέθοδοι να δημιουργηθούν μέσα στο μπλοκ *do together*, για να κινηθεί το spiderRobot και να περπατήσει ταυτόχρονα. Η μέθοδος *move* είναι εύκολη. Απλά προσθέστε την μέθοδο και βάλτε ορίσματα κατεύθυνσης μπροστά και απόσταση 1 μέτρο, όπως φαίνεται στην εικόνα 2-2-15.

ΕΙΚΟΝΑ 2-2-15. Προσθήκη μιας εντολής *move* μέσα σε ένα μπλοκ *do together*

Το spiderRobot έχει πολλά πόδια. Για να απλοποιήσουμε το πρόγραμμά μας, θα υλοποιήσουμε την κίνηση δυο μόνο ποδιών (backLeft και frontRight). Ένα πόδι κινείται αλλάζοντας θέση της άρθρωσης (ίδιο με το λύγισμα του γονάτου). Ας ξεκινήσουμε δημιουργώντας μια εντολή κίνησης του backLeftLegUpperJoint (αλλαγής θέσεως της πάνω άρθρωσης του πίσω αριστερού ποδιού). Πρώτα επιλέξτε το backLeftLegUpperJoint υπομέρος του backLeftLegBase υπομέρος του spiderRobot και μετά τοποθετήστε την μέθοδο *turn* μέσα στο μπλοκ *do together* όπως φαίνεται στην εικόνα 2-2-16.

Παρατηρήστε ότι το popup μενού επιτρέπει να επιλέξετε ορίσματα *direction (κατεύθυνσης)* και *amount (ποσού κίνησης)*. Επιλέξτε *forward (μπροστά)* για direction και *other (άλλο)* για το amount. Όταν επιλεχθεί το *other,* εμφανίζεται ένα πλαίσιο με αριθμούς (σαν αριθμομηχανή). Επιλέγουμε 0,1 περιστροφή, κάνοντας κλικ στους αριθμούς. Πως γνωρίζαμε ότι το ποσό των 0,1 περιστροφών ήταν το κατάλληλο; Δεν γνωρίζαμε. Απλά επιλέξαμε διάφορα ποσά μέχρι να βρούμε τελικά αυτό που δίνει το καλύτερο οπτικό αποτέλεσμα της κίνησης λυγίζοντας την άρθρωση του ποδιού. Αυτό είναι ένα

παράδειγμα στρατηγικής δοκιμής και λάθους. Πάντα προτείνουμε ότι οι καλά σχεδιασμένες στρατηγικές δοκιμής και λάθους είναι χρήσιμες.

Λογικά όταν μια άρθρωση γυρίζει προς μια κατεύθυνση μετά πρέπει να γυρίσει προς την άλλη κατεύθυνση (για να επιτευχθεί η ισορροπία). Οι δυο πληροφορίες *turn* για το πίσω αριστερό πόδι φαίνονται στην εικόνα 2-2-17.

Χρησιμοποιώντας την ίδια τεχνική, χρησιμοποιούνται εντολές για να κινηθεί και το frontRight πόδι. Οι τελικές εντολές που συνθέτουν το περπάτημα φαίνονται στην εικόνα 2-2-18.

ΕΙΚΟΝΑ 2-2-16. Τοποθέτηση μιας εντολής *turn* για το backLeftLegUpperJoint

ΕΙΚΟΝΑ 2-2-17. Εντολές *turn forward* και *backward*

ΕΙΚΟΝΑ 2-2-18. Εντολές για την προσομοίωση βαδίσματος του spiderRobot

Λάθη

Θα θυμάστε ότι τα τέσσερα βήματα στην δημιουργία ενός κινούμενου σχεδίου είναι: διάβασμα, σχεδιασμός, υλοποίηση και τεστ. Τώρα που έχουν δημιουργηθεί αρκετές γραμμές κώδικα (υλοποίηση), είναι καλή ιδέα να τεστάρουμε αυτό που γράψαμε. Δεν χρειάζεται να περιμένετε να ολοκληρωθεί το πρόγραμμα. Για να τεστάρετε τις πληροφορίες που έχετε γράψει μέχρι τώρα, πατήστε το κουμπί **Play**. Το alienOnWheels εμφανίζεται πίσω από τους βράχους και μετά λέει “σύντροφος;” Το κεφάλι του spiderRobot γυρίζει και μετά το spiderRobot γυρίζει προς το μέρος του alienOnWheels. Μέχρι εδώ καλά, αλλά όταν το spiderRobot κινείται μπροστά, τα πόδια δεν κινούνται. Αυτό είναι, οι αρθρώσεις δεν φαίνονται να γυρίζουν καθόλου!

Ο λόγος για τον οποίο οι αρθρώσεις δεν κινούνται σημαίνει ότι το πρόγραμμα έχει λάθος. (Όταν διορθώνουμε τα λάθη ενός προγράμματος λέμε ότι κάνουμε αποσφαλμάτωση). Το πρόβλημα είναι, ότι στον κώδικα που φαίνεται παραπάνω, οι εντολές της *turn* της άρθρωσης του ποδιού είναι γραμμένες μέσα σε ένα μπλοκ *do together*. Εάν οι αρθρώσεις κινούνται μπροστά και πίσω ταυτόχρονα η μια θα ακυρώνει την άλλη και έτσι τα πόδια του spiderRobot δεν θα κινούνται καθόλου! Για να φτιαχτεί αυτό το πρόβλημα, είναι απαραίτητο να μπούν οι εντολές της *turn* για την άρθρωση του δεξιού ποδιού μέσα σε ένα μπλοκ *do in order* και το ίδιο για το αριστερό πόδι, όπως φαίνεται στην εικόνα 2-2-19.

ΕΙΚΟΝΑ 2-2-19. Διορθωμένες εντολές για το περπάτημα του spiderRobot

Τώρα το backLeft και το frontRight και το πόδι του spiderRobot κινούνται. Υπάρχει μια χρήσιμη παρατήρηση. Οι προκαθορισμένες εντολές κίνησης απαιτούν ένα δευτερόλεπτο για να εκτελεστούν. Φυσιολογικά μέσα σε ένα *κάνε do together* μπλοκ, κάθε μια από τις εντολές θα έπρεπε να χρησιμοποιεί το ίδιο ποσό χρόνου. Εφόσον χρειάζεται ένα δευτερόλεπτο για να κινηθεί το spiderRobot μπροστά 1 μέτρο, το ίδιο θα κάνει και για να κινηθεί

πίσω για όλα τα πόδια. Παρόλα αυτά γίνονται δυο βήματα κατά το λύγισμα της άρθρωσης (μπροστά και μετά πίσω). Κάθε βήμα κατά το λύγισμα πρέπει να διαρκεί μισό δευτερόλεπτο. Για να αλλάξετε την διάρκεια, πατήστε στο *more* (πρόσθετα), επιλέξετε το αντικείμενο του μενού *duration* (διάρκεια) και επιλέξτε 0,5 δευτερόλεπτα, όπως φαίνεται στην εικόνα 2-2-20.

ΕΙΚΟΝΑ 2-2-20. Αλλαγή της διάρκειας μιας εντολής

Χρήση μιας ιδιότητας

Πρέπει ακόμη να συμπληρώσουμε τις τελευταίες τέσσερις πράξεις που περιγράφονται στην διάταξη σκηνών (το alienOnWheels κρύβεται, το spiderRobot γυρίζει και κοιτάζει την κάμερα, το κεφάλι του spiderRobot γίνεται κόκκινο, και το spiderRobot λέει “Houston, έχουμε πρόβλημα!”). Έχετε ήδη χρησιμοποιήσει τις εντολές: *move (κίνηση), turn to face (αντικρίζω)* και *say (μιλάω)*-οπότε αυτές είναι εύκολες να δημιουργηθούν. Η μόνη νέα εντολή είναι αυτή που απαιτεί το κεφάλι του spiderRobot να γίνει κόκκινο (ένα σήμα κινδύνου). Για να το επιτύχουμε αυτό, χρησιμοποιούμε τις ιδιότητες χρώματος του κεφαλιού του ρομπότ. Για να δείτε μια λίστα ιδιοτήτων του κεφαλιού του spiderRobot, επιλέξτε το κεφάλι του spiderRobot στο δέντρο αντικειμένων και επιλέξτε την καρτέλα ιδιότητες στην περιοχή λεπτομερειών (κάτω αριστερά στο παράθυρο του Alice), όπως φαίνεται στην εικόνα 2-2-21.

ΕΙΚΟΝΑ 2-2-21. Οι ιδιότητες του κεφαλιού του spiderRobot

Αυτό που θέλουμε να κάνουμε είναι να αλλάξουμε το χρώμα του κεφαλιού του spiderRobot, αφού γυρίσει και αντικρίσει την κάμερα. Φυσικά οι ιδιότητες μπορούν να αλλάξουν όταν δημιουργείται ο αρχικός κόσμος. Αλλά εμείς θέλουμε να αλλάξουμε το χρώμα κατά την διάρκεια της σκηνής. Μια εντολή πρέπει να δημιουργηθεί για να αλλάξει το χρώμα. Η εικόνα 2-2-22 περιγράφει την παραπάνω εντολή. Το πλακίδιο του χρώματος για το κεφάλι του spiderRobot τοποθετείται στο μπλοκ *do in order*. Μετά επιλέγεται το χρώμα κόκκινο από το popup μενού των διαθέσιμων χρωμάτων.

ΕΙΚΟΝΑ 2-2-22. Αλλαγή το χρώματος του κεφαλιού του spiderRobot

Ο τελικός κώδικας για ολόκληρο το πρόγραμμα φαίνεται στην εικόνα 2-2-23.

ΕΙΚΟΝΑ 2-2-23. Ο κώδικας του προγράμματος για ολόκληρο το κινούμενο σχέδιο

Σχόλια

Τώρα που γράψαμε το πρώτο μας πρόγραμμα, είναι ώρα να κοιτάξουμε, μια χρήσιμη συνιστώσα στα προγράμματα-σχόλια. Τα σχόλια δεν είναι εντολές που προκαλούν κάποια ενέργεια. Αυτό σημαίνει ότι το Alice μπορεί να

αγνοήσει τα σχόλια όταν εκτελεστεί το πρόγραμμα. Παρόλα αυτά, τα σχόλια θεωρούνται ένα καλό προγραμματιστικό στυλ και είναι άκρως χρήσιμα για τα άτομα που διαβάζουν το πρόγραμμα. Τα σχόλια βοηθούν τον αναγνώστη να καταλάβει τι κάνει το πρόγραμμα. Αυτό είναι πρακτικά χρήσιμο όταν κάποιος άλλος θέλει να διαβάσει τον κώδικα του προγράμματος για να δει τι έχετε γράψει και πως το γράψατε.

Τα σχόλια στο Alice δημιουργούνται χρησιμοποιώντας τις πράσινες γραμμές // σε ένα πρόγραμμα και μετά γράφοντας μια περιγραφή του τι κάνει ο κώδικας. Η εικόνα 2-2-24 περιγράφει την μέθοδο *World.my first method* με ένα σχόλιο. Ένα σχόλιο θα έπρεπε να συμπεριλαμβάνεται στην αρχή μιας μεθόδου για να επεξηγεί το τι κάνει η μέθοδος. Αυτό το είδος του σχόλιου είναι σαν να γράφετε ένα συμπέρασμα.

Επίσης, μικρά τμήματα αρκετών γραμμών κώδικα που εκτελούν μια ενέργεια μπορούν να τεκμηριωθούν χρησιμοποιώντας ένα σχόλιο. Ένα πρόσθετο σχόλιο φαίνεται στην εικόνα 2-2-25.

Αυτό το σχόλιο επεξηγεί ότι αυτό το μικρό τμήμα κώδικα χρησιμοποιείται για να κινήσει μπροστά το spiderRobot όσο τα πόδια του κινούνται.

ΕΙΚΟΝΑ 2-2-24. Ένα περιληπτικό σχόλιο για την μέθοδο *World.my first method*

ΕΙΚΟΝΑ 2-2-25. Ένα σχόλιο για ένα μικρό τμήμα κώδικα

Μυστικά και Τεχνικές 2

Οδηγίες για προσανατολισμό και κίνηση

Η μέθοδος *orient to* (προσανατολισμού)

Κάθε αντικείμενο σε ένα κόσμο του Alice έχει το δικό του σύστημα συντεταγμένων το οποίο παρέχει μια αίσθηση κατεύθυνσης-προσανατολισμού. Για να υλοποιήσουμε τον τρόπο με τον οποίο το σύστημα συντεταγμένων παρέχει προσανατολισμό για ένα αντικείμενο, τοποθετήσαμε ένα ορατό σύστημα αξόνων σε μια μαϊμού όπως φαίνεται στην εικόνα Τ-2-1.

ΕΙΚΟΝΑ Τ-2-1. Ο προσανατολισμός της μαϊμούς

Όταν δυο αντικείμενα πρέπει να κινηθούν ταυτόχρονα, ο προσανατολισμός των δυο αντικειμένων πρέπει να συγχρονιστεί. Στον κόσμο που φαίνεται στην εικόνα Τ-2-2, θέλουμε η μαϊμού να μείνει πάνω στην μπάλα, όσο η μπάλα θα κινείται μπροστά για μια μικρή απόσταση.

Ο κώδικας που γράψαμε για να κάνουμε τη μαϊμού και την μπάλα να κινηθούν μπροστά φαίνεται στην εικόνα Τ-2-3.

Φανταστείτε την έκπληξη μας όταν τα δυο αντικείμενα ακολουθούν διαφορετικές πορείες. Γιατί συνέβη αυτό; Γιατί η μπάλα είναι ένα αντικείμενο για το οποίο δεν είναι εύκολο να πούμε, μόνο κοιτάζοντάς το, ποια κατεύθυνση είναι μπροστά και ποια πίσω. Προφανώς, όταν τοποθετήσαμε την μπάλα στη σκηνή δεν την τοποθετήσαμε έτσι ώστε η μπροστινή της κατεύθυνση να είναι ίδια με την μπροστινή κατεύθυνση της μπάλας, όπως φαίνεται στην εικόνα Τ-2-4. Έτσι όταν η μπάλα και η μαϊμού κινούνται μπροστά, κινούνται σε διαφορετικές κατευθύνσεις.

ΕΙΚΟΝΑ Τ-2-2. Η μαϊμού πηδάει πάνω σε μια μπάλα

ΕΙΚΟΝΑ Τ-2-3. Κώδικας για να κινηθεί η μαϊμού και η μπάλα μπροστά

ΕΙΚΟΝΑ Τ-2-4. Η μαϊμού και η μπάλα κινούνται μπροστά σε διαφορετικές κατευθύνσεις

Ο τρόπος επίλυσης αυτού του προβλήματος είναι να συγχρονίσουμε τις κατευθύνσεις των δυο αντικειμένων. Σε αυτό το παράδειγμα, θα χρησιμοποιήσουμε μια εντολή προσανατολισμού της μπάλας προς την μαϊμού (*toyball.orient to (monkey)*), όπως φαίνεται στην εικόνα Τ-2-5.

Το αποτέλεσμα φαίνεται στην εικόνα Τ-2-6. Τώρα η μπάλα έχει τον ίδιο προσανατολισμό με την μαϊμού. Αυτό σημαίνει ότι τα δυο αντικείμενα θα κινούνται στην ίδια κατεύθυνση, όταν δοθεί μια εντολή για κίνηση εμπρός σε καθένα από αυτά.

Η εντολή *orient to* (προσανατολισμού) μπορεί να φανεί λίγο παράξενη, αλλά απλά λέει στο λογισμικό Alice ότι το πρώτο αντικείμενο πρέπει να έχει τον ίδιο προσανατολισμό με το δεύτερο αντικείμενο. Έτσι αν ορίσουμε δυο αντικείμενα να έχουν τον ίδιο προσανατολισμό, τότε αυτά τα αντικείμενα είναι συγχρονισμένα-έχουν την ίδια αίσθηση για το πάνω, κάτω, δεξιά, αριστερά.

ΕΙΚΟΝΑ Τ-2-5. Διαδοχικά μενού για την *orient to*

ΕΙΚΟΝΑ Τ-2-6. Τώρα η μαϊμού και η μπάλα έχουν τον ίδιο προσανατολισμό

Η ιδιότητα *vehicle (όχημα)*

Ένας άλλος τρόπος να συγχρονίσουμε τις κινήσεις δυο αντικειμένων, είναι να εκμεταλλευτούμε μια ιδιότητα που καλείται *vehicle*. Σαν υλοποίηση της ιδιότητας όχημα, θεωρείστε ένα τσίρκο όπου το αντικείμενο chicken καβαλάει το horse, όπως φαίνεται στην εικόνα Τ-2-7. Σαν μέρος της παράστασης, το horse κάνει μια συγκεκριμένη βόλτα με το chicken στην πλάτη του.

Για να συγχρονίσετε την κίνηση του chicken και του horse, μπορείτε να κάνετε το horse *vehicle* για το chicken. Για να δημιουργήσετε αυτό το ειδικό εφέ, επιλέξτε το chicken στο δέντρο αντικειμένων και μετά επιλέξτε την καρτέλα ιδιότητες. Πηγαίνετε στην ιδιότητα *vehicle* και κάντε κλικ δίπλα της. Εμφανίζεται μια λίστα από πιθανά vehicles για το chicken και εσείς επιλέγετε το horse. Αυτό φαίνεται στην εικόνα Τ-2-8. Τώρα όταν το horse κινείται, θα κινείται και το chicken.

ΕΙΚΟΝΑ Τ-2-7. Το chicken καβαλάει τohorse

ΕΙΚΟΝΑ Τ-2-8. Επιλέγοντας το horse ως vehicle για το chicken

Ορίσματα: d*uration (διάρκεια), style (στυλ), asSeenBy (όπως φαίνεται από)*

Οι εντολές κίνησης (π.χ. *move, turn, roll*) τελειώνουν με μια ετικέτα "*more…*" η οποία μπορεί να αλλάξει, όπως φαίνεται στην εικόνα Τ-2-9.

Όταν κάνουμε κλικ στο *more,* ένα popup μενού σας επιτρέπει να επιλέξετε από μια λίστα τριών ορισμάτων, duration, style, asSeenBy όπως στην εικόνα Τ-2-10.

ΕΙΚΟΝΑ Τ-2-9. Η ετικέτα "*more…*"

ΕΙΚΟΝΑ Τ-2-10. Popup μενού για το "*more…*"

Το όρισμα duration γνωστοποιεί στο Alice ένα ποσό χρόνου (σε δευτερόλεπτα) που διαρκεί η κίνηση. (Το Alice υποθέτει οτι ο χρόνος μιας κίνησης είναι 1 δευτερόλεπτο). Μια κίνηση που διαρκεί μηδέν δευτερόλεπτα είναι μια στιγμιαία κίνηση. Αρνητικές τιμές δεν χρησιμοποιούνται. Στο πρώτο μας παράδειγμα, ένα όρισμα duration χρησιμοποιήθηκε για να μικρύνει το διάστημα του χρόνου κίνησης των ποδιών του ρομπότ.

Το όρισμα style καθορίζει τον τρόπο με τον οποίο μια εντολή κίνησης ταιριάζει με την επόμενη. Οι επιλογές είναι *gently (*αρχή και τέλος ευγενικά), *abruptly (*αρχή και τέλος απότομα*)*, *begin gently (*και τέλος απότομο*),end gently (*απότομη αρχή*).* Για να πάρετε τον σωστό βαθμό εξομάλυνσης μιας κίνησης, είναι καλό να πειραματιστείτε με το style.

Όπως περιγράψαμε στο κεφάλαιο 1, κάθε αντικείμενο του Alice έχει την δική του αίσθηση της κατεύθυνσης (προσανατολισμός). Αλλά μπορείτε να χρησιμοποιήσετε το όρισμα *asSeenby* για να πείτε στο Alice να χρησιμοποιήσει τον προσανατολισμό ενός αντικειμένου για να καθοδηγήσει ένα άλλο αντικείμενο. Αυτό θα εξηγηθεί καλύτερα με ένα παράδειγμα. Υποθέστε ότι έχουμε ένα ελικόπτερο σε μια εκπαιδευτική αποστολή ενός πιλότου, όπως φαίνεται στην εικόνα Τ-2-11.

ΕΙΚΟΝΑ Τ-2-11. Εκπαιδευτική αποστολή

Ο κώδικας της εικόνας Τ-2-12, προορίζεται να κινήσει το ελικόπτερο αριστερά και μετά να το κινήσει προς τα επάνω.

ΕΙΚΟΝΑ Τ-2-12. Κώδικας για roll και στη συνέχεια move

Τρέχοντας το πρόγραμμα, βλέπουμε ότι το αποτέλεσμα δεν είναι αυτό που περιμέναμε. Όταν το ελικόπτερο κινείται προς τα επάνω, το κάνει άλλα με τον δικό του προσανατολισμό, όπως φαίνεται στην εικόνα Τ-2-13.

ΕΙΚΟΝΑ Τ-2-13. Επάνω σύμφωνα με τον προσανατολισμό του ελικοπτέρου

Αυτό που θέλαμε όμως είναι μια κίνηση προς τα πάνω σε σχέση προς το έδαφος. Για να επιλύσετε αυτό το πρόβλημα, κάνουμε κλικ στο *more* και μετά επιλέγουμε το *asSeenBy* → *ground* (έδαφος), όπως φαίνεται στην εικόνα Τ-2-14.

ΕΙΚΟΝΑ Τ-2-14. Επιλογή του *asSeenBy* Ground

Ο τελικός κώδικας, φαίνεται στην εικόνα Τ-2-15, δίνει το επιθυμητό αποτέλεσμα.

ΕΙΚΟΝΑ Τ-2-15. Ο τροποποιημένος κώδικας

Οι μέθοδοι: *turn to face (γυρίζω για να αντικρίσω)* και *point at (σημαδεύω το).*

Όπως συζητήθηκε παραπάνω, το όρισμα *asSeenBy* σε μια εντολή κίνησης χρησιμοποιεί τον προσανατολισμό ενός αντικειμένου για να οδηγήσει την κίνηση ενός άλλου αντικειμένου. Δυο ειδικές μέθοδοι είναι χρήσιμες για να κάνουμε ένα αντικείμενο να γυρίσει και να αντικρίσει ένα άλλο αντικείμενο. Μια μέθοδος *turn to face (γυρίζω να αντικρίσω)* κάνει ένα αντικείμενο να στρέφεται γύρω από τον άξονα του μέχρι το μπροστά τμήμα του να αντικρίσει ένα άλλο αντικείμενο. Στην προηγούμενη ενότητα αυτού του κεφαλαίου, η *turn to face* χρησιμοποιήθηκε γα να γυρίσει το spiderRobot προς το alienOnWheels.

Μια δεύτερη μέθοδος η *point at (σημαδεύω το)*, χρησιμοποιείται για να ευθυγραμμίσει δυο αντικείμενα από το κέντρο του ενός στο κέντρο του άλλου. Είναι ευκολότερο να εξηγήσουμε την *point at* με ένα παράδειγμα. Στην εικόνα Τ-2-16, οι κωπηλάτες στο lifeboat (Vehicles) θέλουν να κωπηλατήσουν προς τον lighthouse (Beach) του island (Environments). Ένα εμφανές πρώτο βήμα είναι να κάνετε το lifeboat και τους κωπηλάτες να κοιτάξουν το lighhouse του island.

ΕΙΚΟΝΑ Τ-2-16. Το lifeboat, lighthouse και island στην αρχική σκηνή

Η μέθοδος *point at* μπορεί να χρησιμοποιηθεί για να δώσει ως στόχο του lifeboat στο lighthouse. Μετέπειτα εντολές κίνησης μπορούν να χρησιμοποιηθούν για να κινήσουν το lifeboat προς το lighthouse, όπως φαίνεται στην εικόνα Τ-2-17.

ΕΙΚΟΝΑ Τ-2-17. Κώδικας για να στοχεύσει το lifeboat προς το lighthouse

Αυτή η εντολή που γυρίζει το lifeboat προς το μέρος του lighhouse, έχει ως συνέπεια να γείρει το lifeboat από την μια μεριά έτσι ώστε να φαίνεται ότι το lifeboat βουλιάζει, όπως φαίνεται στην εικόνα Τ-2-18.

ΕΙΚΟΝΑ Τ-2-18. Το lifeboat γέρνει στην εντολή *point at*

Η εντολή *point at* ευθυγραμμίζει το κέντρο του lifeboat με το κέντρο του lighthouse, που είναι σε υψηλότερο επίπεδο απ' ότι το lifeboat, όπως φαίνεται στην εικόνα Τ-2-19. Με αποτέλεσμα το lifeboat να γέρνει.

ΕΙΚΟΝΑ Τ-2-19. Ευθυγράμμιση από κέντρο σε κέντρο

Μπορεί να θέλετε ή να μην θέλετε το lifeboat να γέρνει. Για να μπορέσουμε να ελέγξουμε την εντολή *point at,* πρόσθετα σχόλια είναι διαθέσιμα στο popup μενού "*more…*", όπως φαίνεται στην εικόνα Τ-2-20. Επιλέγοντας το *true* για το όρισμα *onlyAffectYaw (επιρροή της παρέκκλισης),* επιτρέπει ένα αντικείμενο να στοχεύσει προς ένα άλλο χωρίς να γέρνει (pitching). Yaw (εκτροπή) είναι ένας τεχνικός όρος που σημαίνει μια αριστερή-δεξιά κίνηση και η κλίση είναι μια κίνηση κουνιστής πολυθρόνας. (Όταν το *onlyAffectYaw* είναι *true,* η εντολή *point at* λειτουργεί όπως η εντολή *turn to face*). Παρατηρήστε ότι στα παιχνίδια στόχων και στις κινήσεις αεροπλάνων η *onlyAffectYaw* πρέπει να είναι *false.*

ΕΙΚΟΝΑ Τ-2-20. Επιλογή της *onlyAffectYaw→ true*

Ασκήσεις

2-1 Ασκήσεις

1.Δημιουργία διατάξεων σκηνικών

Δημιουργήστε μια οπτική και μια βασισμένη σε κείμενο διάταξη σκηνών (δύο) για κάθε ένα από τα ακόλουθα σενάρια:

(α) Ένα παιχνίδι παιδιού: Η Alice, ο λαγός, και η γάτα απολαμβάνουν ένα παιχνίδι με καρέκλες σε ένα σκηνικό πάρτι με τσάι. Ένας από τους χαρακτήρες φωνάζει "αλλάξτε θέση" και τότε όλοι τρέχουν γύρω από το τραπέζι και στέκονται δίπλα στην επόμενη καρέκλα. Μετά την αλλαγή θέσης μια καρέκλα βγαίνει από το παιχνίδι και ο χαρακτήρας που στέκεται δίπλα της αποβάλλεται από το παιχνίδι (φεύγει από το τραπέζι).

(β) Ένα παιχνίδι βίντεο: Ένα μαχητικό αεροπλάνο επιστρέφει στο κατάστρωμα πλοίου μετά από μια εκπαιδευτική επιστολή. Το αεροπλάνο κάνει μισό κύκλο πάνω από το κατάστρωμα για να βρει την σωστή θέση προσγείωσης και σταδιακά κατεβαίνει. Το πλοίο βρίσκεται σε κίνηση, οπότε το αεροπλάνο πρέπει να ρυθμίζει συνεχώς την κατάβαση για να προσγειωθεί τελικά στο κατάστρωμα. Αφού το

αεροπλάνο τελικά ακουμπήσει το κατάστρωμα, συνεχίζει να κινείται μέχρι να σταματήσει εντελώς.

(γ) Μια προσομοίωση Ολυμπιακών αγώνων: Ένας παγοδρόμος κάνει εξάσκηση για τους Ολυμπιακός. Θα εκτελέσει μια ακολουθία αλμάτων και περιστροφών, όσο παίζει κλασσική μουσική.

2-2 Ασκήσεις

2. Η πρώτη αναμέτρηση του ρομπότ – εκτεταμένη

(α) Οι κόσμοι που χρησιμοποιούνται στα παραδείγματα του βιβλίου υπάρχουν στο cd. Κάθε κόσμος που παρέχεται στο cd έχει την αρχική στιγμή έτοιμη με το background σκηνικό και τα αντικείμενα στην θέση τους, όπως φαίνεται στο παράδειγμα. Σε αυτή την ενότητα, ο κόσμος του παραδείγματος είναι μια πρώτη αναμέτρηση όπου ένα ρομπότ συναντά ένα εξωγήινο στο φεγγάρι. Ξεκινήστε το Alice. Μετά κάντε αντιγραφή τον κόσμο FirstEncounter.a2w στον υπολογιστή σας. Στο Alice χρησιμοποιήστε File-Open για να ανοίξετε τον κόσμο. Ακολουθήστε τα βήματα που παρουσιάστηκαν στο κεφάλαιο για να αναδομήσετε το πρόγραμμα.

(β) Στον κώδικα που παρουσιάστηκε σε αυτή την ενότητα, μόνο τα δυο πόδια κινούνται (backLeft και frontRight). Προσθέστε κώδικα για να κινήσετε και τα άλλα πόδια. Αποθηκεύστε τον κόσμο σας.

3.Χιονάνθρωποι

Δημιουργήστε ένα κόσμο με χιονάνθρωπους όπως φαίνεται στην σκηνή παρακάτω. Αρκετοί χιονάνθρωποι βρίσκονται σε ένα τοπίο καλυμμένο με χιόνι.

ΕΙΚΟΝΑ

Ένας χιονάνθρωπος προσπαθεί να συναντήσει μια άλλη χιονάνθρωπο που μιλάει με ένα φίλο. Προσπαθεί να της τραβήξει την προσοχή. Γυρίζει να αντικρίσει την χιονάνθρωπο και λέει “μπροστά” . Αυτή γυρίζει και κοιτάζει τον χιονάνθρωπο και αυτός της ανοιγοκλείνει τα μάτια. Αυτή κοκκινίζει, αυτή δεν ενδιαφέρεται να τον συναντήσει. Τον σπρώχνει και γυρίζει πίσω για να μιλήσει με τον φίλο της. Του αγκαλιάζει το κεφάλι απογοητευμένα και αναχωρεί.

4. Ψάρι που κινείται κυκλικά

Δημιουργήστε ένα νησί με ένα ψάρι στο νερό. Τοποθετήστε το ψάρι και την κάμερα έτσι ώστε να πάρετε την εικόνα που φαίνεται παρακάτω. Γράψτε ένα πρόγραμμα που να κάνει το ψάρι να κολυμπάει κυκλικά μπροστά από το νησί. Μετά κάντε το ψάρι να κολυμπάει γύρω από το νησί. Αν θέλετε κάντε το ψάρι να κινηθεί σε σχέση με το νησί *asSeenBy.* Τέλος , κάντε το ψάρι να πηδήξει έξω από το νερό και μετά

να βουτήξει μέσα στο νερό. Η τελική σκηνή πρέπει να μοιάζει με την αρχική, με το ψάρι στην θέση που βρισκόταν στην αρχή.

ΕΙΚΟΝΑ

5. Η Χελώνα παίρνει ένα κουλούρι

Δημιουργήστε ένα κόσμο με μια χελώνα, ένα σκαμπό, και ένα κουλουράκι, όπως φαίνεται παρακάτω. Βάλτε το κουλουράκι πάνω στο σκαμπό. Τοποθετήστε την χελώνα και το σκαμπό δίπλα δίπλα και μετά χρησιμοποιήστε μια μέθοδο κίνησης *move* για να μετακινήσετε την χελώνα δυο μέτρα μακριά από το σκαμπό. Χρησιμοποιήστε μια μέθοδο *turn to face* για να σιγουρευτείτε ότι η χελώνα αντικρίζει το σκαμπό. Γράψτε ένα πρόγραμμα για να κινήσετε την χελώνα μέχρι το σκαμπό και να πάρει το κουλουράκι. Κάντε την χελώνα να δείξει ευχαρίστηση της κοιτάζοντας την κάμερα και κουνώντας το χέρι.

ΕΙΚΟΝΑ

Περίληψη

Αυτό το κεφάλαιο εισήγαγε τις θεμελιώδεις αρχές του προγραμματισμού με το Alice. Ξεκινήσαμε διαβάζοντας το σενάριο και δημιουργώντας μια διάταξη σκηνών. Το σενάριο μας βοηθάει να στήσουμε την σκηνή-μας λέει ποια αντικείμενα θα χρησιμοποιηθούν και τι ενέργειες θα κάνουν. Μια διάταξη σκηνών αναλύει ένα σενάριο σε μια ακολουθία σκηνών που παρέχουν την σειρά με την οποία θα εκτελεστούν οι πράξεις.

Μερικές πράξεις σε ένα πρόγραμμα θα εκτελεστούν σε σειρά και μερικές άλλες ταυτόχρονα. Μια διάταξη σκηνών χρησιμοποιείται σαν καθοδηγητής για την υλοποίηση. Το τεστ του κώδικα είναι ένα σημαντικό βήμα για να βρείτε και να διορθώσετε τα λάθη.

Τα σχόλια χρησιμοποιούνται για να τεκμηριώσουν τις μεθόδους, όπου ο σκοπός της μεθόδου ή ενός μικρού τμήματος μιας μεθόδου δεν είναι εμφανής. Τα σχόλια θεωρούνται καλό προγραμματιστικό στυλ.

Σημαντικά θέματα σε αυτό το κεφάλαιο

- Ένα σενάριο είναι μια δήλωση προβλήματος που περιγράφει το κινούμενο σχέδιο σε σχέση με το πρόβλημα, ή το παιχνίδι, ή τη προσομοίωση.
- Μια διάταξη σκηνών μπορεί να είναι είτε οπτική, είτε βασισμένη σε κείμενο.
- Μια οπτική διάταξη σκηνών είναι μια ακολουθία από ζωγραφισμένων στο χέρι σκηνών ή από φωτογραφίες σκηνών που αποσυνθέτουν ένα σενάριο σε μια ακολουθία σκηνών με μετάβαση μεταξύ των σκηνών.

- Κάθε σκίτσο αναπαριστά ένα κινούμενο σχέδιο που δείχνει την θέση, το χρώμα, το μέγεθος και άλλες ιδιότητες των αντικειμένων της σκηνής.
- Μια διάταξη σκηνών που βασίζεται σε κείμενο είναι κάτι σαν μια λίστα υποχρεώσεων, που παρέχει μια αλγοριθμική λίστα βημάτων που περιγράφουν διαδοχικές και/ή ταυτόχρονες πράξεις.
- Ένα πρόγραμμα αποτελείται από γραμμές κώδικα που καθορίζουν τις ενέργειες που πρέπει να εκτελεστούν.
- Οι χαρακτήρες που βλέπεται σε ένα κόσμο του Alice είναι γνωστά ως αντικείμενα. Γράφουμε δηλώσεις αντικειμένων για να κάνουμε τα αντικείμενα να κινηθούν, τοποθετώντας μεθόδους σε ένα συντάκτη.
- Στο Alice, ο κώδικας του προγράμματος δομείται σε μπλοκ *do together* και *do in order* για να ορίσει στο Alice ποιες εντολές θα εκτελεστούν σε σειρά και ποιες ταυτόχρονα.
- Οι περίπλοκες κινήσεις μπορεί να δομηθούν από απλές συνθέσεις κώδικα *do together* και *do in order.* Γνωρίζοντας τι σημαίνει κάθε ένα από τα δύο και πώς να τα συνδυάσετε (φωλιάζοντας το ένα μέσα στο άλλο) σας παρέχεται ένας τρόπος να εκτελέσετε πιο περίπλοκες πράξεις.

Κεφάλαιο 3
Προγραμματίζοντας: Συναρμολογώντας τα κομμάτια

Σε αυτό το κεφάλαιο, θα δείτε πώς να συναρμολογείτε διαφορετικά είδη κώδικα “κομμάτια κώδικα” για να κάνετε το πρόγραμμα να λειτουργεί όπως θέλετε. Τα κομμάτια του κώδικα θα περιέχουν:

Εντολή-μια διαταγή που εκτελείται έτσι ώστε τα αντικείμενα να κάνουν μια συγκεκριμένη ενέργεια.

Δομές ελέγχου-μια εντολή που ελέγχει την εκτέλεση ενός μπλοκ εντολών

Συνάρτηση-διατυπώνει μια ερώτηση για μια υπόθεση ή υπολογίζει μια τιμή

Παράσταση-μια μαθηματική πράξη μεταξύ αριθμών ή άλλων τιμών.

ΕΙΚΟΝΑ

Έχετε επίσης χρησιμοποιήσει τις δομές ελέγχου *do in order* και *do together.* Η δομή *do in order* λέει στο Alice να εκτελέσει εντολές διαδοχικά, την μια μετά την άλλη. Η δομή *do together* λέει στο Alice να εκτελέσει τις εντολές ταυτόχρονα.

Αυτό το κεφάλαιο παρουσιάζει δυο πρόσθετους ελέγχους εκτέλεσης: υποθετική εκτέλεση (*If/Else*) και επανάληψη. Με αυτό τον τρόπο ελέγχετε πως εκτελείται το πρόγραμμα. Η υποθετική εκτέλεση χρησιμοποιεί συναρτήσεις και παραστάσεις για να ελέγξουν την τρέχουσα κατάσταση του κόσμου. Για παράδειγμα “Είναι το χρώμα του κεφαλιού του spiderRobot κόκκινο;” Προφανώς, αυτή είναι μια ερώτηση και μια συνάρτηση χρησιμοποιείται για να διατυπωθεί η ερώτηση. Συχνά ο έλεγχος μιας υπόθεσης απαιτεί να συγκριθούν δυο η περισσότερα αντικείμενα. Για παράδειγμα, μπορεί να ελέγξουμε εάν “η απόσταση του spiderRobot από το βράχο είναι μεγαλύτερη του ενός μέτρου”. Μια παράσταση χρησιμοποιείται για να συγκρίνει εάν η απόσταση μεταξύ δυο αντικειμένων είναι μεγαλύτερη του ενός μέτρου.

Η ενότητα 3-1 χρησιμοποιεί μικρά παραδείγματα για να δείξει πώς να χρησιμοποιείτε συναρτήσεις στο Alice. Επίσης παρέχονται απλές αριθμητικές πράξεις (άθροισμα, αφαίρεση, πολλαπλασιασμός, διαίρεση).

Η ενότητα 3-2 κάνει μια εισαγωγή στις υποθετικές εκτελέσεις στην φόρμα *If/Else*. Μια *If/Else* εντολή συνεπάγεται μιας απόφασης βασισμένης σε μια τρέχουσα κατάσταση του κόσμου. Παρουσιάζεται μια απλή δομή επαναληπτικού ελέγχου, στην φόρμα μιας *Loop.* Μια δήλωση *Loop* επαναλαμβάνει για συγκεκριμένες φορές την εκτέλεση ενός μέρους ενός προγράμματος.

3-1 Ενσωματωμένες συναρτήσεις και παραστάσεις

Όπως ξέρετε, οι πληροφορίες για τον κόσμο και τα αντικείμενά του αποθηκεύονται στις ιδιότητες. Μπορούμε να χρησιμοποιήσουμε μια συνάρτηση για να διατυπώσουμε ερωτήσεις γι' αυτές τις ιδιότητες. Επίσης, μπορούμε να χρησιμοποιήσουμε αριθμητικές πράξεις στις τιμές αυτών των ιδιοτήτων χρησιμοποιώντας τις παραστάσεις. Αυτή η ενότητα εστιάζει στο πώς να χρησιμοποιήσετε τις ιδιότητες, τις συναρτήσεις και τις παραστάσεις για να επιτρέψετε σε ένα πρόγραμμα να λειτουργήσει με πληροφορίες του κόσμου και των αντικειμένων.

Ενσωματωμένες συναρτήσεις

Δεν είναι όλες οι ιδιότητες των αντικειμένων διαθέσιμες στην λίστα ιδιοτήτων. Μόνο οι συχνά χρησιμοποιούμενες ιδιότητες υπάρχουν (για παράδειγμα *color,* και *opacity*). Άλλες ιδιότητες των αντικειμένων όπως (*height*, *width* και *position*) μπορούν να γίνουν γνωστές ρωτώντας το Alice. Το σύστημα Alice παρέχει ένα σύνολο ενσωματωμένων συναρτήσεων – εντολές που μπορείτε να χρησιμοποιήσετε για να μάθετε τις ιδιότητες των αντικειμένων και τις σχέσεις του ενός αντικειμένου με το άλλο. Επίσης λειτουργικές ερωτήσεις μπορούν να γίνουν για τον κόσμο-για θέματα σχετικά με την θέση του ποντικιού και για μαθηματικές πράξεις.

Για να καταλάβετε καλύτερα την χρησιμότητα των συναρτήσεων, ας δούμε ξανά τον κόσμο του ρομπότ, που φαίνεται στην εικόνα 3-1-1. Κοιτάτε στην οθόνη και εξηγείτε τι βλέπετε.

ΕΙΚΟΝΑ 3-1-1. Αρχική σκηνή του κόσμου First Encounter

Για παράδειγμα, βλέποντας τη σκηνή νομίζετε ότι το spiderRobot είναι πιο κοντά στην κάμερα απ' ότι οι βράχοι. Ένας καλλιτέχνης το ονομάζει αυτό προοπτική. Φυσικά είναι δύσκολο να καταλάβετε πόσο μακριά είναι το ρομπότ από τους βράχους. Όμοια είναι δύσκολο να καταλάβετε πόσο μακριά είναι το spiderRobot από το lunarLander. Η γωνία της κάμερας δεν παρέχει αρκετή πληροφορία.

Εδώ μπαίνουν οι συναρτήσεις. Οι συναρτήσεις μπορούν να χρησιμοποιηθούν για να αντλήσετε πληροφορίες που χρειάζεστε. Το Alice παρέχει αρκετές συναρτήσεις που μπορούν να χρησιμοποιηθούν για κάθε αντικείμενο σε ένα κόσμο. Αυτές ονομάζονται *ενσωματωμένες συναρτήσεις*. Για να δείτε μια λίστα ενσωματωμένων συναρτήσεων για ένα αντικείμενο σε ένα κόσμο, επιλέξτε το αντικείμενο στο δέντρο αντικειμένων και δείτε τις συναρτήσεις στην περιοχή των λεπτομερειών. Σε αυτό το παράδειγμα, μπορείτε να δείτε μια λίστα ενσωματωμένων συναρτήσεων για το spiderRobot, επιλέγοντας το spiderRobot στο δέντρο αντικειμένων και μετά την καρτέλα functions όπως φαίνεται στην εικόνα 3-1-2.

Στα δεξιά της λίστας των συναρτήσεων υπάρχει μια μπάρα που σας επιτρέπει να κατεβάσετε προς τα κάτω το παράθυρο για να δείτε όλες τις διαθέσιμες ενσωματωμένες συναρτήσεις του spiderRobot. Οι ενσωματωμένες συναρτήσεις χωρίζονται στις υποκατηγορίες:

Proximity (εγγύτητα) - πόσο κοντά είναι το αντικείμενο με ένα άλλο αντικείμενο στον κόσμο (όπως απόσταση από, απόσταση με). (Οι συναρτήσεις της εικόνας 3-1-2 είναι συναρτήσεις εγγύτητας).

Size (μέγεθος) - διαστάσεις όπως ύψος, πλάτος και βάθος και πως αυτές συγκρίνονται με τις διαστάσεις άλλων αντικειμένων στον κόσμο.

Spatial Relation (αναφορά σε σχέση με το χώρο) - προσανατολισμός σε σχέση με ένα άλλο αντικείμενο του κόσμου (δεξιά από, αριστερά από)

Point of view (άποψη) - θέση στον κόσμο

Other (άλλα) - διάφορα αντικείμενα όπως το όνομα ενός υπομέρους ενός αντικειμένου.

ΕΙΚΟΝΑ 3-1-2. Συναρτήσεις για ένα αντικείμενο

Στις καθημερινές μας συζητήσεις, όταν τίθεται μια ερώτηση, περιμένουμε να πάρουμε μια απάντηση. Στο Alice, η απάντηση είναι η τιμή της ιδιότητας για την οποία γίνεται η ερώτηση. Τι είδους τιμές μπορείτε να περιμένετε; Αυτό εξαρτάται από το είδος της ερώτησης. Για παράδειγμα οι συναρτήσεις proximity του spiderRobot is within threshold of object (*είναι μέσα σε μια δεδομένη απόσταση*) και θα επιστρέψει true ή false (αληθές ή ψευδές). Αλλά η συνάρτηση *distance to* (*απόσταση από*) επιστρέφει ένα αριθμό (*distance to* σημαίνει η απόσταση σε μέτρα από ένα άλλο αντικείμενο).

Στο Alice οι τιμές μπορεί να είναι πολλών διαφορετικών τύπων. Τέσσερις συχνοί τύποι τιμών υπάρχουν:

Number (αριθμός) (για παράδειγμα, 5 ή -19,5)

Boolean (λογικές τιμές) (true, false)

String (ακολουθία χαρακτήρων) (για παράδειγμα, "*Γεια σου κόσμε*")

Object (αντικείμενο) (για παράδειγμα *spiderRobot*)

Στο πρώτο πρόγραμμα του First Encounter, το spiderRobot θέλει να ρίξει μια πιο κοντινή ματιά στον εξωγήινο. Έτσι το spiderRobot προχωράει μπροστά ένα μέτρο καθώς και τα πόδια του κινούνται. Ο κώδικας επαναλαμβάνεται στην εικόνα 3-1-3.

Δεν ξέρουμε πόσο μακριά βρίσκεται το spiderRobot από τον βράχο που κρύβεται το alienOnWheels. Το ένα μέτρο απόσταση για την μπροστινή κίνηση είναι απλά τυχαίο. Το πρόβλημα είναι ότι δεν ξέρουμε ακριβώς πόσο μακριά να κινήσουμε το spiderRobot. Ένας τρόπος επίλυσης αυτού του προβλήματος είναι να κάνουμε πολλές δοκιμές μέχρι να βρούμε την απόσταση που λειτουργεί καλύτερα. Μια άλλη τεχνική για να βρούμε την απόσταση είναι να χρησιμοποιήσουμε μια συνάρτηση για να κάνουμε την ερώτηση: "Ποια είναι η απόσταση του spiderRobot από το βράχο; (όπου και κρύβεται ο alienOnWheels)". Η Alice θα επιστρέψει την απόσταση σε μέτρα. Μετά το spiderRobot μπορεί να κινηθεί για την συγκεκριμένη απόσταση.

Για να χρησιμοποιήσετε την συνάρτηση *distance to (απόσταση στο)*, απλά επιλέξτε το πλακίδιο της απόστασης και βάλτε το στον συντάκτη στην

περιοχή που πρωτίστως ήταν ένα μέτρο απόσταση. Μετά επιλέξτε τον βράχο ως το αντικείμενο-στόχο, όπως φαίνεται στην εικόνα 3-1-4.

ΕΙΚΟΝΑ 3-1-3. Κώδικας για να κινήσετε το spiderRobot μπροστά ένα μέτρο

ΕΙΚΟΝΑ 3-1-4. Βάζοντας την συνάρτηση *distance to* στον συντάκτη

Ο τελικός κώδικας είναι:

ΕΙΚΟΝΑ

Collision (σύγκρουση)

Αφού κάναμε μια τέτοια αλλαγή στο πρόγραμμα μας, το πρόγραμμα πρέπει να δοκιμαστεί. Σε αυτό το παράδειγμα, όταν το πρόγραμμα εκτελείται το spiderRobot περπατάει στο μέσο του βράχου. Αυτό ονομάζεται collision (σύγκρουση). Σε μερικά κινούμενα σχέδια, μια σύγκρουση είναι επιθυμητή. Σε αυτό το παράδειγμα, δεν θέλουμε το ρομπότ να συγκρούεται με το βράχο. Ο λόγος που συμβαίνει μια σύγκρουση είναι ότι *η distance to* μετριέται από το κέντρο τους ενός αντικειμένου στο κέντρο του άλλου. Σε αυτό το παράδειγμα, *η distance to* μετριέται από το κέντρο του spiderRobot στο κέντρο του βράχου, όπως φαίνεται στην εικόνα 3-1-5.

ΕΙΚΟΝΑ 3-1-5. Η συνάρτηση *distance to* μετριέται κέντρο με κέντρο

Expressions (παραστάσεις)

Πως μπορεί να αποφευχθεί η σύγκρουση μεταξύ των δύο αντικειμένων; Ένας τρόπος είναι να ρυθμίσετε την απόσταση ώστε το spiderRobot να μην κινηθεί για ολόκληρη την απόσταση. Η ρύθμιση της απόστασης απαιτεί την χρήση αριθμητικής παράστασης για να αφαιρεθεί μια μικρή ποσότητα της απόστασης. Σε αυτό το παράδειγμα, ο βράχος είναι μεγαλύτερος από το spiderRobot. Εάν το πλάτος του βράχου αφαιρεθεί από την απόσταση των δυο αντικειμένων, η σύγκρουση θα αποφευχθεί.

Το Alice παρέχει μαθηματικές πράξεις για τις γνωστές μαθηματικές παραστάσεις πρόσθεση (+), αφαίρεση (-), πολλαπλασιασμός (*), διαίρεση (/). Μια μαθηματική παράσταση δημιουργείται επιλέγοντας το *math (μαθηματικά)* από το popup μενού των αριθμητικών τιμών μιας εντολής. Για παράδειγμα, για να χρησιμοποιήσετε μια μαθηματική παράσταση για να ρυθμίσετε την απόσταση που κάνει το spiderRobot, κάντε κλικ στο κάτω βελάκι στα δεξιά της *distance to* που βρίσκεται μέσα στην πληροφορία *move*, μετά επιλέξτε *math* → spiderRobot's distance to rock -1, όπως φαίνεται στην εικόνα 3-1-6.

ΕΙΚΟΝΑ 3-1-6. Χρήση μιας μαθηματικής παράστασης

Η εντολή αφαιρεί ένα μέτρο από την απόσταση. Στην πραγματικότητα, θέλουμε να αφαιρέσουμε το πλάτος του βράχου από την απόσταση (όχι ένα

μέτρο). Αλλά έπρεπε να διαλέξουμε κάτι από το μενού, έτσι επιλέξαμε το ένα μέτρο. Τώρα θα αντικαταστήσουμε το ένα μέτρο με το πλάτος του βράχου, όπως φαίνεται στην εικόνα 3-1-7.

Τώρα το spiderRobot κινείται μπροστά και φτάνει κοντά στο βράχο χωρίς σύγκρουση. Ο τελικός κώδικας φαίνεται στην εικόνα 3-1-8.

ΕΙΚΟΝΑ 3-1-7. Αντικατάσταση του ενός μέτρου με το width (πλάτος) του βράχου

ΕΙΚΟΝΑ 3-1-8. Κώδικας για να κινηθεί το ρομπότ μπροστά χωρίς σύγκρουση

Τεχνική σημείωση: Η παράσταση της απόστασης που χρησιμοποιείται στην εντολή *move* λειτουργεί πολύ καλά. Πρέπει να σημειώσουμε, ότι αυτή η παράσταση είναι σκόπιμα μια αρχική μορφή μιας σύγκρουσης. Για να γίνουμε τεχνικά πιο σωστοί, μια σύγκρουση θα υπολογιζόταν αφαιρώντας το άθροισμα του μισού πλάτους του ρομπότ και του μισού πλάτους του βράχου από την συνολική απόσταση, όπως φαίνεται παρακάτω στον κώδικα.

ΕΙΚΟΝΑ

3-2 Απλές δομές ελέγχου

Για να δημιουργήσουμε ένα πρόγραμμα που να κάνει κάτι παραπάνω από το να εκτελεί μερικές οδηγίες, θα πρέπει να χρησιμοποιήσουμε δομές ελέγχου. Μια δομή ελέγχου είναι μια εντολή προγραμματισμού που σας επιτρέπει να ελέγξετε την σειρά με την οποία θα εκτελεστούν οι εντολές. Έχετε ήδη δει δυο δομές ελέγχου: do together (*κάνε ταυτόχρονα*) και do in order (*κάνε διαδοχικά*). Αυτή η ενότητα εισάγει περισσότερες δομές ελέγχου: *υποθετική εκτέλεση* και *επανάληψη*. Η υποθετική εκτέλεση είναι όταν ελέγχονται οι συνθήκες και σύμφωνα με αυτούς παίρνεται μια απόφαση για το αν θα εκτελεστεί ή όχι ένα συγκεκριμένο τμήμα ενός κώδικα. Η επανάληψη είναι όταν ένα τμήμα κώδικα επαναλαμβάνεται συγκεκριμένες φορές.

Υποθετική εκτέλεση

Μια υποθετική εκτέλεση βασίζεται σε μια *απόφαση*. Μερικές φορές η ζωή είναι "Μια απόφαση μετά την άλλη". Εάν το γρασίδι δεν είναι υγρό, μπορούμε να το κουρέψουμε. Εάν το πλυντήριο είναι γεμάτο από βρώμικα πιάτα, πρέπει να το βάλουμε να πλένει. Εάν βγάλουμε το σκύλο έξω για βόλτα, πρέπει να του βάλουμε κολάρο. Και ο προγραμματισμός συχνά απαιτεί αποφάσεις. Οι αποφάσεις είναι χρήσιμες όταν γράφετε προγράμματα και ορισμένες εντολές εκτελούνται μόνο κάτω υπό καθορισμένους όρους.

Όταν παίρνεται μια απόφαση, γίνεται μια ερώτηση για μια ισχύουσα κατάσταση του κόσμου. Για παράδειγμα, "Είναι το διαστημόπλοιο ορατό;" ή "είναι το χρώμα του καπέλου κόκκινο;" Η απάντηση είναι true ή false (σωστό ή λάθος). Οι τιμές true ή false είναι γνωστές ως λογικές τιμές και ονομάστηκαν έτσι από τον 19^{ο} αιώνα από τον Άγγλο μαθηματικό George Boole, ο οποίος

ήταν ο πρώτος που ενδιαφέρθηκε για παραστάσεις που μπορούν να αποτιμηθούν μόνο με αυτές τις δυο τιμές.

Στο Alice, μια εντολή *If/Else* (εάν/αλλιώς) χρησιμοποιείται ως δομή ελέγχου υποθετικής εκτέλεσης. (Πιο απλά αναφερόμαστε στην εντολή *If/Else* ως εντολή *If*). Η εικόνα 3-2-1 παρουσιάζει την διαδικασία μιας εντολής *If*. Η εντολή ελέγχει για να δει αν μια υπόθεση είναι αληθής ή όχι. Εάν η υπόθεση είναι αληθής, ένα τμήμα του προγράμματος εκτελείται. Εάν η υπόθεση είναι ψευδής, ένα άλλο κομμάτι του προγράμματος εκτελείται. Το μέρος του προγράμματος που εκτελείται ή όχι εξαρτάται από την τιμή της υπόθεσης.

ΕΙΚΟΝΑ 3-2-1. Πως επεξεργάζεται μια εντολή *If*

Μια εντολή *If* δημιουργείται στο Alice τοποθετώντας το πλακίδιο *If/Else* στον συντάκτη. Ένα popup πλαίσιο σας επιτρέπει να επιλέξετε την αρχική κατάσταση (αληθές, ψευδές), όπως φαίνεται στην εικόνα 3-2-2.

Το πράσινο χρώμα αυτού του μπλοκ είναι ένα οπτικό στοιχείο ότι μια εντολή *If* χρησιμοποιείται στο πρόγραμμα. Η εντολή *If* έχει δύο μέρη (ένα μέρος *If* και ένα μέρος *Else*). Παρόλο που η αρχική κατάσταση (*true, false*) επιλέγεται από το popup μενού, το Alice σας επιτρέπει να δημιουργήσετε τη δική σας υποθετική έκφραση στο πάνω μέρος του πλακιδίου της υπόθεσης. Μια υποθετική έκφραση είναι μια συνάρτηση η οποία θα αποτιμήσει σε *true* ή *false*. Εάν η απάντηση είναι *true,* εκτελείται το μέρος της *If* και το μέρος της *Else* παραβλέπεται. Αλλιώς αν η απάντηση είναι *false* το μέρος της *If* παραβλέπεται και εκτελείται το μέρος της *Else.*

ΕΙΚΟΝΑ 3-2-2. Χρήση μιας εντολής *If*

Ένα απλό παράδειγμα

Για να υλοποιήσουμε την χρησιμότητα μιας εντολής *If* , ας συνεχίσουμε με το παράδειγμα του spiderRobot. Όπως αναφέρθηκε αρχικά, η αίσθηση της προοπτικής μπορεί να είναι παραπλανητική για τις αποστάσεις και τα μεγέθη των αντικειμένων στον κόσμο. Το spiderRobot βαδίζει προς το βράχο για να ρίξει μια πιο κοντινή ματιά στο alienOnWheels, αλλά το alienOnWheels κρύβεται (εκτός οπτικού πεδίου). Από αυτή την οπτική γωνία, όπως φαίνεται στην εικόνα 3-2-3, ο βράχος φαίνεται ψηλότερος από το spiderRobot αλλά δεν μπορούμε να είμαστε σίγουροι. Έτσι δεν μπορούμε να πούμε εάν το spiderRobot μπορεί να δει πάνω από το βράχο. Εάν το spiderRobot είναι πιο κοντό από το βράχο, μπορεί να σηκώσει το λαιμό του για να δει πάνω από το βράχο. Αλλιώς μπορεί να δει το alienOnWheels χωρίς κάποια πρόσθετη κίνηση.

ΕΙΚΟΝΑ 3-2-3. Το spiderRobot κινείται προς το βράχο, αλλά το alienOnWheels κρύβεται

Το πρόβλημα είναι, πως γνωρίζουμε ποιο είναι ψηλότερο (το spiderRobot ή ο βράχος;). Μια λύση είναι να χρησιμοποιήσουμε μια εντολή *If* με την ολοκληρωμένη ερώτηση *is shorter than (είναι πιο κοντό από).* Η διάταξη σκηνών είναι:

If spiderRobot is shorter than rock
  Do in order
    spiderRobot neck move up
    spiderRobot neck move down
Else
    Do nothing

Εάν το ρομπότ είναι πιο κοντό από τον βράχο
  Κάνε διαδοχικά
    ο λαιμός του ρομπότ κινείται επάνω
    ο λαιμός του ρομπότ κινείται κάτω
Αλλιώς
    Μην κάνεις τίποτα

Η μετάφραση του παραπάνω πίνακα σε κώδικα προγράμματος είναι απλή. Πρώτα δημιουργείται μια δήλωση *If* τοποθετώντας το πλακίδιο *If/Else* στον συντάκτη και επιλέγοντας το *true* ως προκαθορισμένη κατάσταση, όπως παρουσιάστηκε προηγουμένως στην εικόνα 3-2-2. Μετά η συνάρτηση του spiderRobot *is shorter than* τοποθετείται στον συντάκτη εκεί που προηγουμένως ήταν το *true* και ο βράχος επιλέγεται ως αντικείμενο σύγκρισης όπως φαίνεται στην εικόνα 3-2-4.

ΕΙΚΟΝΑ 3-2-4. Χρήση μιας συνάρτησης για τον έλεγχο μιας υπόθεσης σε μια *If/Else*

Μετέπειτα, εντολές *move* του λαιμού του ρομπότ (spiderRobot.neck) προστίθενται σε ένα μπλοκ *do in order* για το *If* μέρος της *If/Else.* Το *Else* μέρος της *If/Else* παραμένει *do nothing.* Ο τελικός κώδικας φαίνεται παρακάτω.

ΕΙΚΟΝΑ

Όταν εκτελείται αυτός ο κώδικας, μια συνάρτηση χρησιμοποιείται για να διατυπωθεί η ερώτηση: “Είναι το spiderRobot κοντύτερο από τον βράχο;” Η απάντηση είναι είτε *true*, είτε *false*. Μετά παίρνεται μια απόφαση σε σχέση με την απάντηση. Εάν η απάντηση είναι *true* ο λαιμός του spiderRobot κινείται πάνω και κάτω (για να δει πάνω από τον βράχο). Αλλιώς, επιλέγεται ο κώδικας του μέρους *Else* και δεν γίνεται τίποτα. Δεν κινείται ο λαιμός του spiderRobot. Θα μπορούσαμε να γράψουμε εντολές ώστε σε αυτή την περίπτωση το spiderRobot να κάνει κάτι διαφορετικό- π.χ. να γυρίσει το κεφάλι- αλλά θέλαμε να δείξουμε ότι το *do nothing* είναι μια αποδεκτή πράξη.

Σχεσιακοί τελεστές

Στο παράδειγμα παραπάνω, χρησιμοποιήσαμε μια ενσωματωμένη συνάρτηση για να συγκρίνουμε τα ύψη του spiderRobot και του βράχου. Συχνά χρησιμοποιούμε τις ενσωματωμένες συναρτήσεις για να ελέγξουμε μια κατάσταση σε μια εντολή *If.* Ωστόσο μερικές φορές θέλουμε να γράψουμε την δική μας υπόθεση χρησιμοποιώντας ένα σχεσιακό τελεστή (operator). Το Alice παρέχει έξι σχεσιακούς τελεστές που βρίσκονται όλες σε μια ομάδα στην κατηγορία *math* των ενσωματωμένων συναρτήσεων του κόσμου, όπως φαίνεται στην εικόνα 3-2-5. Οι τελεστές λειτουργούν το ίδιο όπως στα μαθηματικά. Παρακάτω εξηγούνται όλοι οι τελεστές. Για παράδειγμα το “==” σημαίνει “είναι ίσο με” και το “!=” σημαίνει “είναι διαφορετικό από”.

ΕΙΚΟΝΑ 3-2-5. Σχεσιακοί τελεστές

Σαν ένα παράδειγμα χρήσης ενός σχεσιακού τελεστή για να γράψετε την δική σας υπόθεση, ας υποθέσουμε ότι γνωρίζουμε ότι το ύψος του βράχου είναι δυο μέτρα. Θα μπορούσαμε να γράψουμε μια εντολή *If* που ελέγχει το ύψος του ρομπότ σε σχέση με τα δύο μέτρα. Έτσι έχουμε:

```
If the spiderRobot's height is less than 2 meters
  Do in order
    spiderRobot's neck move up
    spiderRobot's neck move down
```

```
Εάν το ύψος του ρομπότ είναι λιγότερο από δυο μέτρα
  Κάνε διαδοχικά
    ο λαιμός του ρομπότ κινείται επάνω
    ο λαιμός του ρομπότ κινείται κάτω
```

Η υπόθεση του *If* είναι: “Το ύψος του ρομπότ είναι λιγότερο από δυο μέτρα.” Η υπόθεση πρέπει να γραφεί ως παράσταση λογικής τιμής. Ας γράψουμε μια εντολή *If* για τον παραπάνω πίνακα, δημιουργώντας την δική μας λογική παράσταση με χρήση ενός σχεσιακού τελεστή.

Πρώτα τοποθετήστε το *If* στον συντάκτη, όπως στα προηγούμενα παραδείγματα. Μετά κάντε τα δυο ακόλουθα βήματα.

1. Τοποθετήστε την συνάρτηση του κόσμου “*less than (μικρότερο από a<b)*” επάνω στην λέξη *true* της εντολής *If.* Ένα popup μενού σας επιτρέπει να επιλέξετε τιμές για το a και το b, όπως φαίνεται στην εικόνα 3-2-6.

ΕΙΚΟΝΑ 3-2-6. Δημιουργία της δικής σας λογικής έκφρασης

Από το popup μενού επιλέξτε 1 για το a και 2 για το b. Το αποτέλεσμα είναι κάπως έτσι:

ΕΙΚΟΝΑ

Στην πραγματικότητα, θέλαμε αντί για 1 στην θέση του a να βάλουμε το ύψος του ρομπότ, αλλά επιλέξαμε το 1 για να κρατήσουμε την μεταβλητή και να ολοκληρωθεί το πρώτο βήμα. Θα το φτιάξουμε στο επόμενο βήμα.

2. Τώρα, παίρνουμε την συνάρτηση *height* του ρομπότ για να αντικαταστήσουμε το ένα. Προσθέτουμε στο μέρος της *If,* ένα μπλοκ *do in order* και εντολές για να κάνουμε το λαιμό του spiderRobot να κινηθεί πάνω και κάτω. Το μέρος *Else* της εντολής παραμένει *do nothing.* Εάν η υπόθεση (το *height* του spiderRobot είναι μικρότερο του δύο) είναι false, δεν θα γίνει τίποτα. Ο τελικός κώδικας φαίνεται παρακάτω.

ΕΙΚΟΝΑ

Η ανάγκη της επανάληψης

Στο παράδειγμα της ενότητας 3-1, το spiderRobot κινείται προς το βράχο (μια παράσταση υπολογίζει την απόσταση-αρκετά μέτρα), και τα πόδια του spiderRobot εξομοιώνουν ένα περπάτημα, μόνο μια φορά. (Ο κώδικας φαίνεται στην εικόνα 3-2-7). Όταν εκτελείται το πρόγραμμα, το περπάτημα δεν είναι ρεαλιστικό γιατί το spiderRobot κινείται μπροστά για μια σχετικά μεγάλη απόσταση, αλλά τα πόδια κάνουν ένα μόνο βήμα.

ΕΙΚΟΝΑ 3-2-7. Το spiderRobot κινείται μπροστά αρκετά μέτρα αλλά τα πόδια κάνουν μόνο ένα βήμα

Θα ήταν πιο ρεαλιστικό εάν τα πόδια του spiderRobot έκαναν ένα βήμα για κάθε μέτρο της απόστασης που θα έκανε το spiderRobot. Διαφορετικά, εάν το spiderRobot κινούταν τρία μέτρα μπροστά, τότε το περπάτημα θα συνέβαινε τρεις φορές.

Μπορούμε να διορθώσουμε τον κώδικα του προγράμματος για να κινούμε ένα μέτρο μπροστά το spiderRobot και τα πόδια να κάνουν ένα βήμα. Έτσι ο κώδικας θα είναι αυτός της εικόνας 3-2-8.

ΕΙΚΟΝΑ 3-2-8. Το spiderRobot κινείται ένα μέτρο μπροστά και τα πόδια κινούνται μια φορά

Το πρόβλημα είναι ότι τώρα το spiderRobot κινείται μπροστά ένα μέτρο. Εμείς θέλουμε να κινηθεί αρκετά μέτρα (3 ή 4). Θα ήταν κάπως βαρετό να δημιουργήσουμε αυτή την πληροφορία 3 ή 4 φορές. Επίσης, σκεφτείτε τι θα γινόταν αν θέλαμε το spiderRobot να περπατήσει 20 μέτρα.

Μια πιθανή λύση είναι να χρησιμοποιήσουμε το clipboard. Το clipboard είναι ένα εργαλείο που αντιγράφει ένα σύνολο εντολών από ένα μέρος στο άλλο στον συντάκτη. (Δείτε στο παράρτημα Β για εντολές πώς να χρησιμοποιήσετε το clipboard). Και σε αυτή την περίπτωση όμως είναι πάλι βαρετό.

(Loop) Επανάληψη με βρόγχο

Αυτό που θέλουμε είναι ένας τρόπος να κάνουμε την δουλειά μας πιο εύκολη χρησιμοποιώντας δομή επαναληπτικού ελέγχου, που ονομάζεται εντολή *με βρόγχο (loop).* Μια εντολή με βρόγχο βρίσκεται σε πολλές γλώσσες προγραμματισμού και είναι ένας απλός και εύκολος τρόπος να επαναλάβουμε μια ενέργεια για ένα συγκεκριμένο αριθμό φορών. Για να δημιουργήσετε ένα βρόγχο σε ένα πρόγραμμα, τοποθετήστε το πλακίδιο *Loop* στον συντάκτη, (τοποθετήστε το στον συντάκτη, πριν από το μπλοκ *do together*) όπως φαίνεται στην εικόνα 3-2-9.

ΕΙΚΟΝΑ 3-2-9. Τοποθέτηση μιας εντολής *Loop* στον συντάκτη

Όταν τοποθετηθεί το πλακίδιο του βρόγχου στον συντάκτη, ένα popup μενού σας παρέχει τη δυνατότητα για το πλήθος των επαναλήψεων (τον αριθμό των φορών που θα εκτελεστεί η εντολή βρόγχου). Το Alice χρησιμοποιεί τον όρο “τέλος” για να περιγράψει το τέλος των επαναλήψεων. Επιλέγουμε το *other* (*άλλο*) και μετά εισάγουμε τον αριθμό τρία, όπως φαίνεται στην εικόνα 3-2-10. Σημειώστε ότι ο βρόγχος μπορεί να εκτελεστεί μόνο για ακέραιους αριθμούς.

ΕΙΚΟΝΑ 3-2-10. Επιλογή πλήθους για μια εντολή Loop

Τέλος βάλτε το μπλοκ *do together* μέσα στην εντολή *Loop*, όπως φαίνεται στην εικόνα 3-2-11.

ΕΙΚΟΝΑ 3-2-11. Τοποθέτηση του μπλοκ *do together* μέσα στην εντολή *Loop*

Η τελική εντολή βρόγχου φαίνεται στην εικόνα 3-2-12. Το μπλοκ με χρώμα μπλε-πράσινο εμπεριέχει όλες τις πληροφορίες της επαναληπτικής δομής. Το σχόλιο έχει ενημερωθεί για να δείξει ότι τα βήματα που κάνει το spiderRobot θα επαναληφθούν τρεις φορές.

ΕΙΚΟΝΑ 3-2-12. Η τελική εντολή *Loop*

Μυστικά και Τεχνικές 3

Σχεδιασμός όψης και αίσθησης

Η φράση "όψη και αίσθηση" περιγράφει την εμφάνιση των αντικειμένων. Συχνά περιγράφουμε την όψη και την αίσθηση σε σχέση με τις ιδιότητες, όπως color (χρώμα) και texture (σύσταση). Σαν παράδειγμα, μπορούμε να πούμε ότι ένα μπλουζάκι είναι "κίτρινο και έχει απαλή, βελούδινη υφή". Τα θέματα σε αυτή την ενότητα παρέχουν πληροφορίες για το πώς να τροποποιήσετε την όψη και την αίσθηση ενός κόσμου και των αντικειμένων του.

Χάρτες υφής

Τα αντικείμενα που υπάρχουν στο Alice καλύπτονται με χάρτες υφής και να παρέχουν μια αίσθηση πραγματικότητας. Για παράδειγμα, δείτε το σκηνικό με το πιάτο (Kitchen) στο τραπέζι (Furniture on CD or Web gallery) στην εικόνα Τ-3-1.

ΕΙΚΟΝΑ Τ-3-1. Ένα πιάτο πάνω στο τραπέζι

Ένας χάρτης υφής που ονομάζεται *ground.GrassTexture* καλύπτει την επιφάνεια του γρασιδιού (ground) και ένας χάρτης υφής που ονομάζεται *plate.TextureMap* καλύπτει το πιάτο (plate), όπως φαίνεται στην εικόνα Τ-3-2.

ΕΙΚΟΝΑ Τ-3-2. Οι χάρτες υφής χρησιμοποιούνται ως περίβλημα

Ένα αρχείο γραφικών (.gif, .bmp, .jpg, .tif) μπορεί να χρησιμοποιηθεί για να δώσει σε ένα αντικείμενο μια διαφορετική όψη. Το Internet είναι ο κατάλληλος χώρος για να ψάξετε εικόνες. Για παράδειγμα, ας αλλάξουμε την εμφάνιση του πιάτου για να μοιάζει σαν κουλουράκι. Απαιτούνται δύο βήματα. Το πρώτο βήμα είναι να εισάγουμε ένα χάρτη υφής που θέλουμε να

χρησιμοποιήσουμε γι αυτό το αντικείμενο. Σε αυτό το παράδειγμα, επιλέξαμε το αντικείμενο plate (πιάτο). Κάνουμε κλικ στο κουμπί import texture map (εισαγωγή χάρτη υφής) και μετά επιλέγουμε το cookie.gif (κουλουράκι). (Το cookie.gif δεν είναι αντικείμενο του Alice. Δημιουργήσαμε το αντικείμενο χρησιμοποιώντας πρόγραμμα γραφικών). Η εικόνα Τ-3-3 παρουσιάζει το βήμα εισαγωγής. Το επόμενο βήμα είναι να καθορίσουμε την ιδιότητα του περιβλήματος να χρησιμοποιήσει το νέο χάρτη υφής, όπως φαίνεται στην εικόνα Τ-3-4. Το αποτέλεσμα φαίνεται στην εικόνα Τ-3-5.

ΕΙΚΟΝΑ Τ-3-3. Εισαγωγή ενός χάρτη υφής

ΕΙΚΟΝΑ Τ-3-4. Αλλαγή του χάρτη υφής για το περίβλημα

ΕΙΚΟΝΑ Τ-3-5. Πιάτο κουλουράκι

Ειδικά εφέ: fog (ομίχλη)

Σε μερικούς κόσμους μπορεί να θέλετε να δημιουργήσετε μια ατμόσφαιρα με ομίχλη. Σκεφτείτε την σκηνή στην εικόνα Τ-3-6. Ένας ιππότης (μεσαιωνικός) ψάχνει για ένα δράκο (μεσαιωνικός) στο δάσος (trees στον φάκελο Nature). Θέλουμε να δημιουργήσουμε την εντύπωση ότι ο δράκος κρύβεται. Στις περισσότερες ιστορίες που αναμειγνύονται δράκοι, ο καιρός είναι ζοφερός και σκοτεινός. Λίγο ομίχλη θα έκανε το έργο του ιππότη (εύρεση του δράκου) πιο προκλητική.

ΕΙΚΟΝΑ Τ-3-6. Χωρίς ομίχλη

Για να προσθέσετε fog (ομίχλη), κάντε κλικ στον κόσμο (*World*) στο δέντρο αντικειμένων και επιλέξτε properties, όπως φαίνεται στην εικόνα Τ-3-7. Μετά κάντε κλικ στην δεξιά εικόνα *fogStyle* (της ομίχλης) και επιλέξτε density (πυκνότητα). Για να ρυθμίσετε την πυκνότητα της ομίχλης, κάντε κλικ στο *fogDensity* και ρυθμίστε την τιμή της για να επιτύχετε το επιθυμητό αποτέλεσμα. Μια μεγαλύτερη τιμή πυκνότητας παράγει πιο πυκνή ομίχλη.

ΕΙΚΟΝΑ Τ-3-7. Οι ιδιότητες *fogStyle* και *fogDensity* χρησιμοποιούνται για να δημιουργήσουν ένα ομιχλώδες τοπίο.

Ασκήσεις

3-1 Ασκήσεις

1. Το ρομπότ στο διαστημόπλοιο

Χρησιμοποιήστε τον κόσμο First Encounter για να αναδημιουργήσετε τον κώδικα όπως περιγράφηκε στο κεφάλαιο 3 στην ενότητα 1. Αφού το spiderRobot κινηθεί προς το βράχο, κάντε το spiderRobot να αντικρίσει το lunarLander και μετά να κινηθεί μπροστά έως την μισή απόσταση που έχει από το lunarLander.

2. Σκύλος στην πυροσβεστική κάνουλα

Δημιουργήστε ένα κόσμο με ένα σκύλο (wolf από Animals) και μια πυροσβεστική αντλία (City), όπως φαίνεται στην εικόνα παρακάτω. Γράψτε μια πληροφορία που δημιουργεί μια συνάρτηση *distance to* και μια μαθηματική παράσταση για να κινήσει το σκύλο στην αντλία. Ο σκύλος πρέπει να σταματήσει λίγο πριν γίνει η σύγκρουση με την αντλία.

ΕΙΚΟΝΑ

3.Αναπήδημα

Δημιουργήστε ένα κόσμο με ένα κιβώτιο (Objects) και ένα καγκουρό (Animals). Γράψτε ένα πρόγραμμα για να κάνετε το καγκουρό να αναπηδήσει πάνω στο κιβώτιο, γυρίζοντας τα πόδια του καγκουρό πίσω και μπροστά για να φαίνεται σαν χοροπήδημα. Χρησιμοποιήστε την συνάρτηση *height* (ύψους) για να κάνετε τις κινήσεις μπροστά και πάνω.

ΕΙΚΟΝΑ

4.Αλμα βόλεϊ

Δημιουργήστε ένα νέο κόσμο με ένα δίχτυ του βόλεϊ, μια μπάλα (Sports), το κορίτσι παγοδρόμο και ένα άλλο κορίτσι, όπως φαίνεται παρακάτω. Κάθε άτομο στον κόσμο είναι πιθανό να έχει διαφορετικό ύψος και αθλητική ικανότητα. Ας υποθέσουμε ότι κάθε άτομο μπορεί να πηδήξει 1/4 του ύψους του ώστε να χτυπήσει την μπάλα. Γράψτε οδηγίες για να κάνετε τα άτομα να πηδήξει αυτή την απόσταση και μετά να κατέβει κάτω την ίδια απόσταση. Καλέστε την ενσωματωμένη συνάρτηση ύψους *height* και χρησιμοποιήστε μια παράσταση για να υπολογίσετε την απόσταση που πρέπει να κινηθεί πάνω και κάτω το κάθε άτομο.

ΕΙΚΟΝΑ

3-2 Ασκήσεις

5.Κίνηση του ρομπότ

Στην ενότητα 3-2, ο κώδικας του προγράμματος πραγματοποιούσε μια επανάληψη της κίνησης του spiderRobot τρεις φορές. Αναδημιουργήστε το πρόγραμμα με ένα πλήθος 2,4 και 5. Ποιο πλήθος λειτουργεί καλύτερα; Γιατί;

6.Το αερόπλοιο και ο δράκος

Δημιουργήστε μια σκηνή όπως φαίνεται παρακάτω με ένα αερόπλοιο (Vehicle) και ένα δράκο (Medieval). Σε αυτή τη σκηνή, ο δράκος έχει βρει ένα αερόπλοιο και είναι περίεργος. Ο δράκος πετάει γύρω από το αερόπλοιο για να το ελέγξει. Γράψτε ένα πρόγραμμα για να κάνετε το δράκο να κινείται προς το αερόπλοιο και μετά να πετάει γύρω του lengthwise (κατά μήκος) τρεις φορές.

ΕΙΚΟΝΑ

7. Χιονάνθρωπος σε σκαμνί

Αυτή η άσκηση χρησιμοποιεί ένα μια αριθμητική συνάρτηση ως πλήθος σε ένα Loop (επαναληπτικό βρόγχο). Δημιουργήστε ένα κόσμο με ένα χιονάνθρωπο και ένα σκαμνί, όπως φαίνεται παρακάτω. Χρησιμοποιήστε ένα βρόγχο για να κάνετε τον χιονάνθρωπο (People) να κινείται προς το σκαμνί (Kitchen) ένα μέτρο κάθε φορά. Χρησιμοποιήστε μια συνάρτηση απόστασης *distance to* για να ανακαλύψετε πόσες φορές θα εκτελεστεί ο βρόγχος. (Η συνάρτηση *distance to* μπορεί να επιστρέψει μια δεκαδική απόσταση όπως 3,75 μέτρα. Η εντολή Loop στρογγυλοποιεί το αποτέλεσμα σε δεκαδικό π.χ. 3 μέτρα και έτσι ο βρόγχος επαναλαμβάνεται τρεις φορές.) Σας συνιστούμε να τεστάρετε την λύση σας κινώντας τον χιονάνθρωπο και το σκαμνί σε διαφορετικές θέσεις στην οθόνη και εκτελώντας το πρόγραμμα σε κάθε αλλαγή, για να δείτε αν δουλεύει ανεξαρτήτως της θέσης των αντικειμένων.

ΕΙΚΟΝΑ

Περίληψη

Σε αυτό το κεφάλαιο, είδαμε πώς να συναρμολογούμε “τα κομμάτια” του κώδικα προγράμματος. “Τα κομμάτια” που θα χρησιμοποιήσουμε για να δημιουργήσουμε τα προγράμματα μας συμπεριλαμβάνουν:

instruction *(πληροφορία)*-μια εντολή που εκτελείται έτσι ώστε τα αντικείμενα να κάνουν μια συγκεκριμένη ενέργεια.

control structure *(δομές ελέγχου)*-μια εντολή που ελέγχει την εκτέλεση ενός μπλοκ πληροφοριών

function *(συνάρτηση)*-ρωτά μια ερώτηση για μια υπόθεση ή υπολογίζει μια τιμή

expression *(παράσταση)*-μια μαθηματική πράξη νούμερων ή άλλων τιμών.

Το Alice παρέχει ενσωματωμένες συναρτήσεις για τον κόσμο και τα αντικείμενα του. Οι συναρτήσεις μπορεί να έχουν διάφορους τύπους τιμών (για παράδειγμα αριθμός, λογική τιμή, αντικείμενο). Οι παραστάσεις σας επιτρέπουν να υπολογίσετε μια τιμή ή να κάνετε σύγκριση μιας ιδιότητας ανάμεσα σε δυο αντικείμενα. Μια χρήση των συναρτήσεων και των παραστάσεων σε ένα πρόγραμμα με κίνηση είναι να αποφευχθεί η σύγκρουση (όταν ένα αντικείμενο κινείται προς την ίδια κατεύθυνση με ένα άλλο).

Μπορούμε να χρησιμοποιήσουμε ενσωματωμένες συναρτήσεις και λογικές παραστάσεις (παραστάσεις που έχουν τιμή *true* ή *false*) για να ελέγξουμε την τρέχουσα κατάσταση στον κόσμο και να πάρουμε αποφάσεις

για το αν θα εκτελεστεί ή όχι ένα συγκεκριμένο κομμάτι κώδικα. Στο Alice, οι δομές ελέγχου υποθετικής εκτέλεσης είναι η εντολή *If/Else.* Μια εντολή *If* έχει δύο μέρη: το μέρος *If* και το μέρος *Else.* Στο μέρος *If,* ελέγχεται μια υπόθεση και παίρνεται μια απόφαση σύμφωνα με την υπόθεση. Εάν η υπόθεση είναι true (αληθής*),* εκτελείται το μέρος της *If,* στην αντίθετη περίπτωση εκτελείται το μέρος της *Else.* Είναι πιθανό το μέρος της *Else* να είναι *do nothing,* και σε αυτή την περίπτωση δεν γίνεται τίποτα. (Είναι επίσης πιθανό και για το μέρος της *If* ο κώδικας να είναι *do nothing*-αλλά αυτός είναι ένας αδέξιος τρόπος υπόθεσης).

Μια απλή επαναληπτική δομή ελέγχου είναι η *Loop*. Μια εντολή *Loop* σας επιτρέπει να επαναλάβετε ένα τμήμα ενός προγράμματος ένα συγκεκριμένο πλήθος φορών.

Σημαντικά θέματα σε αυτό το κεφάλαιο

- Οι συναρτήσεις μπορούν να χρησιμοποιηθούν για να γίνουν ερωτήσεις σχετικά με τις ιδιότητες ενός κόσμου ή των αντικειμένων του. Επίσης μπορούν να χρησιμοποιηθούν για να υπολογιστεί μια τιμή.
- Όταν καλείται μια συνάρτηση, επιστρέφει ένα συγκεκριμένο τύπο τιμής.
- Μια συνάρτηση λογικής τιμής επιστρέφει *true* ή *false.*
- Μια παράσταση μπορεί να χρησιμοποιήσει μια μαθηματική πράξη (πρόσθεση, αφαίρεση, πολλαπλασιασμός, διαίρεση) για να υπολογίσει μια αριθμητική τιμή.
- Ένα άλλο είδος παραστάσεων συγκρίνει ένα αντικείμενο με ένα άλλο, χρησιμοποιώντας σχεσιακούς τελεστές (==, !=, >, >=, <, <=). Το αποτέλεσμα είναι *true* ή *false.*
- Μια δομή ελέγχου υποθετικής εκτέλεσης (στο Alice μια εντολή *If/Else*) χρησιμοποιείται για να ληφθεί μια απόφαση για το αν θα εκτελεστεί ένα συγκεκριμένο τμήμα κώδικα.
- Μια επαναληπτική δομή ελέγχου χρησιμοποιείται για να επαναλάβει ένα τμήμα κώδικα προγράμματος. Ένας απλός έλεγχος επανάληψης στο Alice είναι η εντολή *Loop.*

Μέρος ΙΙ:

Αντικειμενοστραφής προγραμματισμός και προγραμματισμός οδηγούμενος από γεγονότα

Όταν δημιουργήσατε τα δικά σας κινούμενα σχέδια σε προηγούμενα κεφάλαια, μπορεί να σκεφτήκατε πιο σύνθετα σενάρια με περισσότερους ελιγμούς και στροφές στην ιστορία. Φυσικά, όσο η ιστορία γίνεται πιο περίπλοκη, γίνεται και ο κώδικας. Ο κώδικας του προγράμματος μπορεί να αυξηθεί σε πολλές σειρές. Δεν είναι μόνο τα κινούμενα σχέδια πολύπλοκα. Τα λογισμικά που κυκλοφορούν μπορούν να έχουν χιλιάδες ακόμη και εκατομμύρια γραμμές κώδικα. Πως μπορεί να αντεπεξέλθει ο προγραμματιστής με τόσες πολλές γραμμές κώδικα; Μια τεχνική είναι να χωρίσουμε το μεγάλο πρόγραμμα σε μικρά διαχειρίσιμα κομμάτια, κάνοντας έτσι ευκολότερο τον σχεδιασμό του. Τα μικρότερα κομμάτια είναι και ευκολότερο να διαβαστούν και να διορθωθούν. Ο αντικειμενοστραφής προγραμματισμός χρησιμοποιεί τις κλάσεις, τα αντικείμενα και τις μεθόδους ως κύρια χαρακτηριστικά του προγράμματος, τα οποία βοηθούν να οργανώσετε τα μεγάλα προγράμματα σε μικρότερα διαχειρίσιμα κομμάτια. Σε αυτό και στο επόμενο κεφάλαιο, θα μάθετε πώς να γράφετε πιο πολύπλοκα και εντυπωσιακά προγράμματα χρησιμοποιώντας αντικείμενα και γράφοντας τις δικές σας μεθόδους.

Κλάσεις

Μια κλάση ορίζει ένα συγκεκριμένο είδος αντικειμένου. Στο Alice, οι κλάσεις είναι προκαθορισμένα ως 3D μοντέλα και παρέχονται στην βιβλιοθήκη, όπου και κατηγοριοποιούνται σε ομάδες όπως Animals, People, Buildings, Sets and Scenes, Space κτλ. Η εικόνα 4-0-1 παρουσιάζει μερικές κλάσεις του φακέλου Animals. Σημειώστε ότι το όνομα της κλάσης ξεκινάει με κεφαλαίο.

ΕΙΚΟΝΑ 4-0-1. Κλάσεις 3D μοντέλων στον φάκελο Animals

Κάθε κλάση είναι ένα πρόγραμμα δράσης που λέει στο Alice ακριβώς πώς να δημιουργήσει και να εμφανίσει ένα αντικείμενο της κλάσης. Όταν ένα αντικείμενο δημιουργηθεί και χρησιμοποιηθεί, ονομάζεται στιγμιότυπο (instance) της κλάσης.

Αντικείμενα

Στην εικόνα 4-0-2, Person και Dog είναι κλάσεις, joe, stan και cindy είναι στιγμιότυπα της κλάσης Person και spike, scamp και fido είναι στιγμιότυπα της κλάσης Dog. Σημειώστε ότι το όνομα ενός αντικειμένου ξεκινάει με μικρό. Αυτό το στυλ μας επιτρέπει να ξεχωρίζουμε εύκολα μια κλάση από ένα αντικείμενο. Όλα τα αντικείμενα της ίδιας κλάσης έχουν ομοιότητες. Όλα τα αντικείμενα της κλάσης Person έχουν ιδιότητες όπως δυο πόδια, δυο χέρια, ύψος και χρώμα ματιών. Τα αντικείμενα- Person μπορούν να κάνουν ενέργειες όπως περπάτημα και ομιλία. Όλα τα αντικείμενα- Dog έχουν ιδιότητες όπως τέσσερα πόδια, ύψος, χρώμα τριχώματος και έχουν την

ικανότητα να τρέξουν και να γαβγίσουν. Παρόλο που κάθε αντικείμενο ανήκει σε μια κλάση, είναι μοναδικό με τον δικό του τρόπο. Ο joe είναι ψηλός και έχει πράσινα μάτια. Η cindy είναι κοντή και έχει μπλε μάτια. Ο spike έχει καφέ τρίχωμα και ο γάβγισμά του είναι αδύνατο γρύλισμα και ο scamp έχει χρυσαφί τρίχωμα και το γάβγισμα του είναι τσιριχτό.

ΕΙΚΟΝΑ 4-0-2. Οργάνωση των αντικείμενων στις κλάσεις

Στην εικόνα 4-0-3, ο larry, η lila και ο louis είναι στιγμιότυπα της κλάσης Lemur στο Alice. Ονομάσαμε τους Lemur σε αυτό τον κόσμο, τους δώσαμε διαφορετικό ύψος και αλλάξαμε το χρώμα της lila. Ο larry, η lila και ο louis είναι αντικείμενα της ίδιας κλάσης Lemur και έχουν πολλά κοινά χαρακτηριστικά. Αλλά διαφέρουν στο ότι ο larry είναι ο υψηλότερος, η lila έχει σκούρο πλούσιο τρίχωμα και ο louis είναι ο πιο κοντός.

ΕΙΚΟΝΑ 4-0-3. Αντικείμενα της κλάσης Lemur στο Alice

Μέθοδοι

Οι μέθοδοι είναι μια συντονισμένη ακολουθία εντολών που θα εκτελεστεί όταν ζητηθεί. Έχετε ήδη χρησιμοποιήσει μεθόδους στα προγράμματα σας. Κάθε αντικείμενο στο Alice έχει ένα ρεπερτόριο εντολών που ξέρει πως θα εκτελέσει-*move, turn, turn to face* κτλ. Αυτές οι εντολές είναι στην πραγματικότητα είναι οι αρχικές μέθοδοι που είναι ενσωματωμένες στο λογισμικό Alice. Οι αρχικές μέθοδοι μπορούν να οργανωθούν σε μια δική σας μέθοδο-για να εκτελεστεί ένα μικρό κομμάτι του προγράμματος. Κάθε μέθοδος εκτελεί την δική της διεργασία, άλλα όλες οι μέθοδοι σε ένα πρόγραμμα συνεργάζονται για να δημιουργήσουν το τελικό αποτέλεσμα.

Όσο το πρόγραμμα σας μεγαλώνει, είναι αναγκαίο να χρησιμοποιήσετε πολλές μεθόδους έτσι ώστε να οργανώσετε το πρόγραμμά σας. Οι μέθοδοι χωρίζουν ένα πρόγραμμα σε μικρά διαχειρίσιμα κομμάτια που συνεργάζονται για να δημιουργήσουν ένα σημαντικό σύνολο. Έτσι όπως οι παράγραφοι, οι ενότητες και τα κεφάλαια κάνουν ένα βιβλίο ευκολότερο στην ανάγνωση, οι μέθοδοι κάνουν ένα πρόγραμμα ευκολότερο στην ανάγνωση και στη διαχείριση. Οι μέθοδοι επίσης παρέχουν πολλά πλεονεκτήματα. Για παράδειγμα όταν γραφεί μια μέθοδος μας επιτρέπει να ασχοληθούμε με μια συνολική διεργασία αντί για μικρές ενέργειες που χρειαζόταν για να ολοκληρωθεί η διεργασία. Αυτό καλείται αφαίρεση (abstraction).

Μερικές μέθοδοι χρειάζονται συγκεκριμένες πληροφορίες για να εκτελέσουν μια ενέργεια. Για παράδειγμα μια μέθοδος *move* χρειάζεται *direction* (*κατεύθυνση), (forward, backward, left, right, up, down)* (*μπροστά, πίσω, πάνω, κάτω, δεξιά, αριστερά*) και μια *απόσταση* (σε μέτρα). Μια παράμετρος λειτουργεί σαν καλάθι όπου συλλέγονται οι πληροφορίες που στέλνουμε στην μέθοδο. Μπορείτε να σκεφτείτε την μέθοδο κάτι σαν συνταγή-ένα σύνολο εντολών που περιγράφει πώς να εκτελέσετε κάποιες ενέργειες.

(Σαν παράδειγμα δείτε την συνταγή στην αρχή αυτού του κεφαλαίου για να κάνετε τάρτα φράουλα). Οι παράμετροι κρατούν συγκεκριμένα τμήματα πληροφοριών. Σε μια συνταγή, μια παράμετρος μπορεί να καθορίσει λεπτομερώς το ποσό του νερού. Σε μια μέθοδο μια παράμετρος μπορεί να καθορίσει την απόσταση που θα κινηθεί ένα διαστημόπλοιο.

Στο Alice, μπορείτε να ορίσετε μεθόδους για ένα μόνο αντικείμενο, ή για δυο ή περισσότερα αντικείμενα που αλληλεπιδρούν. Αυτό μοιάζει με τον τρόπο που δουλεύει ένας σκηνοθέτης με το σύνολο των ηθοποιών σε ένα έργο. Ο σκηνοθέτης δίνει οδηγίες κάποιες φορές σε ένα μόνο ηθοποιό και άλλες φορές σε μια ομάδα ηθοποιών ώστε να συνεργαστούν για την σκηνή. Οι μέθοδοι που αναφέρουν περισσότερα από ένα αντικείμενα είναι μέθοδοι επιπέδου κόσμου. Οι μέθοδοι που καθορίζουν την συμπεριφορά ενός μόνο αντικειμένου μπορούν να θεωρηθούν μέθοδοι επιπέδου κλάσης.

Η ενότητα 4-1 παρουσιάζει μια εισαγωγή στις μεθόδους επιπέδου κλάσης. Ένα πλεονέκτημα των μεθόδων επιπέδου κλάσης είναι ότι μόλις ορισθούν νέες μέθοδοι, μπορούμε να δημιουργήσουμε μια νέα κλάση με όλες τις νέες μεθόδους (και τις παλιές) σαν δυνατές ενέργειες. Αυτή είναι γνωστή ως κληρονομικότητα (inheritance)-η νέα κλάση κληρονομεί τις μεθόδους της παλιάς κλάσης.

4-1 Μέθοδοι επιπέδου κλάσης και κληρονομικότητα

Οι βιβλιοθήκες των 3D μοντέλων στο Alice μας παρέχουν διάφορες και όμορφα σχεδιασμένες κλάσεις αντικειμένων για δημιουργία ενός πλούσιου σκηνικού υπόβαθρου σε ένα εικονικό κόσμο. Όταν προσθέτετε ένα στιγμιότυπο ενός 3D μοντέλου σε ένα κόσμο του Alice, αυτό γνωρίζει ήδη πώς να εκτελέσει ένα σύνολο μεθόδων-*move, turn, roll, resize.* Η κλάση του 3D μοντέλου έχει ήδη ορίσει αυτές τις μεθόδους. Εφόσον γράψετε πολλά προγράμματα, είναι φυσικό να σκεφτείτε να επεκτείνετε τις ενέργειες που “γνωρίζει” να εκτελεί ένα αντικείμενο.

Σε αυτή την ενότητα, θα μάθετε να γράφετε νέες μεθόδους που καθορίζουν την εκτέλεση νέων ενεργειών από ένα αντικείμενο που δρα μόνο του. Αυτές τις καλούμε μεθόδους επιπέδου κλάσης. Οι μέθοδοι επιπέδου κλάσης είναι ξεχωριστές, επειδή μπορούμε να σώσουμε ένα αντικείμενο μαζί με τις νέες του μεθόδους σαν ένα νέο είδος μεθόδου. Στο Alice, το νέο είδος μεθόδου αποθηκεύεται σαν μια νέα κλάση 3D μοντέλου. Τα στιγμιότυπα της νέας κλάσης θα γνωρίζουν πώς να εκτελέσουν όλες τις ενέργειες που είχαν καθοριστεί στην αρχική κλάση, αλλά επίσης θα γνωρίζουν και όλες τις ενέργειες στις νέα καθορισμένες μεθόδους. Λέμε ότι η νέα κλάση κληρονομεί όλες τις ιδιότητες και τις μεθόδους της αρχικής κλάσης.

Παράδειγμα

Σκεφτείτε την iceSkater (παγοδρόμο) που φαίνεται στην εικόνα 4-1-1. (Η κλάση IceSkater βρίσκεται στην συλλογή People και η κλάση Lake είναι από την συλλογή Environments στην βιβλιοθήκη).

ΕΙΚΟΝΑ 4-1-1. Η iceSkater

Θέλουμε η παγοδρόμος να εκτελέσει τυπικές φιγούρες στον πάγο. Είναι ντυμένη με ειδική στολή και φοράει παγοπέδιλα, αλλά αυτό δεν σημαίνει ότι ξέρει να κάνει skate. Παρόλ' αυτά όλα τα αντικείμενα στο Alice "γνωρίζουν" πώς να εκτελούν απλές μεθόδους όπως *move, turn* και *roll.* Μπορούμε να χρησιμοποιήσουμε ένα συνδυασμό αυτών των απλών μεθόδων για να "διδάξουμε" στην παγοδρόμο πώς να εκτελεί αυτές τις φιγούρες, δηλαδή μια πιο πολύπλοκη ενέργεια. Ξεκινούμε με μια μέθοδο για να κάνουμε την παγοδρόμο να εκτελέσει μια κίνηση παγοδρομίας.

Μέθοδος επιπέδου κλάσης

Οι ελιγμοί στην παγοδρομία είναι πολύπλοκες πράξεις που απαιτούν πολλές εντολές κινήσεων, συμπεριλαμβάνοντας διάφορα μέρη του σώματος. (Οι επαγγελματίες ανιματέρ της Disney και της Pixar μπορεί να περάσουν πολλές ώρες παρατηρώντας την κίνηση των διαφόρων μερών του ανθρώπινου σώματος για να δημιουργήσουν μια ρεαλιστική κίνηση). Για να κάνει skate, ένας παγοδρόμος ολισθαίνει μπροστά πρώτα με το αριστερό και μετά με το δεξί πόδι. Φυσικά ολόκληρο το σώμα κινείται μπροστά όσο τα πόδια εκτελούν τις κινήσεις. Τα βήματα σε μια παγοδρομία γίνονται διαδοχικά, σαν μια ακολουθία κινήσεων σε μια διάταξη σκηνών, όπως φαίνεται παρακάτω.

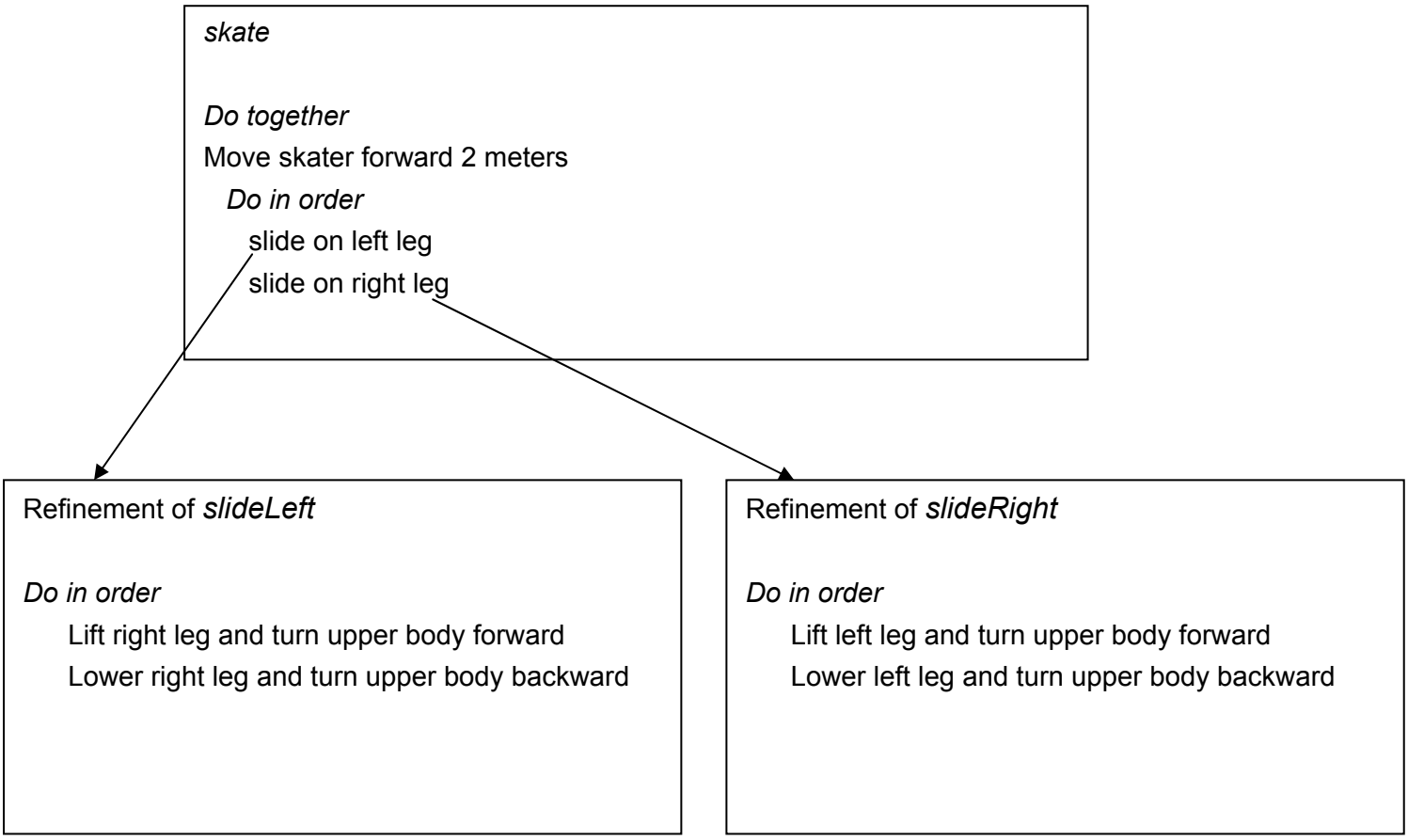
```
skate

Do together
Move skater forward 2 meters
  Do in order
    slide on left leg
    slide on right leg
```

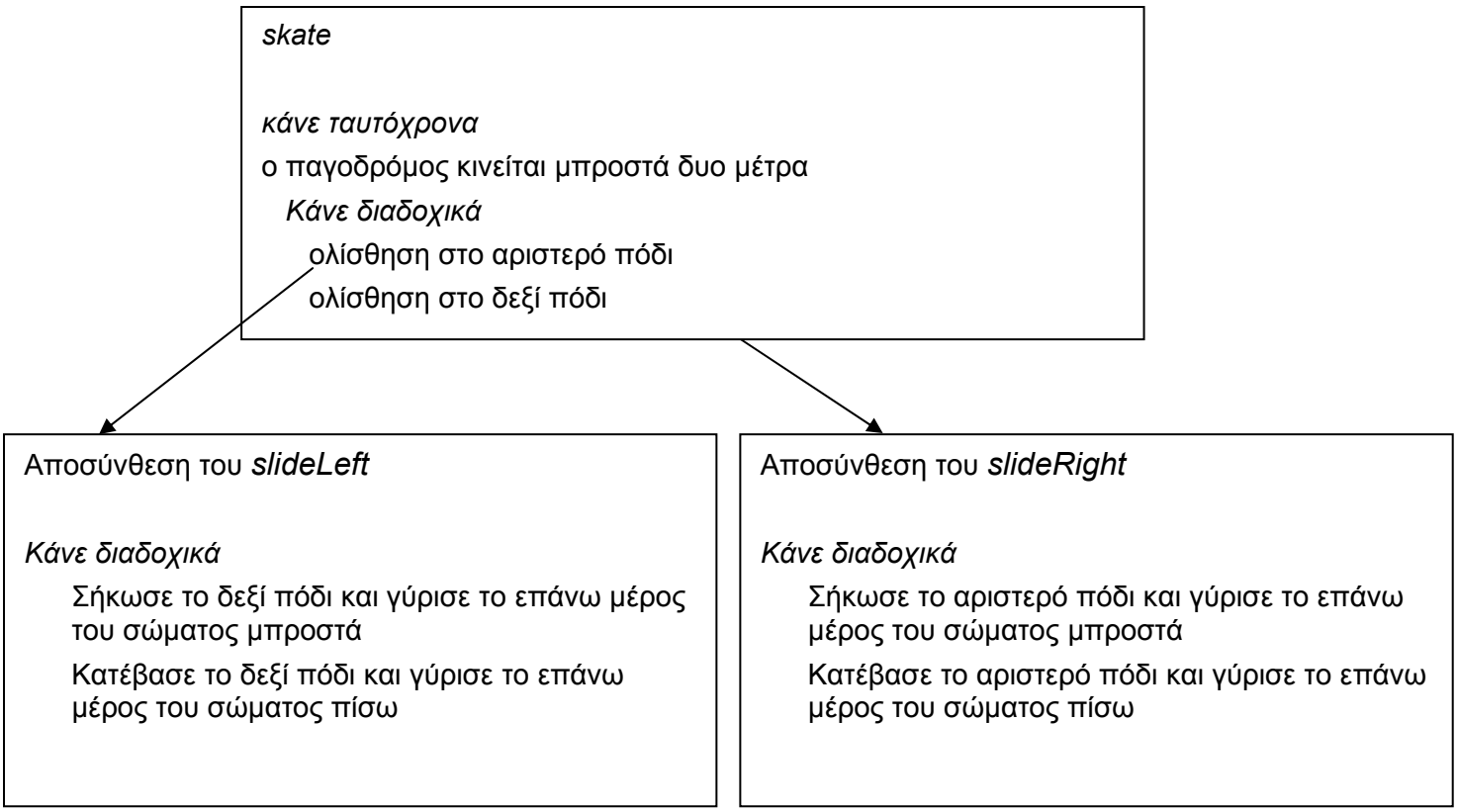
```
skate

κάνε ταυτόχρονα
ο παγοδρόμος κινείται μπροστά δυο μέτρα
  Κάνε διαδοχικά
    ολίσθηση στο αριστερό πόδι
    ολίσθηση στο δεξί πόδι
```

Προσέξτε ότι στο παραπάνω σενάριο η παγοδρομία χωρίζεται σε δυο κομμάτια-ολίσθηση στο αριστερό πόδι, ολίσθηση στο δεξί πόδι. Οι κινήσεις ολίσθησης μπορούν να διασπαστούν σε απλούστερες μεθόδους. Η

αποσύνθεση μιας πολύπλοκης πράξης σε απλούστερες πράξεις ονομάζεται διάσπαση (refinement). Εδώ χρησιμοποιήθηκε μια τεχνική γνωστή ως βηματική διάσπαση. Πρώτα περιγράψαμε τις γενικές πράξεις και μετά διασπάσαμε την κάθε πράξη σε μικρότερα και μικρότερα κομμάτια (επιτυχώς επεξεργασμένα) μέχρι η όλη διεργασία να ορίζεται από απλές πράξεις. Κάθε κομμάτι συνεισφέρει ένα μικρό μέρος στο όλο πρόγραμμα. Όλα τα κομμάτια μαζί ολοκληρώνουν την διεργασία.

Το επόμενο διάγραμμα παρέχει την διάσπαση της *slideLeft* (ολίσθησης αριστερά) και της *slideRight* (ολίσθησης δεξιά). Η *slideLeft* απαιτεί να σηκώσει το δεξί πόδι και να γυρίσει το επάνω μέρος του σώματος ελαφρώς μπροστά. Μετά να κατεβάσει το δεξί πόδι και να γυρίσει το πάνω μέρος του σώματος ελαφρώς πίσω. Οι αντίστοιχες κινήσεις θα γίνουν για την *slideRight.*





Τίποτε άλλο δεν χρειάζεται να αποσυντεθεί. Είμαστε έτοιμοι να μεταφράσουμε το σχέδιο σε κώδικα προγράμματος. Θα μπορούσαμε να μεταφράσουμε αυτό το σχέδιο σε εντολές με μια μέθοδο, αλλά αυτό θα ήταν πολύ μακρύ. Επιπλέον μπορείτε να δείτε ότι χρησιμοποιήσαμε βηματική διάσπαση για να χωρίσουμε την διεργασία μας σε διακριτά κομμάτια. Έτσι, θα παρουσιάσουμε την συγγραφή πολλών μικρών μεθόδων και την ταυτόχρονη λειτουργία τους για να επιτύχουμε μια μεγαλύτερη διεργασία.

Η παγοδρομία είναι μια πολύπλοκη πράξη που έχει σχεδιαστεί ειδικά για την iceSkater και δεν περιέχει άλλα αντικείμενα. Δηλαδή οι *slideLeft* και *slideRight* έχουν σχεδιαστεί μόνο για την iceSkater. Οι μέθοδοι πρέπει να γραφούν σαν μέθοδοι επιπέδου κλάσης επειδή χρησιμοποιούν ως αντικείμενο μόνο την iceSkater. Ξεκινάμε με την *slideLeft* μέθοδο. Στο δέντρο αντικειμένων επιλέγεται η iceSkater και στην καρτέλα methods πατάμε create new method (δημιουργία νέας μεθόδου). Στο παράθυρο New Method (νέα μέθοδος) δίνουμε το όνομα *slideLeft*. Η αριστερή πλευρά της εικόνας 4-1-2 παρουσιάζει το πρώτο βήμα. Πατάμε OK. Ο συντάκτης ανοίγει ένα νέο πλαίσιο όπου δημιουργείται ο κώδικας. Η δεξιά πλευρά της εικόνας 4-1-2 παρουσιάζει τη μέθοδο και το νέο πλαίσιο.

ΕΙΚΟΝΑ 4-1-2. Η *slideLeft* μέθοδος επιπέδου κλάσης

Για να υλοποιήσουμε την μέθοδο *slideLeft,* εισάγουμε πληροφορίες στον συντάκτη. Η ιδέα είναι να μεταφράσουμε το σχέδιο σε εντολές προγράμματος. Για παράδειγμα, για να μεταφράσουμε το σχέδιο της ολίσθησης στο αριστερό πόδι, χρησιμοποιούμε τα παρακάτω:

Βήμα σχεδιασμού	**Πληροφορία**
Σήκωσε το δεξί πόδι	*turn* the rightLeg forward (Γύρισε το δεξί πόδι μπροστά)
Γύρισε μπροστά το επάνω μέρος του σώματος	*turn* the upperBody forward
(Εισάγουμε ένα μικρό χρονικό διάστημα αναμονής)	(Γύρισε το επάνω μέρος του σώματος μπροστά*)*
Κατέβασε το δεξί πόδι	*turn* the rightLeg backward (Γύρισε το δεξί πόδι πίσω)
Γύρισε πίσω το επάνω μέρος του σώματος	*turn* the upperBody backward (Γύρισε το επάνω μέρος του σώματος πίσω)

ΕΙΚΟΝΑ 4-1-3. Η *slideLeft* μέθοδος

Η εικόνα 4-3-1 παρουσιάζει τις εντολές για ολίσθηση στο αριστερό πόδι. Οι εντολές για ολίσθηση στο δεξί πόδι είναι παρόμοιες. Η εικόνα 4-1-4 παρουσιάζει τις εντολές για ολίσθηση στο δεξί πόδι.

Με τις μεθόδους *slideLeft* και *slideRight,* είμαστε τώρα έτοιμοι να γράψουμε την μέθοδο *skate.* Η μέθοδος *skate* είναι εύκολη: η *slideLeft* και μετά η *slideRight* την ίδια στιγμή που η παγοδρόμος κινείται μπροστά. Για να δημιουργήσετε τη μέθοδο *skate,* οι μέθοδοι *slideLeft* και *slideRight* τοποθετούνται στον συντάκτη, όπως ακριβώς είχε τοποθετηθεί κάθε άλλη μέθοδος. Η εικόνα 4-1-5 παρουσιάζει την τοποθέτηση των δυο μεθόδων μέσα στην μέθοδο *skate.*

ΕΙΚΟΝΑ 4-1-4. Η *slideRight* μέθοδος

ΕΙΚΟΝΑ 4-1-5. Η *skate* μέθοδος

Η μέθοδος *skate* κινεί την παγοδρόμο μπροστά και καλεί διαδοχικά τις μεθόδους *slideLeft* και *slideRight.* Παρατηρήστε ότι η κλήση των μεθόδων *slideLeft* και *slideRight* γίνεται μέσα σε ένα μπλοκ *do in order* και αυτό βρίσκεται ενσωματωμένο μέσα σε ένα μπλοκ *do together.* Το μπλοκ *do together,* χρειάζεται για να σιγουρέψει ότι οι εντολές για την κίνηση της παγοδρόμου μπροστά και οι πληροφορίες για την ολίσθηση γίνονται ταυτόχρονα. Η διάρκεια της μπροστινής κίνησης της παγοδρόμου είναι το άθροισμα της διάρκειας των δυο ολισθήσεων. Αν προσέξετε λίγο την διάρκεια των εντολών σε ένα μπλοκ *do together* θα σας βοηθήσει να συγχρονίσετε τις κινήσεις ώστε να ξεκινούν και να τελειώνουν ταυτόχρονα. Η εικόνα 4-1-5 παρουσιάζει την μέθοδο *skate* ολοκληρωμένη.

Εάν πατήσετε το κουμπί Play το πρόγραμμα δεν θα τρέξει. Παρόλο που η μέθοδος *skate* έχει ορισθεί, δεν έχετε πει στο Alice να την εκτελέσει. Η μέθοδος δεν έχει κληθεί. Για να καλέσετε την μέθοδο *skate* στο σκηνικό, είναι απαραίτητο να τοποθετήσετε την μέθοδο *skate* μέσα στην μέθοδο *world.my first method.* Όταν κληθεί η μέθοδος *skate,* η παγοδρόμος κάνει την φιγούρα της.

Ένα δεύτερο παράδειγμα-χρήση παραμέτρου

Η κίνηση της παγοδρόμου είναι εντυπωσιακή! Ας κάνουμε μια δεύτερη μέθοδο ώστε η παγοδρόμος να κάνει μια περιστροφή. Πρέπει πάλι να γράψουμε μεθόδους που θα λειτουργούν μαζί για να εκτελέσουμε μια πολύπλοκη πράξη. Σε μια περιστροφή, η παγοδρόμος πρέπει να γυρίσει γύρω από τον εαυτό τις αρκετές φορές.

Μια τεχνική περιστροφής έχει τρία μέρη, η προετοιμασία για την σβούρα, η σβούρα και το τελείωμα της σβούρας. Στην προετοιμασία της σβούρας, τα πόδια και τα χέρια της παγοδρόμου αλλάζουν θέση για να παρέχουν την δύναμη που χρειάζεται ώστε να προωθηθεί το σώμα της για την περιστροφή. Μετά η παγοδρόμος στριφογυρίζει. Μετά την σβούρα, τα χέρια και τα πόδια θα πρέπει να τοποθετηθούν ξανά στις αρχικές τους θέσεις. Μια μέθοδος για περιστροφή *(spin)* πιθανά θα κάνει την παγοδρόμο να εκτελέσει αρκετές περιστροφές. Δεν ξέρουμε ακριβώς πόσες φορές θα γυρίσει η παγοδρόμος. Αυτό είναι ένα παράδειγμα όπου μια παράμετρος μπορεί να βοηθήσει. Μια παράμετρος σας επιτρέπει να στείλετε μια πληροφορία σε μια μέθοδο. Για παράδειγμα σε μια μέθοδο *move* μπορείτε να στείλετε πληροφορίες που ορίζουν την απόσταση, την κατεύθυνση και την διάρκεια. Λέμε ότι οι τιμές στέλνονται ως ορίσματα στις παραμέτρους. Μια παράμετρος, *howManyTimes (πόσες φορές),* είναι απαραίτητη για να καθοριστεί ο αριθμός των περιστροφών της παγοδρόμου. Η διάταξη σκηνών φαίνεται παρακάτω:

spin

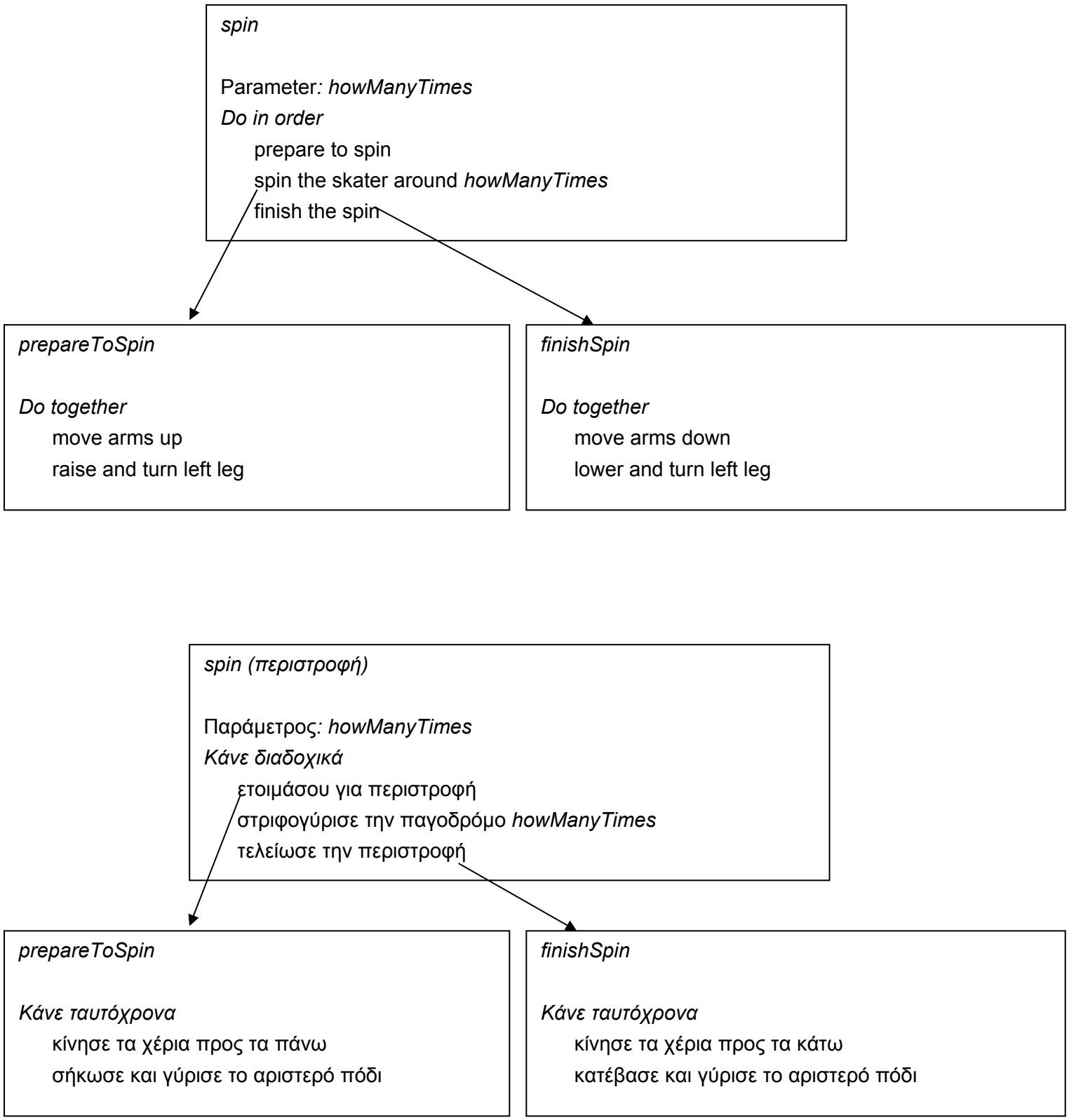
Parameter*: howManyTimes*
Do in order
    prepare to spin
    spin the skater around *howManyTimes*
    finish the spin

spin (περιστροφή)

Παράμετρος*: howManyTimes*
Κάνε διαδοχικά
    ετοιμάσου για περιστροφή
    στριφογύρισε την παγοδρόμο *howManyTimes*
    τελείωσε την περιστροφή

Μπορούμε να χρησιμοποιήσουμε την βηματική αποσύνθεση για να σχεδιάσουμε τα απλά βήματα για το κάθε κομμάτι της περιστροφής. Το βήμα “prepare to spin” (ετοιμάσου για περιστροφή) μπορεί να γραφεί ως μέθοδος (*prepareToSpin*) όπου τα χέρια της παγοδρόμου κινούνται προς τα πάνω και το ένα πόδι γυρίζει. Το βήμα “finish spin” (τελείωσε την περιστροφή) μπορεί επίσης να γραφεί σαν μέθοδος *(finishSpin)* η οποία κινεί τα χέρια και τα πόδια

στην αρχική τους θέση, πριν την περιστροφή. Το επόμενο διάγραμμα υλοποιεί την αποσύνθεση της μεθόδου *spin.*



Τίποτε άλλο δεν χρειάζεται να αποσυντεθεί. Είμαστε έτοιμοι να μεταφράσουμε το σχέδιο σε κώδικα προγράμματος. Ας τονίσουμε πάλι ότι πρέπει να οριστούν μέθοδοι επιπέδου κλάσης γιατί ορίζουμε την κίνηση ενός μόνο αντικειμένου.

Η εικόνα 4-1-6 παρουσιάζει την μέθοδο *prepareToSpin*, όπου η παγοδρόμος ανυψώνει το αριστερό της πόδι όσο σηκώνει τα χέρια της.

ΕΙΚΟΝΑ 4-1-6. Η *prepareToSpin* μέθοδος για την ανύψωση του ποδιού και των χεριών

Η εικόνα 4-1-7 παρουσιάζει την μέθοδο *finishSpin* η οποία τοποθετεί τα χέρια και τα πόδια της παγοδρόμου στην αρχική τους θέση.

ΕΙΚΟΝΑ 4-1-7. Η *finishSpin* μέθοδος για το κατέβασμα του ποδιού και των χεριών

Τώρα που έχουν γραφεί οι *prepareToSpin, finishSpin,* μπορεί να δημιουργηθεί και η *spin.* Στην μέθοδο *spin,* υπάρχει το κουμπί δημιουργία νέας παραμέτρου. Πατάμε το κουμπί και εμφανίζεται ένα πλαίσιο κειμένου όπως φαίνεται στην εικόνα 4-1-8. Επιλέγουμε name (όνομα) και type (τύπο) της παραμέτρου. Ο *τύπος* της παραμέτρου μπορεί να είναι *Number, Boolean, Object ή Other* (για παράδειγμα χρώμα ή ήχος). Η παράμετρος *howManyTimes* είναι ένας αριθμός που ορίζει πόσες φορές θα γυρίσει η παγοδρόμος (μια περιστροφή είναι μια ολόκληρη σβούρα). Η σειρά με την οποία καλούνται οι μέθοδοι είναι σημαντική έτσι ώστε να γίνει η κίνηση με ακρίβεια. Η ολοκληρωμένη μέθοδος *spin* φαίνεται στην εικόνα 4-1-9.

ΕΙΚΟΝΑ 4-1-8. Δημιουργία μιας νέας παραμέτρου στην μέθοδο *spin*

Ο κώδικας για τα δύο παραδείγματα παραπάνω (skate, spin) είναι λίγο μεγαλύτερος από τους κώδικες των προηγούμενων κεφαλαίων. Είναι σημαντικό ότι ο κώδικας είναι εύκολα κατανοητός, επειδή αποσυνθέσαμε με προσοχή την συνολική διεργασία σε μικρές μεθόδους. Οι μικρές μέθοδοι συνεργάζονται για να εκτελεστεί η συνολική ενέργεια. Επίσης οι μέθοδοι είναι καλά τεκμηριωμένες, με σχόλια που μας επεξηγούν τι κάνει η κάθε μια. Ο καλός σχεδιασμός και τα σχόλια κάνουν τον κώδικα μας κατανοητό, εύκολο στην χρήση και στην διόρθωση.

ΕΙΚΟΝΑ 4-1-9. Η μέθοδος *spin*

Δημιουργία μιας νέας κλάσης

Η κλάση iceSkater τώρα έχει δυο μεθόδους την *skate* και την *spin.* (Επίσης έχει και μικρότερες μεθόδους που υλοποιούν τις παραπάνω). Το γράψιμο και ο έλεγχος αυτών των μεθόδων απαίτησε χρόνο και κόπο. Θα ήταν κρίμα να βάλετε όλες αυτές τις μεθόδους μόνο σε ένα κόσμο και να μην μπορείτε να τις χρησιμοποιήσετε ξανά σε κάποιο άλλο πρόγραμμα. Θέλουμε να αποθηκεύσουμε την iceSkater και τις νέες της μεθόδους έτσι ώστε να τις χρησιμοποιήσουμε σε ένα άλλο κόσμο. (Να μην ξαναχρειαστεί να γράψουμε αυτές τις μεθόδους για το άλλο πρόγραμμα). Για να το πετύχουμε αυτό, η iceSkater πρέπει να αποθηκευτεί σαν ένα νέο 3D μοντέλο (κλάση).

Η αποθήκευση της iceSkater (με τις νέες της μεθόδους) ως μια νέα κλάση είναι μια διαδικασία δυο βημάτων. Το πρώτο βήμα είναι να μετονομάσουμε την κλάση. Αυτό είναι ΣΗΜΑΝΤΙΚΟ ΒΗΜΑ. Θέλουμε το Alice να σώσει την νέα κλάση με ένα διαφορετικό 3D όνομα από αυτό της αρχικής κλάσης. Για να μετονομάσουμε ένα αντικείμενο, κάνουμε δεξί κλικ πάνω στο όνομα στο δέντρο των αντικειμένων, επιλέγουμε *rename* (μετονομασία) και δίνουμε το νέο όνομα. Σε αυτό το παράδειγμα μετονομάζουμε την αρχική κλάση της iceSkater σε cleverSkater, όπως φαίνεται στην εικόνα 4-1-10.

Το δεύτερο βήμα είναι να την αποθηκεύσουμε ως νέα κλάση: δεξί κλικ στην cleverSkater στο δέντρο αντικειμένων και αυτή τη φορά πατάμε αποθήκευση αντικειμένου *(save object)*. Στο μενού, πηγαίνετε στον φάκελο ή τον κατάλογο που θέλετε να αποθηκεύσετε την νέα κλάση. Εικόνα 4-1-11. Μετά πατήστε *save* (αποθήκευση).

ΕΙΚΟΝΑ 4-1-10. Μετονομασία της κλάσης

Η κλάση αυτόματα παίρνει το όνομα του αντικειμένου, ξεκινώντας με ένα κεφαλαίο γράμμα και με κατάληξη αρχείου .a2c από το “Alice version 2.0 Class” (όπως και η κατάληξη .a2w για ένα κόσμο από το “ Alice version 2.0 World”).

ΕΙΚΟΝΑ 4-1-11. Αποθήκευση νέας κλάσης

Εφόσον δημιουργήθηκε η νέα κλάση, μπορεί να χρησιμοποιηθεί σε ένα νέο κόσμο επιλέγοντας “Import” (εισαγωγή) από το μενού File, όπως στην εικόνα 4-1-12. Όταν ένα στιγμιότυπο της κλάσης cleverSkater προστίθεται στον κόσμο, θα είναι σαν ένα στιγμιότυπο της κλάσης της παγοδρόμου εκτός από το ότι ένα αντικείμενο- cleverSkater γνωρίζει όχι μόνο όλες τις μεθόδους που μπορεί να κάνει ένα αντικείμενο-iceSkater αλλά και πώς να κάνει skate και spin.

ΕΙΚΟΝΑ 4-1-12. Εισαγωγή ενός νέου αντικειμένου από μια αποθηκευμένη κλάση

Κληρονομικότητα-Οφέλη

Η δημιουργία μιας νέας κλάσης βασισμένης σε μια άλλη κλάση ονομάζεται κληρονομικότητα. Η κληρονομικότητα στις περισσότερες αντικειμενοστραφείς γλώσσες είναι πιο πολύπλοκη απ' ότι στο Alice. Η βασική ιδέα είναι η ίδια-πρόσθεση λειτουργικότητας με τον ορισμό νέων μεθόδων για ένα νέο είδος κληρονόμου κλάσης. Η κληρονομικότητα θεωρείται ένα δυνατό σημείο για τις αντικειμενοστραφείς γλώσσες προγραμματισμού γιατί επιτρέπει να γράψετε μια φορά κώδικα και να τον χρησιμοποιήσετε και σε άλλα προγράμματα.

Ένα άλλο όφελος από την δημιουργία νέων κλάσεων είναι η δυνατότητα να μοιραστείτε τον κώδικα με άλλους σε ομαδικές εργασίες. Για παράδειγμα, αν δουλεύετε σε ένα πρόγραμμα του Alice σαν ομάδα, κάθε άτομο μπορεί να γράψει μεθόδους επιπέδου κλάσης για ένα αντικείμενο του κόσμου. Μετά κάθε μέλος της ομάδας μπορεί να αποθηκεύσει τη νέα κλάση. Τα αντικείμενα της νέας κλάσης προστίθενται σε ένα κόσμο που έχει φτιαχτεί από μια ομάδα με αποτέλεσμα να γίνει ένα ομαδικό έργο. Στον "πραγματικό κόσμο", οι επαγγελματίες των υπολογιστών, δουλεύουν γενικά σε ομαδικά έργα. Η δημιουργία λογισμικού με συνεργασία είναι ο πιο συχνός τρόπος που λειτουργούν οι ομάδες των ανιματέρ στα στούντιο.

Οδηγίες για το γράψιμο μεθόδων επιπέδου κλάσης

Οι μέθοδοι επιπέδου κλάσης είναι σημαντικό χαρακτηριστικό του Alice. Φυσικά για να αποφύγετε κάποια λανθασμένη εφαρμογή των μεθόδων, σας παρέχουμε οδηγίες.

1. Δημιουργήστε πολλές διαφορετικές μεθόδους επιπέδου κλάσης. Είναι χρήσιμες. Μερικές κλάσεις του Alice έχουν ήδη καθορισμένες μερικές μεθόδους. Για παράδειγμα η κλάση Lion έχει μεθόδους όπως *startStance, walkForward, completeWalk, roar, charge.* Η εικόνα 4-1-13 δείχνει μια λίστα της κλάσης Lion (από την βιβλιοθήκη web) που περιλαμβάνει τις μεθόδους και τους ήχους της.

ΕΙΚΟΝΑ 4-1-13. Μέθοδοι επιπέδου κλάσης και ήχοι για την κλάση Lion

2. Παίξτε ένα ήχο σε μια μέθοδο επιπέδου κλάσης MONO EAN ο ήχος έχει εισαχθεί για το αντικείμενο. Εάν ο ήχος έχει εισαχθεί για το αντικείμενο και το αντικείμενο έχει αποθηκευτεί ως νέα κλάση, ο ήχος σώζεται μαζί με το αντικείμενο. Ο ήχος μπορεί να παιχτεί οπουδήποτε σε οποιονδήποτε κόσμο όπου υπάρχει το συγκεκριμένο αντικείμενο. Από την άλλη μεριά, εάν ο ήχος έχει εισαχθεί για τον κόσμο, δεν έχει αποθηκευτεί μαζί με το αντικείμενο και δεν μπορείτε να βασιστείτε ότι αυτός ο ήχος θα είναι διαθέσιμος και σε άλλο κόσμο.

ΕΙΚΟΝΑ 4-1-14. Η παγοδρόμος θα γυρίσει γύρω από τον πιγκουΐνο

3. Μην χρησιμοποιείτε εντολές για άλλα αντικείμενα από μια άλλη μέθοδο επιπέδου κλάσης. Οι μέθοδοι επιπέδου κλάσης ορίζονται για μια

συγκεκριμένη κλάση. Αποθηκεύουμε τα αντικείμενα ως μια νέα κλάση και τα χρησιμοποιούμε σε άλλους κόσμους. Δεν μπορούμε να βασιστούμε στην παρουσία άλλων αντικειμένων σε προγράμματα άλλων κόσμων. Για παράδειγμα, ένας πιγκουΐνος (Animals) προστίθεται σε ένα χιονισμένο σκηνικό, όπως στην εικόνα 4-1-14. Γράφουμε μια μέθοδο επιπέδου κλάσης με το όνομα *skateAround,* όπου ο πιγκουΐνος καλείται σε δυο από τις εντολές. Εικόνα 4-1-15. Εάν η cleverSkater με την μέθοδο *skateAround,* αποθηκευτεί σαν μια νέα κλάση και μετά ένα αντικείμενο αυτής της κλάσης τοποθετηθεί σε ένα άλλο κόσμο όπου δεν υπάρχει πιγκουΐνος, το Alice θα ανοίξει ένα παράθυρο λάθους για να σας πει ότι λείπει ένα αντικείμενο. Το λάθος θα είναι ότι η cleverSkater δεν μπορεί να κάνει skate γύρω από τον πιγκουΐνο γιατί δεν υπάρχει στον κόσμο!

Σημείωση: Πιθανές εξαιρέσεις στις οδηγίες #4 είναι τα αντικείμενα κάμερα και κόσμος που είναι πάντα παρόν.

ΕΙΚΟΝΑ 4-1-15. Λανθασμένο παράδειγμα: οι εντολές καθορίζουν ένα άλλο αντικείμενο σε μια μέθοδο επιπέδου κλάσης

Μια μέθοδος επιπέδου κλάσης με ένα αντικείμενο ως παράμετρο

Τι γίνεται σε περίπτωση που θέλετε να γράψετε μια μέθοδο επιπέδου κλάσης στην οποία να περιέχονται άλλα αντικείμενα; Η λύση είναι να χρησιμοποιήσετε παράμετρο αντικείμενο στην μέθοδο. Ας χρησιμοποιήσουμε το ίδιο παράδειγμα με παραπάνω, όπου θέλαμε μια cleverSkater να κάνει skate γύρω από ένα άλλο αντικείμενο. Η μέθοδος *skateAround,* μπορεί να τροποποιηθεί ώστε να χρησιμοποιεί μια παράμετρο, που αυθαιρέτως να ονομάζεται " *whichObject* " *(*ποιο αντικείμενο*)*, όπως φαίνεται στην εικόνα 4-1-16. Η παράμετρος *whichObject* χρησιμοποιείται μόνο για να κρατήσει την θέση, δεν είναι πραγματικό αντικείμενο, οπότε δεν χρειάζεται να ανησυχούμε για ένα συγκεκριμένο αντικείμενο (όπως ένας πιγκουΐνος) αν βρίσκεται στον κόσμο. Το Alice δεν θα επιτρέψει να κληθεί η μέθοδος *skateAround* εάν δεν αντιστοιχηθεί ένα αντικείμενο στην παράμετρο *whichObject.* Έτσι είμαστε σίγουροι ότι κάποιο είδος αντικειμένου θα υπάρχει εκεί ώστε να εκτελεστεί η κίνηση.

ΕΙΚΟΝΑ 4-1-16. Χρήση ενός αντικειμένου ως παράμετρο σε μια μέθοδο επιπέδου κλάσης

Τεστ

Εφόσον δημιουργήσατε και αποθηκεύσατε μια νέα κλάση, θα πρέπει να ελεγχθεί σε ένα νέο κόσμο. Η αρχική σκηνή δόθηκε στην εικόνα 4-1-14. Ένα παράδειγμα προγράμματος φαίνεται στην εικόνα 4-1-17. Σε αυτό το τεστ, καλέσαμε τις μεθόδους *skate*, *spin* και *skateAround* για να εξετάσουμε την κάθε μέθοδο.

ΕΙΚΟΝΑ 4-1-17. Ένα απλό πρόγραμμα

Μυστικά και Τεχνικές 4

Ορατά και αόρατα αντικείμενα

Οι ιδιότητες των αντικειμένων μερικές φορές χρησιμοποιούνται στα παιχνίδια και στα κινούμενα σχέδια ώστε να επιτευχθεί ένα ειδικό εφέ, όπως να γίνει ένα αντικείμενο ορατό ή αόρατο. Σε αυτή την ενότητα θα μελετήσουμε τεχνικές και παραδείγματα αλλαγής της ορατότητας των αντικειμένων.

Η ιδιότητα opacity (αδιαφάνειας)

Το ακόλουθο παράδειγμα αλλάζει την opacity ενός ψαριού σε ένα ωκεανό. (Αδιαφάνεια είναι το πόσο αδιαφανές είναι κάτι: πόσο δύσκολο είναι να δείτε μέσα από αυτό). Η εικόνα Τ-4-1 δείχνει μια υδρόβια σκηνή. Αυτός ο κόσμος δημιουργείται εύκολα προσθέτοντας ένα oceanFloor (Ocean) και ένα ψάρι lilfish σε χρώμα λιλά (Ocean). Προαιρετικά αντικείμενα- φύκια και κοράλλια προστίθενται από τον φάκελο oceanFloor στο cd ή στην web gallery).

Το lilfish κολυμπάει για να βρει τροφή και το αγαπημένο του φαγητό είναι τα φύκια. Εντολές για να στραφεί το ψάρι προς τα φύκια και μετά να κολυμπήσει προς αυτά φαίνονται στην εικόνα Τ-4-2. Η εντολή για την *wiggletail* (κίνηση της ουράς) είναι μια μέθοδος που φαίνεται στην εικόνα Τ-4-2(b), η οποία κάνει το ψάρι να κουνάει την ουρά του δεξιά-αριστερά.

Όσο το lilfish κινείται προς τα φύκια, θα απομακρύνεται ταυτόχρονα από την κάμερα. Έτσι πρέπει να εξαφανίζεται σιγά, σιγά επειδή το νερό συσκοτίζει την όραση μας όσο απομακρύνονται τα αντικείμενα.

ΕΙΚΟΝΑ Τ-4-1. Ένα σκηνικό ωκεανού με ένα lilfish

ΕΙΚΟΝΑ Τ-4-2(a). Κώδικας για να κινηθεί το lilfish προς το φύκι

ΕΙΚΟΝΑ Τ-4-2(b). Η μέθοδος *wiggletail*

ΕΙΚΟΝΑ Τ-4-3. Τοποθέτηση της opacity στον συντάκτη

ΕΙΚΟΝΑ Τ-4-4. Ο κώδικας τώρα περιέχει ένα σύνολο εντολών opacity

Μπορούμε να κάνουμε το lilfish λιγότερο ορατό αλλάζοντας την ιδιότητα της opacity. Όσο μειώνεται η opacity, το αντικείμενο γίνεται λιγότερο διακριτό (πιο δύσκολο να το δείτε). Για να γράψετε μια εντολή αλλαγής της opacity, κάντε κλικ στην καρτέλα properties του lilfish και τοποθετήστε την opacity στον συντάκτη. Από το μενού που βγαίνει επιλέξτε την επί τοις εκατό αδιαφάνεια, όπως φαίνεται στην εικόνα Τ-4-3.

Ο κώδικας φαίνεται στην εικόνα Τ-4-4.

Όταν εκτελείται το πρόγραμμα, το lilfish θα γίνει λιγότερο ορατό, όπως φαίνεται στην εικόνα Τ-4-5. Στο 0% το αντικείμενο θα εξαφανιστεί. Αυτό δεν σημαίνει ότι το αντικείμενο έχει διαγραφεί. Είναι ακόμη μέρος του κόσμου αλλά δεν φαίνεται στην οθόνη.

Η ιδιότητα isShowing (είναι εμφανής)

Κάθε αντικείμενο έχει μια ιδιότητα που καλείται *isShowing.* Μόλις ένα αντικείμενο προστεθεί στον κόσμο, είναι ορατό στον σκηνικό και η ιδιότητα του *isShowing* είναι *true.* Η αλλαγή της τιμής αυτής της ιδιότητας είναι ιδιαίτερα χρήσιμη σε παιχνίδια όπως προγράμματα όπου θέλετε να σηματοδοτήσετε το τέλος του παιχνιδιού. Η εικόνα Τ-4-6 παρουσιάζει την ιδιότητα *isShowing* για το αντικείμενο "You Won!" (κέρδισες!). Κάνοντας την ιδιότητα *false* το αντικείμενο "You Won!" γίνεται αόρατο όπως στην εικόνα Τ-4-7. (Γι αυτό τον κόσμο χρησιμοποιήσαμε το αντικείμενο bottleThrow από τον φάκελο Amusement Park).

ΕΙΚΟΝΑ Τ-4-5. Το lilfish γίνεται αόρατο

ΕΙΚΟΝΑ Τ-4-6. Η ιδιότητα *isShowing* είναι *true* και το "You Won!" είναι ορατό

ΕΙΚΟΝΑ Τ-4-7. Η ιδιότητα *isShowing* είναι *false* και το "You Won!" είναι αόρατο

Όταν η ιδιότητα *isShowing* γίνει *false,* το αντικείμενο υπάρχει ακόμη στον κόσμο, απλά δεν φαίνεται στην οθόνη. Το αντικείμενο μπορεί να ξαναεμφανιστεί βάζοντας ξανά την ιδιότητα *isShowing* γίνει *true.*

Σε αυτό το παράδειγμα, θέλουμε το κείμενο να φανεί όταν ο παίχτης κερδίσει. Για να δημιουργήσουμε μια εντολή που θέτει την ιδιότητα *isShowing* σε *true*, τοποθετούμε το πλακίδιο της ιδιότητας μέσα στον κόσμο και επιλέγουμε *true*. Το αποτέλεσμα φαίνεται στην εικόνα Τ-4-8.

ΕΙΚΟΝΑ Τ-4-8. Μια εντολή για να θέσετε την *isShowing true*

Σχέση των ιδιοτήτων *isShowing* και *opacity*

Οι ιδιότητες *isShowing* και *opacity* είναι διαφορετικές αλλά συσχετίζονται μεταξύ τους. Η ιδιότητα *isShowing* είναι αυστηρά *true* ή *false.* Η *opacity* είναι μια κλίμακα. Όταν η *opacity* είναι 0% παρόλο που το αντικείμενο γίνεται αόρατο, το Alice δεν κάνει αυτόματα την ιδιότητα *isShowing false.* Αντίστοιχα όταν η *isShowing* γίνει *false,* η *opacity* δεν γίνεται 0%.

Μια καλή συμβουλή είναι: "Να είστε συνεπής." Εάν χρησιμοποιείτε την *isShowing* στο πρόγραμμα σας για να καθορίσετε την ορατότητα, τότε μην χρησιμοποιείτε την *opacity* για να ελέγξετε αν το αντικείμενο είναι ορατό. Αντίστοιχα, αν χρησιμοποιείτε την *opacity* για να καθορίσετε την ορατότητα, μην χρησιμοποιείτε την *isShowing* για να ελέγξετε αν το αντικείμενο είναι ορατό.

Ασκήσεις

4-1 Ασκήσεις

1.Εμπλουτισμένη cleverSkater

Δημιουργήστε μια πιο έξυπνη παγοδρόμο από αυτή της ενότητας 4-1. Πέρα από τις *μεθόδους skateForward, spin* και *skateAround,* δημιουργήστε τις μεθόδους επιπέδου κλάσης *skateBackward* και *jump.* Στην *skateBackward* η παγοδρόμος πρέπει να κάνει τα ίδια με αυτά που έκανε στην *skateForward* αλλά να κινείται προς τα πίσω. Στην *jump* η παγοδρόμος πρέπει να κινηθεί μπροστά, να σηκώσει το ένα πόδι μετά να πηδήξει στον αέρα και μετά κάτω για να προσγειωθεί ομαλά στον πάγο και να κατεβάσει το πόδι πίσω στην αρχική του θέση. Αποθηκεύστε την νέα κλάση ως *EnhancedCleverSkater.*

Δοκιμάστε την νέα σας κλάση αρχίζοντας ένα νέο κόσμο με μια παγωμένη λίμνη. Τοποθετήστε μια κλάση *EnhancedCleverSkater* στο κόσμο, ένα πιγκουΐνο και μια πάπια.

(α) Καλέστε κάθε μια από τις μεθόδους που γράψατε.

(β) Μετά καλέστε την μέθοδο *skateAround*-για να κάνετε την παγοδρόμο να κάνει skate γύρω από τον πιγκουΐνο και μετά από την πάπια. (Αυτό απαιτεί δύο κλήσεις της μεθόδου)

2.Συνδιασμός κλειδαριάς

Δημιουργήστε ένα κόσμο με ένα *comboLock* (κλειδαριά με συνδυασμό). Δημιουργήστε τέσσερις μεθόδους επιπέδου κλάσης *leftOne, rightOne, leftRevolution, rightRevolution*-αυτό γυρίζει την επιλογή 1 νούμερο αριστερά, 1 νούμερο δεξιά, 1 περιστροφή αριστερά και μια περιστροφή δεξιά αντίστοιχα. Μετά δημιουργήστε μια μέθοδο επιπέδου κλάσης που ονομάζεται *open* που ανοίγει και μια άλλη *close* που κλείνει την κλειδαριά.

Υπόδειξη: Μία θέση της επιλογής είναι 1/40 της περιστροφής. Χρησιμοποιήστε το στυλ *endGently* για να κάνετε την κίνηση πιο ρεαλιστική. Μετονομάστε την κλάση comboLock σε TurningcomboLock και αποθηκεύστε την ως νέα κλάση.

ΕΙΚΟΝΑ

3.Χορός κοτόπουλου

Ξεκινήστε με ένα κοτόπουλο, δημιουργήστε μια μέθοδο για περπάτημα *walk* όπου το κοτόπουλο θα κάνει μια ρεαλιστική κίνηση. Ένα βήμα αριστερά, ένα δεξιά. Δημιουργήστε μια δεύτερη μέθοδο για να κάνετε το κοτόπουλο να χορεύει *(funkyChicken),* όπου το κοτόπουλο περπατάει, στριφογυρίζει το σώμα του σε πολλές διαφορετικές κατευθύνσεις! Αποθηκεύστε την νέα κλάση με το όνομα *CoolChicken.* Δημιουργήστε ένα νέο κόσμο και προσθέστε ένα *CoolChicken.* Στην *my first method,* καλέστε την *walk* και την *funkyChicken.* Παίξτε ένα ήχο ή χρησιμοποιήστε μια εντολή *say* για να συνοδεύσει την κίνηση.

ΕΙΚΟΝΑ

4.Εξάσκηση Ninja

Δημιουργήστε ένα κόσμο με ένα evilNinja και γράψτε μεθόδους για τις κινήσεις ενός Ninja. Για παράδειγμα μπορείτε να γράψετε *rightJab, leftJab, kickLeft, kickRight, leftSpin* και *rightSpin.* Κάθε μέθοδος πρέπει να περιέχει πάνω από μια εντολή. Για παράδειγμα στην *kickLeft,* το κάτω μέρος του αριστερού ποδιού πρέπει να γυρίσει και να στριφογυρίσει το άκρο την ίδια στιγμή που ολόκληρο το πόδι θα κλωτσήσει αριστερά. Αποθηκεύστε ως νέα κλάση TrainedNinja. Ξεκινήστε ένα νέο κόσμο και προσθέστε δύο αντικείμενα trainedNinja. Δημιουργήστε ένα κινούμενο σχέδιο όπου οι δύο Ninja εξασκούνται αντιμετωπίζοντας ο ένας τον άλλο.

Περίληψη

Σε αυτό το κεφάλαιο είδαμε πώς να γράφουμε δικές μας μεθόδους και πώς να χρησιμοποιούμε παραμέτρους για να στέλνουμε πληροφορίες σε μια μέθοδο όταν καλείται. Ένα πλεονέκτημα της χρήσης μεθόδων είναι ότι ο προγραμματιστής μπορεί να αποσυνθέσει μια εντολή. Επίσης οι μέθοδοι κάνουν ευκολότερη την διόρθωση σφαλμάτων του κώδικα.

Οι παράμετροι χρησιμοποιούνται για να επικοινωνήσουν οι τιμές από μια μέθοδο σε μια άλλη. Σε μια μέθοδο η παράμετρος χρησιμοποιείται ως θέση για να τοποθετηθεί η τιμή ενός συγκεκριμένου τύπου. Οι τιμές που στέλνονται σε μια μέθοδο ονομάζονται arguments (ορίσματα). Όταν ένα όρισμα στέλνεται σε μια μέθοδο, η παράμετρος αναπαριστά αυτό το όρισμα στις εντολές της μεθόδου. Τα παραδείγματα αυτού του κεφαλαίου περιείχαν παραμέτρους για αντικείμενο, ήχο, χαρακτήρες και αριθμούς. Οι παράμετροι σας επιτρέπουν να γράψετε μια μέθοδο και να την χρησιμοποιήσετε πολλές φορές με διαφορετικά αντικείμενα, ήχους, αριθμούς και άλλα είδη τιμών.

Κατά κάποιο τρόπο οι μέθοδοι επιπέδου κλάσης μπορούν να θεωρηθούν ως επέκταση των χαρακτηριστικών ενός αντικειμένου. Εφόσον δημιουργηθούν νέες μέθοδοι επιπέδου κλάσης, μπορεί να αποθηκευτεί μια νέα κλάση. Η νέα κλάση έχει διαφορετικό όνομα και έχει όλες τις νέες μεθόδους άλλα και τις παλιές. Κληρονομεί τις ιδιότητες και τις ενέργειες της αρχικής κλάσης αλλά ορίζει περισσότερα πράγματα απ΄ ότι η αρχική κλάση. Ένα όφελος είναι ότι μπορείτε να χρησιμοποιήσετε αντικείμενα της νέας κλάσης ξανά και ξανά σε νέους κόσμους. Αυτό σας επιτρέπει να εκμεταλλευτείτε τις μεθόδους που γράψατε χωρίς να τις ξαναγράψετε.

Η βηματική διάσπαση είναι μια σχεδιαστική τεχνική όπου μια πολύπλοκη διεργασία αποσυντίθεται σε μικρά κομμάτια και μετά κάθε κομμάτι αποσυντίθεται περισσότερο-ώσπου ολόκληρη η διεργασία να ορίζεται από μικρές ενέργειες. Όλες οι απλές ενέργειες λειτουργούν μαζί για να εκτελεστεί η διεργασία.

Σημαντικά θέματα σε αυτό το κεφάλαιο

- Για να εκτελέσετε μια μέθοδο, η μέθοδος πρέπει να κληθεί.
- Οι παράμετροι χρησιμοποιούνται για επικοινωνία με μια μέθοδο.
- Σε μια κλήση μιας μεθόδου, μια τιμή που στέλνεται σε μια παράμετρο μιας μεθόδου, ονομάζεται όρισμα.
- Μια παράμετρος μπορεί να αναπαριστά μια τιμή συγκεκριμένου τύπου. Οι τύποι των τιμών των παραμέτρων είναι αντικείμενα, λογικές τιμές, αριθμοί, χρώματα, ήχοι, χαρακτήρες και άλλα.
- Μια νέα κλάση μπορεί να δημιουργηθεί ορίζοντας μεθόδους επιπέδου κλάσης και μετά αποθηκεύοντας την κλάση με ένα νέο όνομα.
- Η κληρονομικότητα είναι μια αντικειμενοστραφής αρχή όπου μια νέα κλάση βασίζεται σε μια ήδη υπάρχουσα κλάση.
- Οι μέθοδοι επιπέδου κλάσης μπορούν να γραφούν ώστε να δέχονται για παραμέτρους αντικείμενα. Αυτό σας επιτρέπει να γράψετε μια μέθοδο επιπέδου κλάσης και να περάσετε σε άλλο αντικείμενο. Το αντικείμενο για το οποίο γράφεται η μέθοδος αλληλεπιδρά με το αντικείμενο παράμετρο.

Κεφάλαιο 5
Αλληλεπίδραση: Γεγονότα και χειρισμός γεγονότων

Ο αληθινός κόσμος γύρω μας είναι αλληλεπιδρών (interactive). Μια συζήτηση μεταξύ της Alice και της Queen είναι μια “πάρε δώσε” φόρμα αλληλεπίδρασης. Καθώς αλληλεπιδρούμε με αντικείμενα στο κόσμο μας, συχνά τους δίνουμε οδηγίες. Για παράδειγμα, αλλάζουμε το κανάλι στην τηλεόραση στέλνοντας ένα σήμα από το τηλεκοντρόλ. Πατάμε ένα πλήκτρο σε ένα τηλεχειριστήριο ενός παιχνιδιού για να κάνουμε τον χαρακτήρα στο παιχνίδι να πηδήξει και να πυροβολήσει ώστε να ξεφύγει τον κίνδυνο.

Εστιάσαμε στα προγράμματα που δεν είχαν αλληλεπίδραση-βλέπαμε τα αντικείμενα να εκτελούν κινήσεις και ενέργειες σαν να είναι ταινία. Ήρθε η ώρα να δούμε πώς να δημιουργήσουμε αλληλεπιδρόντα προγράμματα στο Alice-όπου τα αντικείμενα στην σκηνή θα ανταποκρίνονται στα κλικ του ποντικιού και στο πάτημα ενός πλήκτρου. Σε αυτό το κεφάλαιο θα δούμε πώς να φτιάχνουμε προγράμματα με αλληλεπίδραση.

Πολλά από τα προγράμματα έχουν ως βάση τον υπολογιστή. Ο υπολογιστής δουλεύει όσο ο προγραμματιστής το ορίσει. Ο προγραμματιστής καθορίζει την σειρά των πράξεων και ελέγχει την ροή του προγράμματος. Παρόλα αυτά πολλά προγράμματα έχουν ως βάση τον χρήστη. Δηλαδή ο χρήστης-και όχι ο προγραμματιστής- καθορίζει την σειρά των πράξεων. Ο χρήστης κάνει κλικ ή πατάει ένα πλήκτρο και στέλνει ένα σήμα στο Alice για το τι να κάνει μετά. Το κλικ του ποντικιού ή το πάτημα ενός πλήκτρου είναι ένα γεγονός (event). Ένα γεγονός είναι κάτι που γίνεται . Σε απάντηση προς το γεγονός, εκτελείται μια πράξη (ή μια αλληλουχία πράξεων). Λέμε ότι το γεγονός προκαλεί μια απάντηση.

Η ενότητα 5-1 εστιάζει στους μηχανισμούς με τους οποίους ο χρήστης δημιουργεί ένα γεγονός και πως το πρόγραμμα ανταποκρίνεται στο γεγονός. Φυσικά όλο αυτό χρειάζεται σχεδιασμό και τακτοποίηση. Πρέπει να πούμε στο Alice να ακούσεί ένα συγκεκριμένο είδος γεγονότος και μετά τι να κάνει όταν συμβεί το γεγονός. Αυτό σημαίνει ότι χρειάζεται να γράψουμε μεθόδους που περιγράφουν τις ενέργειες που πρέπει να κάνουν τα αντικείμενα ως απάντηση σε ένα γεγονός.

Ελπίζουμε ότι θα βρείτε εύκολο τον χειρισμό γεγονότων στο Alice.

5-1 Αλληλεπιδρών προγραμματισμός

Έλεγχος της ροής

Η υλοποίηση ενός προγράμματος με αλληλεπίδραση έχει μια μεγάλη διαφορά από την υλοποίηση ενός απλού προγράμματος (όπως οι ταινίες που γράψαμε στο προηγούμενο κεφάλαιο). Η διαφορά είναι στο πως ελέγχεται η ακολουθία των πράξεων. Σε ένα πρόγραμμα χωρίς αλληλεπίδραση, η ακολουθία των πράξεων είναι προκαθορισμένη από τον προγραμματιστή. Ο προγραμματιστής σχεδιάζει ολόκληρη την διάταξη σκηνών και μετά γράφει τον κώδικα του προγράμματος. Εφόσον το πρόγραμμα σχεδιάζεται και

ελέγχεται, μετά κάθε φορά που το πρόγραμμα εκτελείται, θα συμβαίνει η ίδια ακολουθία ενεργειών. Σε ένα πρόγραμμα με αλληλεπίδραση η ακολουθία των ενεργειών αποφασίζεται την ώρα που τρέχει το πρόγραμμα, όταν:

- Ο χρήστης κάνει κλικ ή πατήσει ένα πλήκτρο του πληκτρολογίου.
- Τα αντικείμενα κινούνται στην σκηνή (τυχαία ή καθοδηγούμενα από το χρήστη) για να δημιουργήσουν κάποια κατάσταση, όπως μια σύγκρουση.

Γεγονότα

Κάθε φορά που ο χρήστης κάνει κλικ με το ποντίκι ή πατάει ένα πλήκτρο, δημιουργείται ένα γεγονός που προκαλεί μια απάντηση. Τα αντικείμενα της σκηνής μπορεί να μετακινούνται σε θέσεις που προκαλούν μια απάντηση. Κάθε φορά που εκτελείται το πρόγραμμα, διαφορετικοί επιδράσεις των χρηστών ή διαφορετικές πράξεις των αντικειμένων μπορεί να συμβούν και το κινούμενο σχέδιο να είναι διαφορετικό από ορισμένες προηγούμενες εκτελέσεις. Για παράδειγμα σε ένα videogame που εξομοιώνει ένα αγώνα αυτοκινήτου, όπου ο παίχτης οδηγεί ένα αυτοκίνητο, οι σκηνές του παιχνιδιού καθορίζονται από το εάν ο παίχτης είναι ικανός οδηγός σε δύσκολες πίστες της σκηνής.

Μέθοδοι για χειρισμό γεγονότων

Πως επηρεάζουν τα γεγονότα το τι θα κάνετε σαν προγραμματιστής; Πρέπει να σκεφτείτε όλα τα πιθανά γεγονότα και να σχεδιάσετε τι θα συμβεί-απαντήσεις στα γεγονότα. Μέθοδοι γράφονται για να εκτελεστούν οι απαντήσεις. Τέλος, το γεγονός πρέπει να διασυνδεθεί με την μέθοδο της απάντησης. Η μέθοδος καλείται μέθοδος για χειρισμό γεγονότων.

Όταν συμβεί ένα γεγονός και κληθεί μια μέθοδος για χειρισμό γεγονότων, η θέση των αντικειμένων της σκηνής μπορεί να είναι η ίδια με την τελευταία φορά, μπορεί και όχι. Αυτό γίνεται γιατί οι ενέργειες του χρήστη μπορεί να αλλάξουν το σκηνικό και την θέση των αντικειμένων μεταξύ των κλήσεων των μεθόδων για χειρισμό γεγονότων.

Παράδειγμα ελέγχου μέσω πληκτρολογίου

Ξεκινάμε με ένα ακροβατικό θέαμα στον αέρα ενός εξομοιωτή πτήσης. Η αρχική σκηνή, όπως φαίνεται στην εικόνα 5-1-1, αποτελείται από το αεροσκάφος (Vehicles) να βρίσκεται στον αέρα και μερικά αντικείμενα στο έδαφος (σπίτι, αχυρώνας από τους καταλόγους Buildings και Farm). Ένα σύστημα πλοήγησης θα επιτρέπει στον χρήστη να γίνει ο πιλότος. Το αεροσκάφος έχει ρυθμίσεις που επιτρέπουν στον πιλότο να κάνει μανούβρες το αεροπλάνο μπροστά, αριστερά και δεξιά. Θέλουμε να προγραμματίσουμε το αεροσκάφος να εκτελέσει ένα δημοφιλές ακροβατικό κατόρθωμα-ένα κυλινδρικό ελιγμό. Στις ασκήσεις στο τέλος του κεφαλαίου, μπορούν να προστεθούν και άλλα ακροβατικά.

ΕΙΚΟΝΑ 5-1-1. Αρχική σκηνή

Εισαγωγή δεδομένων

Η ιδέα ενός εξομοιωτή πτήσης είναι να επιτρέψει στο χρήστη να αλληλεπιδράσει με το αεροσκάφος. Ο χρήστης δίνει πληροφορία στην είσοδο που στέλνει ένα σήμα στο πρόγραμμα ώστε να κάνει μια συγκεκριμένη κίνηση. Αυτή η πληροφορία μπορεί να είναι το πάτημα μιας σειράς πλήκτρων στο πληκτρολόγιο. Για παράδειγμα, τα βέλη του πληκτρολογίου μπορούν να χρησιμοποιηθούν έτσι ώστε το κάθε ένα από αυτά να αντιστοιχεί σε μια κατεύθυνση. Φυσικά η είσοδος μπορεί να τροφοδοτηθεί και με κλικ του ποντικιού ή με ένα τηλεχειριστήριο ενός παιχνιδιού. Σε αυτό το παράδειγμα θα χρησιμοποιήσουμε το πληκτρολόγιο και το ποντίκι για να εισάγουμε δεδομένα στο πρόγραμμα.

Στον εξομοιωτή πτήσης, τα βέλη του πληκτρολογίου και το κενό διάστημα θα χρησιμοποιηθούν από τον χρήστη ως είσοδος στο πρόγραμμα. Εάν ο χρήστης πατήσει το επάνω βέλος, το αεροσκάφος θα κινηθεί μπροστά. Εάν ο χρήστης πατήσει το δεξί ή το αριστερό βέλος, το αεροσκάφος θα κινηθεί δεξιά ή αριστερά. Για την κυλινδρικό ακροβατικό κατόρθωμα θα χρησιμοποιήσουμε το κενό διάστημα. Η επιλογή αυτών των πλήκτρων είναι αυθαίρετη-θα μπορούσαν να χρησιμοποιηθούν άλλα κουμπιά.

Σχεδιασμός-διάταξη σκηνών

Είμαστε έτοιμοι να σχεδιάσουμε το πρόγραμμα του εξομοιωτή πτήσης-το σύνολο των εντολών που θα εξηγούν στο Alice πώς να εκτελέσει τις κινήσεις. Κάθε φορά που ο χρήστης πατάει ένα βέλος ή το κενό διάστημα, γεννιέται ένα γεγονός. Το πρόγραμμα αποτελείται από δυο μεθόδους που χρησιμοποιούνται ως απάντηση στο γεγονός. Για να απλοποιήσουμε τα πράγματα ας επικεντρωθούμε σε δυο γεγονότα:πατιέται το κενό διάστημα για το κυλινδρικό ακροβατικό και πατιέται το πάνω βελάκι για να κινηθεί το αεροσκάφος μπροστά. Δυο διατάξεις σκηνών είναι απαραίτητες, όπως φαίνεται. Σημειώστε ότι ο ήχος είναι προαιρετικός και μπορεί να μην χρησιμοποιηθεί.

Event: spacebar press

Response:

Do together

roll biplane a full revolution

play biplane engine sound

Event: Up arrow key press

Response:

Do together

move biplane forward

play biplane engine sound

Event (γεγονός): πάτημα του κενού διαστήματος

Response (απάντηση):

Κάνε ταυτόχρονα

στριφογύρισε το αεροσκάφος μια πλήρη περιστροφή

παίξε τον ήχο του κινητήρα του αεροσκάφους

Event (γεγονός): πάτημα του επάνω βέλους

Response (απάντηση):

Κάνε ταυτόχρονα

κίνησε το αεροσκάφος μπροστά

παίξε τον ήχο του κινητήρα του αεροσκάφους

Μέθοδοι για απάντηση σε ένα γεγονός

Το μόνο αντικείμενο που επηρεάζεται από το πάτημα των κουμπιών είναι το αεροσκάφος, έτσι οι μέθοδοι μπορεί να είναι επιπέδου κλάσης. Δύο μέθοδοι θα γραφούν η *flyForward* (κίνηση εμπρός) και η *barrel* (κυλινδρικό ακροβατικό). Η *flyForward* θα χειρίζεται ένα γεγονός που προέκυψε από το πάτημα του επάνω βέλους, κινώντας το αεροπλάνο εμπρός όπως φαίνεται στην εικόνα 5-1-2. Η μέθοδος *barrel* θα χειρίζεται ένα γεγονός από το πάτημα του κενού διαστήματος, στριφογυρίζοντας το αεροσκάφος μια ολόκληρη περιστροφή, όπως φαίνεται στην εικόνα 5-1-3.

ΕΙΚΟΝΑ 5-1-2. Η μέθοδος *flyForward*

ΕΙΚΟΝΑ 5-1-3. Η μέθοδος *barrel*

Σε αυτές τις μεθόδους, ένας ήχος παίζει μαζί με την κίνηση. Η διάρκεια της κίνησης του αεροσκάφους καθορίστηκε να είναι περίπου όσο η διάρκεια του ήχου σε δευτερόλεπτα. Όπως τονίστηκε παραπάνω ο ήχος είναι ένα ωραίο χαρακτηριστικό αλλά μπορεί να παραληφθεί. Εάν χρησιμοποιηθεί, ήχος πρέπει να εισαχθεί για το αεροπλάνο. (Η εισαγωγή ενός αρχείο ήχου, παρουσιάστηκε στο κεφάλαιο 4, ενότητα 2).

Διασύνδεση των γεγονότων με τις μεθόδους

Κάθε μέθοδος πρέπει να διασυνδεθεί με ένα γεγονός που θα χρησιμοποιηθεί για να προκαλέσει την μέθοδο ως απάντηση, ο συντάκτης γεγονότων χρησιμεύει για την διασύνδεση. Ο συντάκτης γεγονότων φαίνεται στην εικόνα 5-1-4. Όπως ξέρετε, το Alice δημιουργεί μια διασύνδεση *When the world starts* (όταν ξεκινάει ο κόσμος-ένα γεγονός) και *World.my first method,* όπως φαίνεται στην εικόνα 5-1-4.

ΕΙΚΟΝΑ 5-1-4. Συντάκτης γεγονότων

Στον εξομοιωτή πτήσης, δύο γεγονότα (το πάτημα του βέλους και το κενού διαστήματος) πρέπει να διασυνδεθούν στις αντίστοιχες μεθόδους *(flyForward, barrel).* Πρώτα δημιουργήστε ένα νέο γεγονός κάνοντας κλικ στο πλήκτρο “create new event” και μετά επιλέξτε το γεγονός από το μενού. Στην εικόνα 5-1-5, επιλέγεται το *When a key is typed* (*όταν πατηθεί ένα πλήκτρο.*

ΕΙΚΟΝΑ 5-1-5. Δημιουργία ενός γεγονότος πατήματος πλήκτρου

Στην εικόνα 5-1-6, έχει προστεθεί στον συντάκτη γεγονότων ένα γεγονός για το πάτημα ενός πλήκτρου. Τα πλακίδια "*any key*" και "*nothing*" είναι απλά για να κρατηθεί η θέση που πρέπει να αντικατασταθούν. Για να πείτε στο Alice να χρησιμοποιήσει το πάνω βέλος, πατήστε πάνω στο πλακίδιο "*any key*" και επιλέξτε το επάνω "*Up*" από το μενού.

ΕΙΚΟΝΑ 5-1-6. Ορισμός του επάνω βέλους

Τώρα που το Alice έχει ενημερωθεί ότι μπορεί να πατηθεί το επάνω βέλος, πρέπει να την ενημερώσουμε για το τι θα κάνει αν γίνει αυτό το γεγονός. Όπως φαίνεται στην εικόνα 5-1-7, κάνετε κλικ στο "*Nothing*" και μετά επιλέξτε το αεροσκάφος και την μέθοδο *flyForward.*

ΕΙΚΟΝΑ 5-1-7. Διασύνδεση της μεθόδου με το γεγονός

Η διαδικασία επαναλαμβάνεται για να διασυνδεθεί το κενό διάστημα με την μέθοδο *barrel.* Η εικόνα 5-1-8 δείχνει τον συντάκτη γεγονότων και με τα δυο γεγονότα ολοκληρωμένα.

ΕΙΚΟΝΑ 5-1-8. Τελικές διασυνδέσεις

Τεστ

Τώρα ο κόσμος πρέπει να ελεγχθεί. Για να ελέγξετε τον εξομοιωτή πτήσης, απλά αποθηκεύστε τον κόσμο και πατήστε το κουμπί Play. Τίποτα δεν γίνεται μέχρι να πατήσουμε το επάνω βέλος, το οποίο προκαλεί το αεροσκάφος να καλέσει την μέθοδο *flyForward.*

Τα γεγονότα και οι μέθοδοι μπορούν να δημιουργηθούν για τα αριστερά και δεξιά βέλη και μπορούν να δημιουργηθούν και άλλα ακροβατικά. Παρόλ' αυτά είναι αναγκαίο να ελέγξετε τις μεθόδους όσο δημιουργούνται. Γράψτε μια μέθοδο και ελέγξτε την, γράψτε μια μέθοδο και ελέγξτε την, μέχρι να ολοκληρωθεί το πρόγραμμα. Αυτή είναι μια προγραμματιστική τεχνική που συνιστάται και ονομάζεται σταδιακή ανάπτυξη (incremental development). Το πλεονέκτημα της είναι ότι ο κώδικας με αυτό τον τρόπο διορθώνεται πιο εύκολα. Όταν κάτι δεν λειτουργεί, μπορεί να διορθωθεί πριν να δημιουργήσει προβλήματα κάπου αλλού.

Σημείωση: Ένας κόσμος με αλληλεπίδραση, όπως ένας εξομοιωτής πτήσης απαιτεί ο χρήστης να γνωρίζει ποια πλήκτρα να πατήσει για να λειτουργήσει σωστά. Μια αρχική μέθοδος πρέπει να χρησιμοποιηθεί στο *World.myfisrt method* ώστε να εμφανίζει ένα κείμενο 3D ή ένα πίνακα γρήγορης εξήγησης του χειρισμού του αεροσκάφους. Μετά από μερικά δευτερόλεπτα, το 3D κείμενο θα εξαφανιστεί (θέτοντας την *isShowing* ιδιότητα του *false*), και έτσι

μπορεί να ξεκινήσει η εξομοίωση. Τα 3D κείμενα και οι πίνακες ανάρτησης περιγράφονται στο “Μυστικά και Τεχνικές 2”.

Μυστικά και Τεχνικές 5

Γεγονότα

Μια γρήγορη αναφορά στα γεγονότα

ΕΙΚΟΝΑ Τ-5-1. Πιθανά γεγονότα στο Alice

- *When the world starts.* Αυτό το γεγονός γίνεται μια φορά, όταν πατηθεί το κουμπί Play και ξεκινήσει ο κόσμος του Alice.
- *When a key is typed.* Κάθε φορά που ο χρήστης πατάει ένα πλήκτρο τότε καλείται η μέθοδος.
- *When the mouse is clicked on something.* Κάθε φορά που κάνουμε κλικ με το ποντίκι πάνω σε ένα αντικείμενο, καλείται μια μέθοδος για να κατευθύνει το γεγονός.
- *While something is true.*Όσο μια υπόθεση είναι αληθής, εκτελείται η ενέργεια της μεθόδου.
- *When a variable changes.* Κάθε φορά που η τιμή μιας ιδιότητας ενός αντικειμένου αλλάζει καλείται η μέθοδος.
- *Let the mouse move objects.* Αυτό το γεγονός καλεί αυτόματα μια μέθοδο της Alice η οποία κινεί το αντικείμενο.
- *Let the arrow keys move <subject>.* Αυτό επιτρέπει στον χρήστη να κινήσει ένα συγκεκριμένο αντικείμενο πατώντας τα πλήκτρα βέλη. Το επάνω βέλος κινεί το αντικείμενο μπροστά, το κάτω κινεί πίσω και το αριστερά και δεξιά το κινεί αριστερά και δεξιά αντίστοιχα.
- *Let the mouse move the camera.* Αυτό επιτρέπει στον χρήστη να διευθύνει την κάμερα με το ποντίκι. Σημειώστε ότι είναι πιθανό να θέσετε την κάμερα μακριά από την σκηνή. Εάν συμβεί αυτό, δεν θα παρακολουθείτε τι συμβαίνει.
- *Let the mouse orient the camera.* Σαν το προηγούμενο γεγονός, αυτό πρέπει να χρησιμοποιηθεί με προσοχή, γιατί μπορείτε εύκολα να στοχεύσετε μακριά από την σκηνή και να χάσετε όλη την κίνηση.

Ασκήσεις

5-1 Ασκήσεις

1.Συμπλήρωμα του εξομοιωτή πτήσης

(α) Δημιουργήστε τον κόσμο για το σόου όπως περιγράφτηκε σε αυτή την ενότητα. Υλοποιήστε τις μεθόδους *flyForward* και *barrel* και διασυνδέστε τις με τα γεγονότα. Κάντε τις ενέργειες *move* και *roll* να έχουν ένα απότομο στυλ για να μειώσετε την παύση στην κίνηση μεταξύ των πατημάτων των πλήκτρων. Εάν ο υπολογιστής σας έχει ήχο, χρησιμοποιήστε τον ήχο της μηχανής του αεροσκάφους για να κάνετε πιο ρεαλιστικό το παράδειγμα.

(β) Όταν κάνετε τις *flyForward* και *barrel* να λειτουργήσουν, προσθέστε την *flyLeft* και *flyRight* για να οδηγήσετε το αεροπλάνο αριστερά και δεξιά.

(γ) Προσθέστε ένα ακροβατικό *forwardLoop* το οποίο λειτουργεί όταν πατηθεί το enter.

2.Τηλεχειριστήριο του ρομπότ

Ο κόσμος αυτής της άσκησης είναι ίδιος με τον κόσμο του ρομπότ First Encounter στα κεφάλαια 2 μέχρι 4. Σε αυτό τον κόσμο, θέλουμε να επιτρέψουμε στον χρήστη να χρησιμοποιήσει ένα είδος τηλεχειριστηρίου για να κινήσει το ρομπότ. Μια πιθανότητα είναι να χρησιμοποιήσουμε τον διακόπτη TwoButton (controls) για να εξομοιώσουμε ένα σύστημα ελέγχου. Όταν ο χρήστης κάνει κλικ στο πράσινο κουμπί το ρομπότ θα κινείται μπροστά, με δυο από τα πόδια του να περπατάνε. Όταν ο χρήστης πατάει το κόκκινο κουμπί το ρομπότ θα κινείται πίσω με δύο διαφορετικά πόδια να περπατάνε στην αντίθετη κατεύθυνση. Χρησιμοποιήστε την μέθοδο επιπέδου κόσμου *ask the user for a number* για να επιτρέψετε στον χρήστη να καθορίσει πόσα μέτρα θα διανύσει το ρομπότ.

ΕΙΚΟΝΑ

3.Βοηθός δακτυλογράφου

Το να μάθετε να δακτυλογραφείτε γρήγορα (χωρίς να κοιτάτε το πληκτρολόγιο) είναι μια ικανότητα που απαιτεί πολύ εξάσκηση. Σε αυτή την άσκηση, θα δημιουργήσετε ένα βοηθό δακτυλογράφου που ενθαρρύνει τον αρχάριο χρήστη να πληκτρολογήσει ένα σύνολο χαρακτήρων. Χρησιμοποιήστε γράμματα 3D για να δημιουργήσετε μια λέξη στον κόσμο (για παράδειγμα μπορείτε να γράψετε την λέξη ALICE με τα γράμματα A,L,I,C,E) και δημιουργήστε μια μέθοδο για κάθε γράμμα η οποία στρίβει το γράμμα δυο φορές. Όταν ο χρήστης πληκτρολογήσει ένα γράμμα στο πληκτρολόγιο το οποίο ταιριάζει με τα γράμματα στην οθόνη, το συγκεκριμένο γράμμα πρέπει να εκτελέσει την μέθοδο στροβιλισμού. Επίσης συμπεριλάβετε μια μέθοδο, *spinWord,* η οποία στροβιλίζει όλα τα γράμματα της λέξης όταν ο χρήστης πατάει το κενό διάστημα.

Υπόδειξη: Χρησιμοποιήστε την *asSeenBy* για να στροβιλίσετε τη λέξη.

ΕΙΚΟΝΑ

4.Περιστροφική κίνηση

Ένα δημοφιλές θέμα στην φυσική είναι η περιστροφική κίνηση. Δημιουργήστε ένα κόσμο, με τουλάχιστον τέσσερα αντικείμενα (πυξίδα, γραμματοκιβώτιο, λάστιχο). Δημιουργήστε μια μέθοδο για ρεαλιστική περιστροφική κίνηση για κάθε ένα από τα αντικείμενα, η οποία προκαλεί τα αντικείμενα να κάνουν μια πλήρη περιστροφή και μετά να κάνουν μια άλλη ενέργεια. Για παράδειγμα, εάν ένα από τα αντικείμενα είναι μια πυξίδα, κάντε την βελόνα της πυξίδας να γυρίσει γρήγορα προς μια κατεύθυνση και μετά προς την αντίθετη. Προσθέστε γεγονότα τα οποία θα καλούν την μέθοδο της περιστροφικής κίνησης για ένα αντικείμενο όταν κάνουμε κλικ πάνω του. (Εάν ο κόσμος έχει τέσσερα αντικείμενα, θα δημιουργηθούν τέσσερα γεγονότα).

ΕΙΚΟΝΑ

5.Κίνηση Ninja

Ένας ninja προσπαθεί να κάνει μια κίνηση καράτε και χρειάζεται πρακτική. Δημιουργήστε ένα κόσμο με ένα αντικείμενο ninja. Οι κινήσεις για τις οποίες πρέπει να εξασκηθεί ο ninja είναι: άλμα, σκύψιμο, χτύπημα, κλωτσιά. Εάν δεν έχετε κάνει ήδη (δείτε άσκηση 12 στο κεφάλαιο 4), γράψτε μεθόδους για τον ninja οι οποίες να συμπεριλαμβάνουν:

kickRight, kickLeft: επιτρέπουν στον ninja να κλωτσήσει με το δεξί και το αριστερό πόδι αντίστοιχα, περιλαμβάνοντας όλες τις κατάλληλες διαδικασίες.

rightJab, leftJab: επιτρέπει στον ninja να χτυπήσει μια γροθιά με το δεξί, αριστερό χέρι.

Δημιουργήστε γεγονότα και μεθόδους που επιτρέπουν στον χρήστη να χειριστεί τον ninja.

6.Γάτα Cheshire

Χρησιμοποιήστε την γάτα Cheshire από τα βιβλία “Alice in Wonderland”. Μερικές φορές, η γάτα θα εξαφανίζεται αφήνοντας μόνο το χαμόγελο της. Άλλες φορές θα εμφανίζεται. Δημιουργήστε ένα τέτοιο κόσμο, όπου η γάτα (εκτός από το χαμόγελο) εξαφανίζεται όταν πατηθεί το κόκκινο κουμπί του διακόπτη και μετά εμφανίζεται όταν πατηθεί το πράσινο. (Το δέντρο βρίσκεται στο φάκελο Nature).

ΕΙΚΟΝΑ

7.Έλεγχος κίνησης της χελώνας

Σε αυτή την άσκηση, θα δημιουργήσετε έναν ελεγκτή κίνησης της χελώνας (Animals) ώστε να βοηθήσετε τη χελώνα να εκτελέσει ασκήσεις για τον

ερχόμενο αγώνα με τον λαγό. Δημιουργήστε ένα κόσμο που να περιέχει μόνο την χελώνα και μετά δημιουργήστε τις μεθόδους για την χελώνα:

headBob: ελαφρύ κούνημα του κεφαλιού της χελώνας

tailWag: κούνημα της ουράς της χελώνας

oneStep: η χελώνα κινείται ένα βήμα μπροστά. Τα πόδια της πρέπει να κινηθούν όσο κάνει το ένα βήμα

walkForward: συνδυάζει τις παραπάνω τρεις μεθόδους, για να κάνει ένα ρεαλιστικό βήμα. Όλες οι κινήσεις πρέπει να κατέχουν το ίδιο ποσό χρόνου και να γίνονται ταυτόχρονα

turnAround: η χελώνα γυρίζει 180 μοίρες. Πρέπει να περπατάει όσο θα γυρίζει

turnLeft, turnRight: η χελώνα γυρίζει αριστερά, δεξιά και περπατάει όσο γυρίζει

Δημιουργήστε έλεγχο μέσω πληκτρολογίου:

Όταν πατηθεί το πλήκτρο του επάνω βέλους, η χελώνα πρέπει να κινηθεί μπροστά.

Όταν πατηθεί το πλήκτρο του κάτω βέλους, η χελώνα πρέπει να γυρίσει.

Όταν πατηθεί το πλήκτρο του αριστερά βέλους, η χελώνα πρέπει να γυρίσει αριστερά.

Όταν πατηθεί το πλήκτρο του δεξιά βέλους, η χελώνα πρέπει να γυρίσει δεξιά.

Τεστάρετε το σύστημα ελέγχου κίνησης της χελώνας, τρέχοντας τον κόσμο σας και δοκιμάζοντας όλες τις αλληλεπιδράσεις τουλάχιστον μια φορά.

Περίληψη

Αυτό το κεφάλαιο εστίασε στην δημιουργία κόσμων με αλληλεπίδραση. Δημιουργώντας κόσμους με γεγονότα σας επιτρέπουν να δομήσετε πιο ενδιαφέροντες κόσμους όπως παιχνίδια και εξομοιωτές. Σε πολλές αντικειμενοστραφείς γλώσσες προγραμματισμού, ο προγραμματισμός που οδηγείται από γεγονότα απαιτεί γνώση εξειδικευμένων θεμάτων. Ο συντάκτης γεγονότων σας επιτρέπει να δημιουργήσετε γεγονότα και να τα διασυνδέσετε με μεθόδους που χρησιμοποιούνται για χειρισμό γεγονότων. Η μέθοδος χειρισμού γεγονότων έχει την ευθύνη να ανταποκριθεί και να κάνει μια ενέργεια κάθε φορά που γίνεται ένα γεγονός. Ο συντάκτης γεγονότων διαχειρίζεται πολλές από τις μπερδεμένες λεπτομέρειες του προγραμματισμού που οδηγείται από γεγονότα.

Σημαντικά θέματα σε αυτό το κεφάλαιο

- Ένα γεγονός είναι κάτι που γίνεται.
- Ένα γεγονός δημιουργείται από μια είσοδο ενός χρήστη (πάτημα πλήκτρου, κλικ ποντικιού, κίνηση joystick).
- Ένα γεγονός συνδέεται με μια μέθοδο χειρισμού γεγονότων.
- Κάθε φορά που γίνεται ένα γεγονός, καλείται η αντίστοιχη μέθοδος χειρισμού γεγονότων. Αυτό σημαίνει προγραμματισμός καθοδηγούμενος από γεγονότα.
- Οι μέθοδοι χειρισμού γεγονότων περιέχουν πληροφορίες για την εκτέλεση μιας απάντησης στο γεγονός.

Παραρτήματα

Παράρτημα Α Χρήση του Alice

Ο σκοπός αυτών των ασκήσεων σε στυλ παραρτήματος είναι να σας βοηθήσει να μάθετε τα βασικά του προγράμματος Alice. Σας προτείνουμε να δουλέψετε αυτές τις ασκήσεις με ένα φίλο. Θα περάσετε καλά μαζί, και θα είστε σε θέση να βοηθήσετε ο ένας τον άλλο σε οποιοδήποτε σημείο που θα συναντήσετε δυσκολίες. Εάν κολλήσετε οπουδήποτε, πηγαίνετε στην αρχή της ενότητας, ξαναφορτώστε τον κόσμο και δοκιμάστε ξανά. Θα χάσετε μόνο λίγα λεπτά εργασίας.

Μέρος 1: Εκτέλεση οπτικού κόσμου στο Alice

Σε αυτό το μέρος, θα δουλέψετε με δύο κόσμους (FirstWorld και DancingBee- βρίσκονται στο cd που συνοδεύει το βιβλίο). Θα δημιουργήσετε και θα αποθηκεύσετε τον δικό σας κόσμο.

Όταν δείτε κείμενο γραμμένο σε αυτό το χρώμα, συγκεκριμένες πληροφορίες θα σας δοθούν για το τι να κάνετε.

Πώς να ξεκινήσετε την Alice: Η Alice μπορεί να ξεκινήσει με τους εξής δύο τρόπους:

1. Κάντε κλικ στο εικονίδιο στην επιφάνεια εργασίας. (Εικόνα Α-1-1)
2. Χρησιμοποιήστε την ιδιότητα των windows αναζήτηση για να βρείτε το “Alice.exe”

Το Alice θα κάνει ένα ή δυο λεπτά να φορτώσει.

Σε μερικές εγκαταστάσεις, το αρχικό μενού του Alice μπορεί να έχει την μορφή της εικόνας Α-1-2. Εάν αποφασίσετε ότι δεν θέλετε να βλέπετε αυτό το μενού κάθε φορά που ξεκινάτε το πρόγραμμα, ξετσεκάρετε το κουτάκι στην κάτω αριστερή γωνία που έχει τον τίτλο “Show this dialog at start”. (Σημείωση: Μερικοί υπολογιστές πανεπιστημίων επαναφέρουν τον υπολογιστή στην αρχική του μορφή μετά από κάθε επανεκκίνηση. Σας συμβουλεύουμε να μην ξετσεκάρετε το κουτάκι αν ο υπολογιστής σας βρίσκεται σε δίκτυο).

Κάντε κλικ στο φύλλο εργασίας Template και επιλέξτε τον κόσμο “grass” για αρχική σκηνή και μετά πατήστε άνοιγμα. Το Alice ξεκινά με ένα κενό κόσμο. Στο παράθυρο που φαίνεται ο κόσμος, πρέπει να δείτε το πράσινο γρασίδι και τον μπλε ουρανό όπως στην εικόνα Α-1-3. Εάν δεν βλέπετε κάτι τέτοιο πηγαίνετε στις πληροφορίες στο www.alice.org για το πώς να αντιμετωπίσετε προβλήματα εγκατάστασης.

Κόσμος 1: Ένα πρώτο πρόγραμμα με κίνηση

Ας ξεκινήσουμε ανοίγοντας ένα κόσμο.

Επιλέξτε το μενού File-Open World όπως φαίνεται στην εικόνα Α-1-4.

Το Alice ανοίγει ένα πλαίσιο διαλόγου για το άνοιγμα ενός κόσμου. Επιλέξτε το φύλλο Textbook για να δείτε παραδείγματα του βιβλίου, όπως φαίνεται στην εικόνα Α-1-5.

ΕΙΚΟΝΑ Α-1-1. Συντόμευση του Alice στην επιφάνεια εργασίας

ΕΙΚΟΝΑ Α-1-2. Αρχικό πλαίσιο διαλόγου του Alice

ΕΙΚΟΝΑ Α-1-3. Το interface του Alice

ΕΙΚΟΝΑ Α-1-4. Μενού File open world

ΕΙΚΟΝΑ Α-1-5. Άνοιγμα του παραδείγματος AppendixA_FirstWorld από τα παραδείγματα του βιβλίου.

Επιλέξτε το AppendixA_FirstWorld και κάντε κλικ στο κουμπί Open. Τα αντικείμενα μέλισσα και λαγός θα εμφανιστούν στον κόσμο, όπως φαίνεται στην εικόνα Α-1-6. Τα αντικείμενα βρίσκονται στο δέντρο αντικειμένων.

Πατήστε Play για να τρέξετε τον κόσμο.

ΕΙΚΟΝΑ Α-1-6. Το κουμπί Play στην κορυφή, το δέντρο αντικειμένων αριστερά και η αρχική σκηνή του κόσμου δεξιά.

Κοιτάξτε προσεκτικά τα κουμπιά ελέγχου στο επάνω μέρος του παραθύρου (Α-1-7) που εμφανίζεται όταν πατήσουμε το Play. Κάποια κουμπιά όπως το Pause (παύση), Restart (επανεκκίνηση) είναι προφανή. Το κουμπί Take picture θα πάρει μια φωτογραφία του έργου και θα την αποθηκεύσει ως εικόνα γραφικών. (Ένα popup μενού σας επιτρέπει να επιλέξετε που θα σώσετε το αρχείο). Το κουμπί Speed σας επιτρέπει να καθορίσετε την ταχύτητα του έργου.

ΕΙΚΟΝΑ Α-1-7. Το παράθυρο animation

Φυσικά η ταχύτητα των κινουμένων σχεδίων επηρεάζεται από τις δυνατότητες του υπολογιστή σας. Μια βοηθητική υπόδειξη είναι ότι το μέγεθος του παραθύρου στο οποίο εκτελείται το πρόγραμμα (το παράθυρο εκτέλεσης του κόσμου) επηρεάζει την ταχύτητα της κίνησης. Εάν το πρόγραμμα σας εκτελείται πολύ αργά, κάντε το παράθυρο πιο μικρό. (Επιλέξτε την κάτω δεξιά γωνία του παραθύρου και φέρτε το παράθυρο στο μέγεθος που θέλετε). Αφού αλλάξει το μέγεθος του παραθύρου, το Alice θα το θυμάται για την επόμενη φορά που θα εκτελέσετε πρόγραμμα.

Κλείστε το παράθυρο εκτέλεσης του προγράμματος για τον FirstWorld (κάντε κλικ στο X στην πάνω δεξιά γωνία του παραθύρου ή πατήστε το stop).

Κόσμος 2

Ο κόσμος 1 είναι ένα πρόγραμμα σε στυλ ταινίας. Μια ταινία εκτελείται από την αρχή μέχρι το τέλος όσο εσείς, σαν απλός χρήστης μόνο παρακολουθείτε. Ας κοιτάξουμε και σε ένα κόσμο που έχει αλληλεπίδραση και στον οποίο μπορείτε να κάνετε επιλογές όσο εκτελείται το πρόγραμμα.

Χρησιμοποιήστε File→ open world για να ανοίξει το πλαίσιο διαλόγου

Στο πλαίσιο διαλόγου επιλέξτε Textbook και AppendixA_Dancing Bee, όπως φαίνεται στην εικόνα A-1-8.

Πατήστε Play

Προσπαθήστε κάθε μια από τις δυο επιλογές (τα πλήκτρα μπορούν να πατηθούν με οποιαδήποτε σειρά)

1. Πατήστε το πλήκτρο του επάνω βέλους
2. Πατήστε το κενό διάστημα

Όταν τελειώσει το πρόγραμμα κλείστε το παράθυρο.

Αυτός ο κόσμος χρησιμοποιεί την ίδια αρχική σκηνή όπως στον πρώτο κόσμο, που είδαμε παραπάνω, άλλα το πρόγραμμα έχει αλλάξει ώστε να έχει αλληλεπίδραση και να είναι καθοδηγούμενο από γεγονότα. Πατώντας το πλήκτρο του επάνω βέλους δημιουργείται ένα γεγονός. Το Alice απαντάει στο γεγονός του επάνω βέλους κάνοντας τον λαγό να πηδήξει. Πατώντας το κενό διάστημα δημιουργείται ένα άλλο γεγονός. Το Alice ανταποκρίνεται και σε αυτό το γεγονός κάνοντας την μέλισσα να εκτελέσει μια πιρουέτα στον αέρα. Αυτός ο κόσμος είναι ένα παράδειγμα ενός προγράμματος με αλληλεπίδραση, καθοδηγούμενο από γεγονότα.

Σύνοψη

Ανακεφαλαίωση των όσων παρουσιάσαμε στους κόσμους 1 και 2. Εάν δεν αισθάνεστε ότι κατέχετε κάποιο από αυτά τα θέματα, πηγαίνετε πίσω στην αρχή αυτής της ενότητας και διαβάστε την ξανά.

ΕΙΚΟΝΑ A-1-8. Επιλέξτε τον κόσμο Appendix_A Dancing Bee

- Πώς να ξεκινήσετε την Alice
- Πώς να ανοίξετε ένα αποθηκευμένο κόσμο
- Πώς να εκτελέσετε ένα κόσμο
- Πώς να σταματήσετε ένα κόσμο
- Πώς να τρέξετε κόσμους με αλληλεπίδραση, με γεγονότα και αποκρίσεις

Κόσμος 3: Δημιουργία και αποθήκευση του δικού σας νέου κόσμου

ΕΙΚΟΝΑ A-1-9. Επιλέξτε New World από το μενού File

Κάντε κλικ στο μενού File, επάνω αριστερά στο Alice. Επιλέξτε “New World” όπως φαίνεται στην εικόνα A-1-9. Μετά επιλέξτε την φόρμα χιονιού για την αρχική σκηνή.

Αποθήκευση ενός κόσμου

Κάθε φορά που δημιουργείται ένας νέος κόσμος, πρέπει να αποθηκευτεί. Μετά, αν ο υπολογιστής σβήσει ξαφνικά (μπορεί να συμβεί ακόμη και στα καλύτερα μηχανήματα), η δουλεία σας θα είναι ασφαλής και θα μπορέσει να επαναφορτωθεί όταν ανοίξει ξανά ο υπολογιστής. Ένας κόσμος μπορεί να αποθηκευτεί σε διάφορες τοποθεσίες (επιφάνεια εργασίας, δίσκο). Εάν ένας δίσκος χρησιμοποιείται προτείνουμε μια μνήμη αποθήκευσης, παρά μια δισκέτα (το μέγεθος του κόσμου μπορεί να μην χωράει στην δισκέτα). Το παράδειγμα παρακάτω δείχνει καταλόγους αρχείων σε ένα δίσκο. Και οι άλλες συσκευές λειτουργούν ακριβώς το ίδιο.

Πηγαίνετε File→Save World As

Αυτό εμφανίζει ένα πλαίσιο κειμένου για αποθήκευση το οποίο σας επιτρέπει να πλοηγηθείτε στην τοποθεσία που θέλετε να αποθηκεύσετε τον κόσμο. Εικόνα A-1-10.

ΕΙΚΟΝΑ A-1-10. Πλαίσιο κειμένου Save World

Πηγαίνετε στον φάκελο όπου θέλετε να αποθηκεύσετε τον κόσμο, όπως φαίνεται στην εικόνα A-1-11. Προτείνουμε να δημιουργήσετε ένα φάκελο με το όνομα Alice Worlds όπου θα αποθηκεύετε όλα σας τα προγράμματα.

ΕΙΚΟΝΑ A-1-11. Επιλέξτε ένα φάκελο όπου θέλετε να αποθηκεύσετε τον κόσμο

Δώστε ένα όνομα στον κόσμο σας-συνιστούμε μια λέξη, όπως SnowmanExercise, που χρησιμοποιεί μικρά και κεφαλαία. Εισάγετε το όνομα του κόσμου σας και μετά κάντε κλικ στο πλήκτρο Save, όπως στην εικόνα A-1-12.

ΕΙΚΟΝΑ A-1-12. Εισάγετε ένα όνομα και κάντε κλικ στο save

Ο κόσμος σας θα αποθηκευτεί με μια κατάληξη .a2w (Alice version 2 world).

Όσο δουλεύετε τον κόσμο σας το Alice θα σας προτείνει ανά διαστήματα να κάνετε αποθήκευση. Σας συνιστούμε να αποθηκεύετε κάθε μισή ώρα. Η Alice κάνει αυτόματα back-up αρχεία του κόσμου σας όταν αποθηκεύετε. Ο φάκελος ονομάζεται “Backups of….”

Προσθέτοντας αντικείμενα στον κόσμο

Κάντε κλικ στο Add Objects στην κάτω δεξιά γωνία του παραθύρου, όπως φαίνεται στην εικόνα A-1-13.

ΕΙΚΟΝΑ A-1-13. Κουμπί Add Objects

Το Alice ανοίγει τον διαχειριστή σκηνών. Ένας οπτικός κατάλογος αρχείων στην τοπική και στην web βιβλιοθήκη παρέχεται για 3D μοντέλα. Ένα κουμπί αναζήτησης στην βιβλιοθήκη σας επιτρέπει να ψάξετε για ένα συγκεκριμένο είδος αντικειμένου. (Δείτε το παράρτημα Β για περισσότερες λεπτομέρειες σχετικά με την αναζήτηση ενός αντικειμένου).

Σημαντική σημείωση: Η τοπική βιβλιοθήκη είναι ένα δείγμα, περιέχει μια ποικιλία 3D μοντέλων. Το cd και η web βιβλιοθήκη περιέχουν χιλιάδες μοντέλα. Δείτε την επόμενη ενότητα αυτού του εγχειριδίου χρήσης για πληροφορίες σχετικά με το πώς θα χρησιμοποιήσετε την web βιβλιοθήκη. Η βιβλιοθήκη του cd θα εμφανιστεί μόνο εάν το cd της Alice θα είναι στον υπολογιστή σας. Τα παραδείγματα, οι ασκήσεις και οι εργασίες σε αυτό το κείμενο χρησιμοποιούν 3D μοντέλα και από τις δυο βιβλιοθήκες (τοπική και web). Εάν ψάχνετε για ένα συγκεκριμένο μοντέλο στην τοπική βιβλιοθήκη και δεν μπορείτε να το βρείτε, δοκιμάστε στην web βιβλιοθήκη στο www.alice.org

Κάντε κλικ στην τοπική βιβλιοθήκη, όπως φαίνεται στην εικόνα Α-1-14.

ΕΙΚΟΝΑ Α-1-14. Φάκελος τοπικής βιβλιοθήκης

Σημείωση: Η βιβλιοθήκη οργανώνεται σε συλλογές-για παράδειγμα Animals, Buildings και People.

Κάντε κλικ στην εικόνα των people, όπως φαίνεται στην Α-1-15.

ΕΙΚΟΝΑ Α-1-15. Η συλλογή People από 3D μοντέλα

Κάντε κλικ στο Snowman.

Κάντε κλικ στο “Add instance to your World”. Μπορείτε επίσης να προσθέσετε ένα αντικείμενο στον κόσμο χρησιμοποιώντας το ποντίκι για να σύρετε και να ρίξετε το αντικείμενο μέσα στον κόσμο, όπως στην εικόνα Α-1-16.

ΕΙΚΟΝΑ Α-1-16. Δύο ενδεχόμενα: Σύρετε και ρίξτε ή κάντε κλικ για να προσθέσετε ένα αντικείμενο (Instance)

Σημείωση: Το σύρσιμο και ρίξιμο δουλεύει σε σχέση με το έδαφος. Εάν το έδαφος διαγραφεί από τον κόσμο, αυτή η λειτουργία δεν δουλεύει.

Χρήση της web gallery (προαιρετικά)

Εάν ο υπολογιστής σας είναι συνδεδεμένος με το internet, μπορεί να θελήσετε να χρησιμοποιήσετε την web βιβλιοθήκη, όπως στην εικόνα Α-1-17. Η online βιβλιοθήκη παρέχει πολλά περισσότερα μοντέλα από αυτά που είναι διαθέσιμα στην τοπική βιβλιοθήκη.

ΕΙΚΟΝΑ Α-1-17. Ο φάκελος της web gallery

Σημείωση: Τα μοντέλα στην web βιβλιοθήκη μπορεί να χρειάζονται περισσότερο χρόνο από τα μοντέλα του cd για να φορτωθούν. Αυτό εξαρτάται από την ταχύτητα της σύνδεσης σας.

Ακολουθήστε την ίδια διαδικασία όπως παραπάνω για τον χιονάνθρωπο για να προσθέσετε ένα αντικείμενο από την online συλλογή.

Κουμπιά ελέγχου του ποντικιού στον συντάκτη σκηνών

Πιάστε και σύρτε τον χιονάνθρωπο με το ποντίκι. Μετά κάντε κλικ στο undo, όπως στην εικόνα Α-1-18.

ΕΙΚΟΝΑ Α-1-18. Το πλήκτρο undo

Έτσι διευθετείται ένα σοβαρό θέμα-δεν χρειάζεται να ανησυχείτε αν μπερδέψετε κάτι. Χρησιμοποιήστε το undo για να μεταβείτε σε μια προηγούμενη κατάσταση.

Στα δεξιά του συντάκτη σκηνών είναι μια γραμμή εργαλείων που σας επιτρέπει να ορίσετε τον τρόπο που θα κινεί το ποντίκι ένα αντικείμενο στο 3D χώρο. Εξ' ορισμού, είναι επιλεγμένη η οριζόντια κίνηση (αριστερά-δεξιά, μπρος-πίσω).

Επιλέξτε κάθε ένα από τα κουμπιά ελέγχου του πληκτρολογίου και πειραματιστείτε με τον χιονάνθρωπο, όπως στην εικόνα Α-1-19.

Αναφορικά, η εικόνα Α-1-20 σας παρουσιάζει τις ενέργειες του κάθε πλήκτρου. Το προεπιλεγμένο πλήκτρο, με τίτλο "οριζόντια", επιτρέπει το ποντίκι να κινήσει ένα αντικείμενο αριστερά, δεξιά, μπροστά και πίσω στην σκηνή.

ΕΙΚΟΝΑ Α-1-19. Γραμμή εργαλείων ελέγχου του ποντικιού

ΕΙΚΟΝΑ Α-1-20. Πλήκτρα της γραμμής εργαλείων ελέγχου του ποντικιού

Κινώντας υπομέρη ενός αντικειμένου

Τα πλήκτρα της γραμμής εργαλείων ελέγχου του ποντικιού είναι αυτόματα ορισμένα να κινούν ένα ολόκληρο αντικείμενο. Εάν τσεκάρετε το κουτάκι affect subparts, το ποντίκι μπορεί να χρησιμοποιηθεί για να ελέγχει την κίνηση των υπομερών.

Επιλέξτε το "affect subparts" και χρησιμοποιήστε τον κάθετο έλεγχο του ποντικιού για να κινήσετε το καπέλο του χιονάνθρωπου, όπως στην εικόνα Α-1-21.

ΕΙΚΟΝΑ Α-1-21. Το affect subparts είναι τσεκαρισμένο

Να προσέξετε να ξετσεκάρετε το “affect subparts” πριν συνεχίσετε!

Πειραματιστείτε με την αντιγραφή και διαγραφή

1. Χρησιμοποιήστε το πλήκτρο ελέγχου του ποντικιού αντιγραφή (εικόνα Α-1-20) για να δημιουργήσετε ένα δεύτερο χιονάνθρωπο. Μετά κάντε κλικ στον χιονάνθρωπο. Αυτό δημιουργεί ένα δεύτερο χιονάνθρωπο (στην ίδια τοποθεσία).
2. Στην πραγματικότητα, ένα αντίγραφο είναι κάτι σαν φάντασμα του αρχικού αντικειμένου-όχι εντελώς ανεξάρτητο. Στους περισσότερους κόσμους, προτιμάμε να προσθέσουμε νέα αντικείμενα. Διαγράψτε τον δεύτερο χιονάνθρωπο-δεξί κλικ πάνω στον χιονάνθρωπο και επιλέξτε Delete.
3. Στο τέλος, κάντε κλικ στο κουμπί ελέγχου του πληκτρολογίου για να σταματήσει η αντιγραφή.

Σημείωση: Εάν το έδαφος έχει διαγραφεί σε ένα κόσμο, νέα αντικείμενα μπορούν μόνο να προστεθούν χρησιμοποιώντας το πλήκτρο “Add instance”.

Προσθέστε ένα χιονάνθρωπο στον κόσμο.

Ο κόσμος θα έχει τώρα δύο αντικείμενα, ένα χιονάνθρωπο άντρα και μια γυναίκα, όπως στην εικόνα Α-1-22.

ΕΙΚΟΝΑ Α-1-22. Χιονάνθρωποι άντρας και γυναίκα

Χρησιμοποιώντας quad view (τετραγωνική άποψη)

Θέλουμε οι δυο χιονάνθρωποι να σταθούν απέναντι και να αντικρίζουν ο ένας τον άλλο. Ο διαχειριστής σκηνών μπορεί να χρησιμοποιηθεί ώστε να τακτοποιήσει τα δυο αντικείμενα.

Επιλέξτε “quad view” στον συντάκτη, όπως στην εικόνα Α-1-23.

ΕΙΚΟΝΑ Α-1-23. Πλευρές “quad view”

Το παράθυρο του κόσμου αλλάζει σε ένα παράθυρο με τέσσερις τετραγωνικές απόψεις. Οι τέσσερις απόψεις είναι: Camera, Top, Right και Front. Παρατηρήστε ότι το κουμπί οριζόντιου ελέγχου για το ποντίκι δεν είναι διαθέσιμο γιατί οι πλευρές Front και Right έχουν κάθετη κίνηση. Η γραμμή εργαλείων τώρα περιέχει και μια δεύτερη σειρά από κουμπιά, περιέχοντας ένα για ζουμ και ένα για ανέβασμα και κατέβασμα της εικόνας. Εικόνα Α-1-24.

ΕΙΚΟΝΑ Α-1-24. Μια δεύτερη γραμμή στην γραμμή εργαλείων

Στην τετραγωνική άποψη που φαίνεται παραπάνω, ο ένας χιονάνθρωπος βρίσκεται εκτός οπτικού πεδίου στην όψη από Επάνω. (Ο κόσμος σας μπορεί να είναι διαφορετικός από τον δικό μας). Το εργαλείο για ανέβασμα και κατέβασμα της οθόνης μπορεί να χρησιμοποιηθεί για να ανατοποθετηθεί η σκοπιά σε μια οπτική πλευρά. Όπως φαίνεται στην εικόνα Α-1-25, χρησιμοποιήσαμε το εργαλείο για ανέβασμα και κατέβασμα της οθόνης για να αλλάξουμε την σκοπιά στην Επάνω όψη.

ΕΙΚΟΝΑ Α-1-25. Χρήση του εργαλείου για ανέβασμα και κατέβασμα της οθόνης

Ένα άλλο χρήσιμο χαρακτηριστικό είναι το zoom (ζουμ). Το ζουμ μπορεί να χρησιμοποιηθεί για να φέρετε πιο κοντά ή πιο μακριά μια εικόνα. Όπως στην εικόνα Α-1-26, χρησιμοποιήσαμε το ζουμ για να απομακρυνθούμε από την σκοπιά της Επάνω όψης.

ΕΙΚΟΝΑ Α-1-26. Χρήση του εργαλείου για ζουμ

Χρησιμοποιήστε το ποντίκι για να τακτοποιήσετε τα δύο αντικείμενα έτσι ώστε να έρθουν πλευρά με πλευρά.

Μια θέση πλευρά με πλευρά μπορεί να αναγνωριστεί όταν ένα αντικείμενο κρύβει το άλλο στην όψη Right (δείτε το κυκλωμένη όψη Right στην εικόνα Α-1-27).

ΕΙΚΟΝΑ Α-1-27. Όψη Right

Χρησιμοποιήστε το ποντίκι για να τακτοποιήσετε τα δύο αντικείμενα έτσι ώστε να έρθουν πρόσωπο με πρόσωπο.

Η θέση πρόσωπο με πρόσωπο αναγνωρίζεται από τις όψεις Top και Front, όπως στην εικόνα Α-1-28.

ΕΙΚΟΝΑ Α-1-28. Top και Front όψεις

Κάντε κλικ στο “single view” για να γυρίσετε σε μια μόνο εικόνα στον διαχειριστή σκηνών, όπως στην εικόνα Α-1-29.

ΕΙΚΟΝΑ Α-1-29. Single view

Χρήση των πλήκτρων ctrl και shift για κίνηση με βάση το ποντίκι

Παρόλο που η γραμμή εργαλείων ελέγχου το ποντικιού προσφέρουν διάφορες επιλογές για την κίνηση των αντικειμένων, μερικές φορές είναι πιο βολικό να χρησιμοποιείτε το πληκτρολόγιο μαζί με το ποντίκι. Κρατώντας πατημένο το πλήκτρο ctrl σας επιτρέπει να χρησιμοποιήσετε το ποντίκι για να γυρίσετε ένα αντικείμενο. Κρατώντας πατημένο το πλήκτρο shift σας επιτρέπει να χρησιμοποιήσετε το ποντίκι για να κινήσετε το αντικείμενο επάνω, κάτω.

Κίνηση της κάμερας

Όταν στήνουμε μια σκηνή, μπορούμε να ρυθμίσουμε την σκοπιά της κάμερας έτσι ώστε να βλέπουμε από όποια μεριά θέλουμε την κίνηση. Μπορεί να βοηθήσει αν σκεφτούμε ότι η κάμερα είναι μια τηλεκατευθυνόμενη συσκευή αέρος η οποία περιίπταται στον αέρα πάνω από την σκηνή. Κινώντας την κάμερα, αλλάζουμε την εικόνα που έχουμε στον κόσμο.

Τα μπλε κουμπιά ελέγχου που έχουμε στο κάτω μέρος του διαχειριστή σκηνών είναι τα κουμπιά ελέγχου της κάμερας και φαίνονται στην εικόνα Α-1-30.

ΕΙΚΟΝΑ Α-1-30. Κουμπιά ελέγχου κάμερας

Κάντε κλικ στα κουμπιά ελέγχου της κάμερας για να δείτε τι κάνει το καθένα.

Μπορείτε πάντα να χρησιμοποιείτε το undo σε περίπτωση λάθους.

Τακτοποίηση πολλών αντικειμένων στην αρχική σκηνή

Όταν πρόκειται να μπουν πολλά αντικείμενα στην σκηνή, η τοποθέτηση αντικειμένων ένα την φορά και η τακτοποίηση τους στην σκηνή μπορεί να είναι καταστροφική. Ο λόγος είναι ότι όχι μόνο κινείτε αντικείμενα, αλλά κινείτε και την κάμερα γύρω από την σκηνή. Μετά από αρκετές τέτοιες κινήσεις, ένα νέο αντικείμενο μπορεί να μην φαίνεται στην κάμερα όταν τοποθετηθεί αρχικά. Αυτό σημαίνει ότι πρέπει να ξοδέψετε πολύ χρόνο με τα κουμπιά ελέγχου της κάμερας μέχρι να βρείτε το νέο αντικείμενο και να το τοποθετήσετε στην σκηνή.

Σας συνιστούμε όλα τα αντικείμενα να τοποθετούνται αμέσως μόλις δημιουργηθεί ο κόσμος. Μετά χρησιμοποιήστε τα κουμπιά ελέγχου της κάμερας για να τακτοποιήσετε τα αντικείμενα στην σκηνή. Αυτή η απλή διαδικασία εξοικονομεί πολύ χρόνο και κόπο. Η εικόνα Α-1-31(α) στην επόμενη σελίδα δείχνει ένα νέο κόσμο που έχουν προστεθεί πολλά αντικείμενα, όλα κοντά στο κέντρο του κόσμου. Η εικόνα Α-1-31(β) δείχνει τα αντικείμενα κατανεμημένα στην σκηνή στις σωστές θέσεις.

Σύνοψη

Ανακεφαλαίωση των όσων παρουσιάσαμε στο κόσμο 3. Εάν δεν αισθάνεστε ότι κατέχετε κάποιο από αυτά τα θέματα, πηγαίνετε πίσω στην αρχή αυτής της ενότητας και διαβάστε την ξανά.

ΕΙΚΟΝΑ Α-1-31. (α) όλα τα αντικείμενα κοντά στο κέντρο του κόσμου
(β) τα αντικείμενα τακτοποιημένα στην σκηνή

- Πώς να φτιάξετε ένα νέο κόσμο
- Αποθήκευση του κόσμου
- Πρόσθεση αντικειμένων στον κόσμο
- Χρήση της γραμμής εργαλείων ελέγχου του ποντικιού
- Χρήση της γραμμής εργαλείων ελέγχου του ποντικιού για τα υπομέρη
- Διαγραφή ενός αντικειμένου
- Χρήση του quad view
- Κίνηση της κάμερας
- Τακτοποίηση αντικειμένων στην σκηνή

Μέρος 2: Χρήση popup μενού για την δημιουργία αρχικής σκηνής

Στο μέρος 1, χρησιμοποιήσατε κουμπιά ελέγχου του ποντικιού για να τακτοποιήσετε τα αντικείμενα στην σκηνή. Σε αυτό το μέρος, θα μάθετε πώς να χρησιμοποιείτε μεθόδους για εργασία με τα αντικείμενα της αρχικής σκηνής.

Το Alice παρέχει ένα αριθμό από ενσωματωμένες εντολές που μπορούν να χρησιμοποιηθούν για τον ορισμό του μεγέθους και της θέσης του αντικειμένου. Αυτές οι εντολές καλούνται μέθοδοι. Για να υλοποιήσετε την χρήση τους, ας δουλέψουμε με ένα νέο κόσμο όπου θα χρησιμοποιηθούν μέθοδοι για το στήσιμο της σκηνής (μαζί με τα πλήκτρα ελέγχου του ποντικιού που μάθατε στο μέρος 1 αυτού του παραρτήματος).

Για να ξεκινήσετε αυτή την ενότητα, χρησιμοποιήστε αρχικά File→New World για να ξεκινήσετε ένα νέο κόσμο. Επιλέξτε ένα κόσμο με γρασίδι. Κάντε κλικ στο κουμπί Add Objects για να στήσετε την αρχική σκηνή, όπως στην εικόνα Α-2-1.

Προσθέστε ένα χαρούμενο δέντρο και ένα βάτραχο στον κόσμο.

Ο βάτραχος είναι στον φάκελο Animals της Local Gallery. Το χαρούμενο δέντρο είναι στον φάκελο Nature.

ΕΙΚΟΝΑ Α-2-1. Το κουμπί Add objects για πρόσθεση αντικειμένων

Όπως φαίνεται στην εικόνα Α-2-2, ο βάτραχος είναι μικρός και φαίνεται να χάνεται μέσα στο γρασίδι. Αυτή η κατάσταση είναι κατάλληλη για να

καμουφλαριστεί ο βάτραχος από τους εχθρούς του άλλα δεν είναι κατάλληλο για το πρόγραμμα. Μπορείτε να χρησιμοποιήσετε μια μέθοδο *resize* για να μεγεθύνετε τον βάτραχο.

ΕΙΚΟΝΑ Α-2-2. Ο βάτραχος είναι πολύ μικρός

Δεξί κλικ στο βάτραχο στο δέντρο αντικειμένων.

Επιλέξτε την μέθοδο "*frog resize*" και επιλέξτε δυο ως ποσό, όπως στην εικόνα Α-2-3.

Αυτό θα κάνει το βάτραχο δυο φορές πιο μεγάλο από το τρέχον μέγεθος του.

ΕΙΚΟΝΑ Α-2-3. Ο βάτραχος γίνεται δυο φορές μεγαλύτερος

Σημείωση: Η αλλαγή μεγέθους του αντικειμένου μπορεί να έχει απρόβλεπτα αποτελέσματα. Ένα αντικείμενο που στέκεται στο έδαφος μπορεί να βυθιστεί. Ανατοποθετήστε το αντικείμενο χρησιμοποιώντας το ποντίκι, όταν χρειάζεται.

Η μέθοδος *turn* κάνει ένα αντικείμενο να γυρίσει σε μια δεδομένη κατεύθυνση (μπροστά, πίσω, δεξιά, αριστερά) για συγκεκριμένες περιστροφές. Ένα αντικείμενο κινείται σε σχέση με τον δικό του προσανατολισμό.

Χρησιμοποιήστε μια μέθοδο για να γυρίσετε τον βάτραχο 1/4 περιστροφές δεξιά, όπως στην εικόνα Α-2-4.

Η μέθοδος *roll* γυρίζει ένα αντικείμενο δεξιά, αριστερά.

Χρησιμοποιήστε μια μέθοδο *roll* για να γυρίσετε το δέντρο 1/4 περιστροφές δεξιά όπως στην εικόνα Α-2-5.

Η πληροφορία *stand up* κάνει τον κάθετο άξονα ενός αντικειμένου να ευθυγραμμιστεί με τον κάθετο άξονα του κόσμου. Με άλλα λόγια το αντικείμενο στέκεται όρθιο!

Χρησιμοποιήστε μια πληροφορία *stand up* για να βάλετε το δέντρο κατακόρυφα στην σωστή θέση, όπως στην εικόνα Α-2-6.

ΕΙΚΟΝΑ Α-2-4. Γυρίστε τον βάτραχο 1/4 περιστροφές δεξιά

ΕΙΚΟΝΑ Α-2-5. Γυρίστε το δέντρο 1/4 περιστροφές δεξιά

ΕΙΚΟΝΑ Α-2-6. Σηκώστε όρθιο το δέντρο

Η μέθοδος *turn to face* κάνει ένα αντικείμενο να αντικρίζει ένα άλλο.

Χρησιμοποιήστε τη μέθοδο *turn to face* για να κάνετε το βάτραχο να αντικρίσει το δέντρο, όπως στην εικόνα Α-2-7.

ΕΙΚΟΝΑ Α-2-7. Ο βάτραχος αντικρίζει το δέντρο

Χρήση μεθόδων με τα υπομέρη των αντικειμένων

Τα υπομέρη ενός αντικειμένου μπορούν να διαχειριστούν με μια μέθοδο, κάνοντας κλικ στο υπομέρος στο δέντρο αντικειμένων και μετά δεξί κλικ στο υπομέρος για να πάρετε το popup μενού. Για να δείτε μια λίστα των υπομερών ενός αντικειμένου, κάντε πρώτα αριστερό κλικ στο τετράγωνο κουτί με τον σταυρό στα αριστερά του αντικειμένου στο δέντρο αντικειμένων. Η εικόνα Α-2-8 δείχνει τα υπομέρη του βατράχου στο δέντρο αντικειμένων. Ένα από τα μέρη του βατράχου είναι το σαγόνι του-και η γλώσσα είναι υπομέρος του σαγονιού. (Τα μέρη μπορεί να έχουν μέρη, τα οποία μπορεί να έχουν μέρη κοκ)

Επιλέξτε τη γλώσσα του βάτραχου και μια μέθοδο κίνησης εμπρός της γλώσσας του βατράχου. Ως ποσό επιλέξτε το “άλλο”, όπως στην εικόνα Α-2-9. Ένα πλαίσιο με αριθμούς εμφανίζεται και μπορείτε να επιλέξετε αριθμό. Επιλέξτε 0.05. (Αυτή είναι μια αυθαίρετη τιμή-μπορείτε να δοκιμάσετε και άλλες μέχρι να πάρετε το επιθυμητό αποτέλεσμα).

ΕΙΚΟΝΑ Α-2-8. Υπομέρη του βατράχου

Ο βάτραχος βγάζει την γλώσσα έξω όπως στην εικόνα Α-2-10.

ΕΙΚΟΝΑ Α-2-9. Κινήστε τη γλώσσα του βάτραχου μπροστά

ΕΙΚΟΝΑ Α-2-10. Η γλώσσα του βάτραχου πετιέται έξω

Κοιτάξτε καλά αυτό το αντικείμενο

Το popup μενού που εμφανίζεται με το δεξί κλικ σε ένα αντικείμενο, είναι ένα μενού με πολλά αντικείμενα. Σε αυτό το εγχειρίδιο, χρησιμοποιήσατε τις μεθόδους. Άλλα στοιχεία (*rename, capture pose, save object*) εξηγούνται στα παραδείγματα στο βιβλίο. Ένα στοιχείο στο μενού είναι διαφορετικό από τα άλλα. Τα στοιχεία *methods, rename, capture pose, save object* και *delete* είναι όλα σχεδιασμένα ώστε να ενεργούν στο αντικείμενο. Αλλά το “*Camera get a look at this*” είναι μια μέθοδος την κάμερας. Όταν επιλέξετε το “*Camera get a look at this*” από το μενού, η κάμερα κάνει ζουμ ώστε να κάνει ένα κοντινό πλάνο σε ένα αντικείμενο. Μπορείτε να χρησιμοποιήσετε το undo για να γυρίσετε στην προηγούμενη σκοπιά.

Κάντε χρήση του “*Camera get a look at this*” για να κάνετε ένα κοντινό στον βάτραχο, όπως στην εικόνα Α-2-11.

(Εάν ένα άλλο αντικείμενο βρίσκεται μεταξύ της κάμερας και του αντικειμένου, και μπλοκάρει την κάμερα, χρησιμοποιήστε το undo και μετακινήστε την κάμερα ή το αντικείμενο).

ΕΙΚΟΝΑ Α-2-11. *Camera get a look at this*

Μια σύγκριση των πλήκτρων ελέγχου της κίνησης

Στο μέρος 1 αυτού του εγχειριδίου, πειραματιστήκατε με πλήκτρα ελέγχου κίνησης του ποντικιού. Στο μέρος 2, δουλέψατε με μεθόδους. Και στις δύο περιπτώσεις, τοποθετούσατε τα αντικείμενα για να δημιουργήσετε μια αρχική σκηνή. Μπορεί να αναρωτιέστε, “Ποιο είδος ελέγχου κίνησης είναι καλύτερο;”

Πιστεύουμε ότι τα κουμπιά ελέγχου ποντικιού είναι κατάλληλα στην προσεγγιστική τοποθέτηση των αντικειμένων, αλλά οι μέθοδοι του popup μενού χρησιμοποιούνται για την ακριβή τοποθέτηση. Ο διαχειριστής σκηνών είναι κατάλληλος για την τοποθέτηση αντικειμένων το ένα σε σχέση με το άλλο. Είναι εύκολο να προσθέσετε ένα αντικείμενο σε ένα κόσμο και μετά να χρησιμοποιήσετε το ποντίκι για να κινήσετε και να περιστρέψετε προσεγγιστικά στην θέση που θέλετε. Εάν κάνουμε λάθος, μπορούμε να πατήσουμε το undo (ή και να διαγράψετε) και να ξαναπροσπαθήσετε. Ενώ η προσεγγιστική τοποθέτηση ενός αντικειμένου είναι εύκολο να γίνει με το ποντίκι, η ακριβή τοποθέτηση του είναι πιο δύσκολη. Για παράδειγμα, το να τοποθετήσετε ένα αντικείμενο πάνω στο άλλο είναι δύσκολο να γίνει με το ποντίκι. Το να το τοποθετήσετε σχεδόν το ένα πάνω στο άλλο δεν είναι τόσο δύσκολο. Οι μέθοδοι δίνουν καλή στοίχιση. Στο στήσιμο των σκηνών του κόσμου, η καλύτερη στρατηγική είναι να χρησιμοποιείτε ένα συνδυασμό μεθόδων και κουμπιά ελέγχου του ποντικιού.

Αναζήτηση της βιβλιοθήκης

Συχνά, θέλετε να βρείτε ένα αντικείμενο, π.χ. μια μπάλα, αλλά δεν ξέρετε που να βρείτε το 3D μοντέλο στην βιβλιοθήκη. Μπορείτε να ψάξετε την βιβλιοθήκη για ένα συγκεκριμένο είδος αντικειμένου. Κάντε κλικ στο κουμπί Search, όπως στην εικόνα Β-1-1.

Μπορείτε να ψάξετε είτε την local gallery, είτε την web στο www.alice.org (εάν ο υπολογιστής σας είναι συνδεδεμένος στο internet). Πάντα πρέπει πρώτα να ψάχνετε την τοπική σας βιβλιοθήκη, αφού αυτή η αναζήτηση είναι γρηγορότερη. Όταν εμφανιστεί το παράθυρο για την αναζήτηση, όπως στην εικόνα Β-1-2, εισάγετε το όνομα του μοντέλου (ή μέρος του ονόματος) το οποίο ψάχνετε, και μετά κάντε κλικ στο Search! δίπλα στο κείμενο που αναζητείται.

Εάν δεν βρείτε αυτό που ψάχνετε, ψάξτε στο www.alice.org όπως φαίνεται στην εικόνα Β-1-3. Την πρώτη φορά που θα ψάξετε στο www.alice.org, μπορεί να πάρει μερικά λεπτά (ανάλογα με την ταχύτητα σύνδεσης στο internet). Οι πρόσθετες αναζητήσεις θα είναι πιο γρήγορες.

Δημιουργία των δικών σας ανθρώπινων μοντέλων

Οι βιβλιοθήκες παρέχουν χιλιάδες 3D μοντέλα για την δημιουργία των δικών σας κόσμων. Το Alice δεν είναι ένα πρόγραμμα που δημιουργεί γραφικά μοντέλα, άλλα έχει δυο ιδιότητες δημιουργίας ειδικών μοντέλων ανθρώπων (hebuilder και shebuilder) στον φάκελο People στην local gallery, όπως φαίνεται στην εικόνα Β-1-4. Ένα κλικ στο hebuilder ή στο shebuilder θα εμφανίσει ένα παράθυρο δημιουργίας ενός ατόμου (Β-1-5), όπου μπορείτε να επιλέξετε το επιθυμητό τύπο σώματος, μαλλιών, δέρματος, χρώμα ματιών και ρούχων. Όταν τελειώσετε με τις επιλογές σας, πατήστε OK για να δώσετε όνομα και να προσθέσετε το αντικείμενο στον κόσμο. Το Alice αυτόματα ορίζει μια μέθοδο για περπάτημα walk. Αυτό είναι ένα πλεονέκτημα, επειδή αυτή η μέθοδος είναι δύσκολο να γραφεί.

ΕΙΚΟΝΑ Β-1-1. Αναζήτηση της βιβλιοθήκης

ΕΙΚΟΝΑ Β-1-2. Εισαγωγή ενός ονόματος για το αντικείμενο

ΕΙΚΟΝΑ Β-1-3. Αναζήτηση της web library στο www.alice.org

ΕΙΚΟΝΑ Β-1-4. Οι ιδιότητες μοντέλων hebuilder και shebuilder

ΕΙΚΟΝΑ Β-1-5. hebuilder

Αντιγραφή και επικόλληση: clipboard

Υποθέστε ότι θέλετε να αντιγράψετε και να επικολλήσετε μια συγκεκριμένη ακολουθία εντολών κίνησης. Εδώ είναι χρήσιμο ένα clipboard. Για να αντιγράψετε μια ακολουθία εντολών, κάντε κλικ στην αριστερή πλευρά (στις

μικρές τελείες) του μπλοκ των εντολών που θέλετε να αντιγράψετε. Με αυτό τον τρόπο επιλέγονται οι εντολές που θέλετε να αντιγράψετε. (Σημείωση: επιλέγοντας μια εντολή *Do in order* ή *Do together* επιλέγονται όλες οι εντολές μέσα σε αυτή). Μετά με το ποντίκι τοποθετήστε τις εντολές στο clipboard. Η εικόνα Β-1-6 εμφανίζει τον τρόπο τοποθέτησης των εντολών από τον συντάκτη στο clipboard.

ΕΙΚΟΝΑ Β-1-6. Τοποθέτηση εντολών στο clipboard

Εφόσον οι εντολές αντιγραφούν στο clipboard, το clipboard αλλάζει χρώμα (άσπρο). Η αλλαγή του χρώματος είναι ένα οπτικό στοιχείο που υποδηλώνει ότι στον πίνακα υπάρχουν αντιγραμμένες εντολές. Τώρα που ο πίνακας περιέχει εντολές, μπορείτε να χρησιμοποιήσετε το ποντίκι για να σύρετε τις επιλεγμένες εντολές από τον πίνακα στον συντάκτη για να δημιουργήσετε ένα νέο αντίγραφο αυτών των εντολών κάπου αλλού στο πρόγραμμα σας, όπως στην εικόνα Β-1-7. Μπορείτε τώρα να έχετε δυο αντίγραφα των εντολών στον συντάκτη. (Μπορείτε να διαγράψετε τις παλιές εντολές σέρνοντας τις στον κάδο ανακύκλωσης).

Προσέξτε ότι ένα clipboard μπορεί να κρατήσει μόνο ένα σύνολο εντολών τη φορά. Εάν είχαν αντιγραφεί προηγουμένως εντολές στο clipboard, η αντιγραφή ενός νέου συνόλου εντολών στον ίδιο πίνακα θα έχει ως αποτέλεσμα να σβηστούν οι παλιές εντολές. Στην αρχική εγκατάσταση η Alice εμφανίζει μόνο ένα clipboard. Ο αριθμός των clipboards μπορεί να αυξηθεί επιλέγοντας το μενού Edit και μετά Preferences. Στο φύλλο εργασίας Seldom Used του παραθύρου Preferences, αλλάξτε τον αριθμό των clipboards, όπως στην εικόνα Β-1-8.

Διαγραφή κώδικα

Τι γίνεται όταν κάνετε λάθος ή θέλετε να αλλάξετε τον κώδικα; Ο πιο εύκολος τρόπος για να διαγράψετε μια γραμμή κώδικα είναι να την σύρετε στον κάδο ανακύκλωσης στην κορυφή του παραθύρου του Alice. Ένα ολόκληρο μπλοκ κώδικα μπορεί να διαγραφεί κατά αυτόν τον τρόπο, όπως στην εικόνα Β-1-9.

Εάν θέλετε να απαλείψετε ένα μπλοκ *Do together* ή *Do in order* αλλά να κρατήσετε τις γραμμές του κώδικα, κάντε δεξί κλικ στο μπλοκ και επιλέξτε *dissolve,* όπως στην εικόνα Β-1-10. Οι γραμμές του κώδικα θα προωθηθούν ένα επίπεδο και το προηγούμενο μπλοκ θα διαγραφεί. Εικόνα Β-1-11.

ΕΙΚΟΝΑ Β-1-7. Τοποθέτηση εντολών από το clipboard στον συντάκτη

ΕΙΚΟΝΑ Β-1-8. Αύξηση του αριθμού των clipboards

ΕΙΚΟΝΑ Β-1-9. Σύρσιμο ενός μπλοκ κώδικα στο κάδο ανακύκλωσης για να διαγραφεί όλο το μπλοκ

ΕΙΚΟΝΑ Β-1-10. Επιλογή του *dissolve* για να διαγραφεί το μπλοκ *Do together*

ΕΙΚΟΝΑ Β-1-11. Μετά το *dissolve,* το μπλοκ *Do together* έχει διαγραφεί αλλά οι εντολές παραμένουν

Εκτύπωση: Εξαγωγή κώδικα προγράμματος σε αρχείο HTML

Μπορεί να θελήσετε να εκτυπώσετε τον κώδικα από μια ή περισσότερες μεθόδους του προγράμματος. Πρώτα βεβαιωθείτε ότι ο κόσμος είναι ανοιχτός στο Alice. Για να εκτυπώσετε μια μέθοδο, κάντε κλικ στο μενού File και επιλέξτε Export Code For Printing, όπως στην εικόνα Β-1-12. Στο πλαίσιο διαλόγου της εκτύπωσης, επιλέξτε όνομα της μεθόδου (ή των μεθόδων) που θέλετε να εκτυπωθούν, το φάκελο που είναι αποθηκευμένο το έγγραφο και το όνομά σας. Μετά κάντε κλικ στο κουμπί Export Code, όπως στην εικόνα Β-1-13. Ο κώδικας των επιλεγμένων μεθόδων θα εξαχθεί σε ένα HTML αρχείο, το οποίο μπορεί να εκτυπωθεί από τον browser σας.

ΕΙΚΟΝΑ Β-1-12. Επιλογή του *Export Code For Printing*

ΕΙΚΟΝΑ Β-1-13. Επιλογή μιας μεθόδου για εκτύπωση

Μορφή web: Εξαγωγή κόσμου

Η εξαγωγή ενός κόσμου για προβολή σε μια ιστοσελίδα είναι ένας υπέροχος τρόπος για να φιγουράρετε την δημιουργικότητα σας. Πρώτα σιγουρευτείτε ότι ο κόσμος είναι ανοιχτός στο Alice. Για να εξάγετε ένα κόσμο, κάντε κλικ στο μενού File και μετά επιλέξτε Export As A Web Page, όπως στην εικόνα Β-1-14.

Ένα πλαίσιο διαλόγου Save World for the Web σας επιτρέπει την εισαγωγή ενός τίτλου, πλάτους, ύψους του παραθύρου του κόσμου που θα εμφανίζεται στον browser, και την θέση όπου θα αποθηκευτούν τα web αρχεία, όπως στην εικόνα Β-1-15. (Η θέση στο πλαίσιο Save Location δεν μπορεί να είναι URL). Κάνοντας κλικ στο κουμπί Save, το Alice αποθηκεύει τρία αρχεία στον κατάλογο που επιλέξατε: το συνηθισμένο .a2W, ένα .html, και ένα Java αρχείο .jar.

Για να επιτρέπετε στους άλλους να βλέπουν τον κόσμο σας μέσω internet, πρέπει να αποθηκεύσετε τα τρία αυτά αρχεία σε ένα web server. Και τα τρία αρχεία πρέπει να βρίσκονται στον ίδιο κατάλογο. Την πρώτη φορά που κάποιος φορτώνει μια ιστοσελίδα, θα πάρει μερικά δευτερόλεπτα (εξαρτάται από την ταχύτητα σύνδεσης τους στο internet) εξαιτίας του μεγέθους του αρχείου Java. (Εάν ο υπολογιστής δεν έχει ένα browser εφοδιασμένο με Java ή Java 3D ή Java Media Frameworks, η ιστοσελίδα θα προτρέψει τον χρήστη να κατεβάσει ένα αρχείο από την ιστοσελίδα της Java, που υποστηρίζεται από την *Sun Microsystems*). Μετά από αυτά η ιστοσελίδα θα ανοίξει γρήγορα.

Εξαγωγή σε ταινία

Την περίοδο που εκδόθηκε αυτό το βιβλίο, αυτό το χαρακτηριστικό δεν είχε ακόμη υλοποιηθεί. Περιμένουμε ότι θα υλοποιηθεί μέχρι να διαβάσετε αυτό το βιβλίο. Ελέγξτε την ιστοσελίδα του Alice www.alice.org για διάφορες ενημερώσεις.

ΕΙΚΟΝΑ Β-1-14. Επιλογή του στοιχείου *Export As A Web Page*

ΕΙΚΟΝΑ Β-1-15. Οι πληροφορίες που χρειάζονται για να αποθηκευτεί ένας κόσμος και να εξαχθεί στο internet

ΚΕΦΑΛΑΙΟ 3
ΥΛΟΠΟΙΗΣΗ ΠΡΟΓΡΑΜΜΑΤΩΝ

Υλοποίηση Προγραμμάτων

Σε αυτό το κεφάλαιο παρατίθενται ορισμένα παραδείγματα που υλοποιούνται με το πρόγραμμα Alice. Ξεκινώντας, τα παραδείγματα είναι εύκολα στην δημιουργία και την κατανόηση τους και όσο προχωράει κανείς γίνονται όλο και πιο σύνθετα. Τα τελευταία παραδείγματα αποτελούνται από πολλές κλάσεις και μεθόδους και εμπεριέχουν αλληλεπίδραση με τον χρήστη.

1. **demo.a2w**
   Είναι ένα απλό πρόγραμμα με μια κλάση Seth. Το αντικείμενο πραγματοποιεί τις κινήσεις που λέει.
2. **chase 32.a2w**
   Στο πρόγραμμα ένας σκύλος κυνηγάει ένα λαγό και τελικά τον πιάνει. Οι κλάσεις που χρησιμοποιούνται είναι οι Husky και Rabbit.
3. **tortoise and snake.a2w**
   Ένα απλό πρόγραμμα με μια χελώνα και ένα φίδι. Το φίδι κινείται προς την χελώνα και αυτή του λέει να απομακρυνθεί. Οι κλάσεις που χρησιμοποιούνται είναι οι Tortoise και Snake.
4. **revolutionary light.a2w**
   Στο πρόγραμμα αυτό υπάρχει κίνηση και αλλαγή χρωμάτων. Οι κλάσεις είναι οι Light_Bulb, Torus και SphereHighPoly.
5. **bungie_intro_final.a2w**
   Σε αυτό το παράδειγμα δεν υπάρχει αλληλεπίδραση μεταξύ του χρήστη και του προγράμματος. Υπάρχουν οι κλάσεις Robo_penguin, penguin_fanatic1, penguin_fanatic2, Squirrel, Globe και άλλες. Χρησιμοποιούνται εντολές για κίνηση των πιγκουΐνων, της γης και της μαϊμούς.
6. **chrisCannonPlusDad.a2w**
   Ένα κανόνι πυροβολεί πάνω σε ένα σπίτι. Το σπίτι παίρνει φωτιά, ένας άνθρωπος βγαίνει από μέσα πεθαίνει και γίνεται φάντασμα. Οι κλάσεις που χρησιμοποιούνται είναι οι Cannon, CannonBall, FuseFire, FireAnim, Smoke, Ghost.
7. **dance sequencer.a2w**
   Το σκηνικό αποτελείται από την Alice, την Kelly και τον JockProm. Ξεκινάει με ένα τοπίο με γρασίδι και μετατρέπεται σε dance floor. Με το που ξεκινάει να παίζει η μουσική οι χαρακτήρες κάνουν χορευτικές κινήσεις.
8. **helicopter.a2w**
   Ένα σκηνικό που αποτελείται από ένα ελικόπτερο, ένα άνθρωπο και μια αποθήκη. Το πρόγραμμα αυτό είναι ένα μικρού μήκους βίντεο χωρίς καμία αλληλεπίδραση του χρήστη. Μερικές κλάσεις είναι οι Guitar, Hammer, Helicopter, Barn.
9. **Orsega Solar System.a2w**
   Το πρόγραμμα αυτό παρουσιάζει το ηλιακό σύστημα. Ο χρήστης δεν μπορεί να αλληλεπιδράσει με το πρόγραμμα. Απλά γίνεται μια προκαθορισμένη κίνηση των πλανητών. Οι χρησιμοποιούμενες κλάσεις είναι οι Sun, Mercury, Venus, Earth και οι υπόλοιποι πλανήτες.

10. **Nasa on Mars.a2w**
    Σε αυτό το πρόγραμμα γίνεται μια περιγραφή του πλανήτη Άρη από ένα άνθρωπο της Νάσα. Με τα βέλη μπορεί κανείς να κατευθύνει την κάμερα. Μερικές κλάσεις είναι οι Dome, SphereHighPoly, Torus, Tube, Parabola, Dynamite.
11. **Nasa on Neptune.a2w**
    Το ίδιο πρόγραμμα με παραπάνω μόνο που το σκηνικό αλλάζει και πλέον είναι ο πλανήτης Νεύτωνας.
12. **Nasa on the Moon.a2w**
    Το ίδιο πρόγραμμα αλλάζοντας το σκηνικό στο φεγγάρι. Σε αυτό το πρόγραμμα δεν μπορεί ο χρήστης να κινήσει την κάμερα με την χρήση των βελών.
13. **flight.a2w**
    Το πρόγραμμα αποτελεί ένα εξομοιωτή πτήσης αεροπλάνου που απογειώνεται από ένα αεροπλανοφόρο. Ο χρήστης δεν έχει καμία επίδραση στο πρόγραμμα. Οι κλάσεις που χρησιμοποιούνται είναι οι Wingcommander, HappySky, Hill, Tank, Wingman.
14. **knight Fight.a2w**
    Μια μικρή ιστορία μιας πριγκίπισσας, ενός δράκου, μερικών ιπποτών και ενός μάγου. Δεν υπάρχει αλληλεπίδραση με το χρήστη. Μερικές κλάσεις είναι οι Troll, Lance, attack troll 1, Horse, Dragon, Smoke, Fire, Knight, Wizard.
15. **Old Academy.a2w**
    Το σκηνικό είναι ένα παλιό κάστρο μέσα στο οποίο υπάρχουν πολλές κλάσεις. Δεν υπάρχει κάποια αλληλεπίδραση με το χρήστη αλλά γίνονται κάποιες κινήσεις. Για παράδειγμα καίει φωτιά και πέφτει καταρράκτης. Οι κλάσεις που χρησιμοποιούνται είναι οι CloudSky, Temple, Naginata, LightHouse, Bridge, Tent, Volcano.
16. **turtlecookie.a2w**
    Σε αυτό το σκηνικό μια χελώνα πηγαίνει προς ένα σκαμπό που πάνω του έχει ένα κουλουράκι. Η χελώνα κάνει μια συζήτηση με το κουλούρι και στο τέλος τρέπεται σε φυγή. Κάνοντας κλικ με το mouse το σκαμπό απλά περιστρέφεται. Χρησιμοποιούνται κλάσεις όπως Ground, ForestSky, Tortoise, Cookie, Stool, PineTree, Rock, Sapling.
17. **cackle.a2w**
    Υπάρχουν τέσσερις βασικές κλάσεις: Penguin, Monkey, Ladybug και Lemur. Κάνοντας κλικ πάνω σε κάθε ένα αντικείμενο, αυτό κάνει μια κίνηση και λέει μια φράση.
18. **click the Cow.a2w**
    Σε αυτό το πρόγραμμα εμφανίζονται δέκα αγελάδες. Κάνοντας κλικ πάνω σε κάθε μια αυξάνεται η βαθμολογία. Οι κλάσεις που χρησιμοποιούνται είναι Cow, Box, Timer.
19. **ClickOnTheEd.a2w**
    Ο χρήστης κάνει κλικ πάνω στον άνθρωπο Ed όσες περισσότερες φορές μπορεί. Η διάρκεια του παιχνιδιού είναι 30 δευτερόλεπτα. Με το πέρας του χρόνου εμφανίζεται ο αριθμός των κλικ που έχουν γίνει. Χρησιμοποιούμενες κλάσεις: Ed, Times, Go, Bump, Quote.

20. wackateacher.a2w

Όπως και παραπάνω εμφανίζονται πρόσωπα δασκάλων πάνω στα οποία ο χρήστης πρέπει να κάνει κλικ για να αυξήσει την βαθμολογία του. Οι κλάσεις που χρησιμοποιούνται είναι οι teacher, Hole, Booth και Start_Switch.

21. cookie monster.a2w

Σε αυτό το παράδειγμα παίρνει μέρος ένας βάτραχος. Το σκηνικό αποτελείται από κλάσεις όπως Frog, Waterfall, Rock, Cookie, Forest, ForestSky. Με την έναρξη του προγράμματος ο βάτραχος ρωτάει αν μπορεί ο χρήστης να του δώσει ένα κουλουράκι. Εάν ο χρήστης κάνει κλικ στο Yes ο βάτραχος τρώει ένα κουλουράκι, εάν κάνει κλικ στο No τότε ο βάτραχος φωνάζει.

22. Ski Shoot.a2w

Το πρόγραμμα αυτό είναι μια εφαρμογή ενός παιχνιδιού. Σκοπός του παιχνιδιού είναι να συγκεντρωθεί υψηλή βαθμολογία. Κάνοντας κλικ σε τέσσερα κουμπιά η μπάλα κινείται αριστερά, δεξιά, αύξηση ταχύτητας και μείωση ταχύτητας. Έτσι ο χρήστης μπορεί να στοχεύσει σε υψηλά νούμερα. Κάνοντας κλικ στην μπάλα αυτή μετακινείται. Οι κλάσεις που χρησιμοποιούνται για την υλοποίηση του προγράμματος είναι οι PowerSwitch, AimSwitch, LeftSideSquare, Ball, Table.

23. cow Spin.a2w

Το σκηνικό αποτελείται από δέκα αγελάδες. Με το πάτημα του Enter κάθε μια αγελάδα λέει ένα αριθμό. Μόλις πει και η ένατη τον αριθμό εννιά τότε η δέκατη αγελάδα κάνει μερικές περιστροφές. Προφανώς χρησιμοποιούνται δέκα κλάσεις Cow και μια Ground.

24. bike.a2w

Στο πρόγραμμα αυτό ένα παιδί οδηγεί ένα ποδήλατο. Με τα βέλη του πληκτρολογίου ο χρήστης μπορεί να το οδηγήσει. Οι κλάσεις είναι οι Ground και BikeKid.

25. racerball.a2w

Στο πρόγραμμα αυτό ο χρήστης μπορεί να κινήσει μια μπάλα με τα βέλη του πληκτρολογίου. Οι κλάσεις που χρησιμοποιούνται είναι οι Sphere, InnerWall, OuterWall, Square.

26. inside the castle.a2w

Ένα απλό παράδειγμα που τα βέλη μετακινείται η κάμερα μέσα σε ένα σκηνικό. Χρησιμοποιούμενες κλάσεις είναι οι Ground, Knight, Cylinder, 2264 met stained glass (+).

27. lab1.a2w

Όπως και παραπάνω και σε αυτό το σκηνικό με τα βέλη μετακινεί ο χρήστης την κάμερα. Χρησιμοποιούμενες κλάσεις είναι οι Tube, sphereHighPoly, cliff_plants, Cylinder, Cone, Axes και Socrates.

28. Naruto Game Jumping.a2w

Το πρόγραμμα αυτό αποτελείται από τις κλάσεις Ground, Naruto_Front, 10. Με τα βέλη του πληκτρολογίου κινείται ο χαρακτήρας και με το spacebar πηδάει. Σκοπός του παιχνιδιού είναι να πηδήξει ο χαρακτήρας 10 φορές.

29. cow heven.a2w

Στο σκηνικό είναι μια αγελάδα στον ουρανό και περπατάει. Πατώντας το εμπρός βέλος προχωράει και με το δεξί και αριστερό στρίβει προς την αντίστοιχη κατεύθυνση. Ενδεικτικές κλάσεις είναι οι Cow, SphereHighPoly.

30. car Darts.a2w

Η βασική κλάση σε αυτό το πρόγραμμα είναι η Car. Ο χρήστης μπορεί να μετακινήσει το αμάξι δεξιά αριστερά με αποτέλεσμα όταν αυτό τρακάρει ο οδηγός να πεταχτεί κοντά στο κέντρο του στόχου.

31. maze demo.a2w

Σκοπός του παιχνιδιού είναι να φτάσει η λευκή μπάλα στην έξοδο. Σε περίπτωση που η λευκή μπάλα συναντηθεί με τη μαύρη τότε ο χρήστης χάνει. Με τα βέλη γίνεται ο χειρισμός της λευκής μπάλας. Κλάσεις είναι οι: Box, Box2, SphereHighPoly, You Lose, You Win.

32. hedge maze game.a2w

Παιχνίδι λαβύρινθου. Με τα βέλη κινείται ο χαρακτήρας του παιχνιδιού και με το spacebar δημιουργείται ομίχλη για δυσκολότερο επίπεδο. Οι κλάσεις είναι οι HedgeMaze, Box, Lightning.

33. Popup.a2w

Το πρόγραμμα αυτό υλοποιεί ένα παιχνίδι στο οποίο ο χρήστης προσπαθεί να βρει τη κατάλληλη θέση για το κατάλληλο σχήμα. Ο χρόνος που έχει για να τοποθετήσει τα σχήματα στην κατάλληλη θέση είναι περιορισμένος και ορίζεται από τον χρήστη. Μερικές κλάσεις που υλοποιούν το πρόγραμμα είναι οι Popup_Model, Dile_Model, Billboard.

34. buddha.a2w

Σκοπός του παιχνιδιού είναι να βρεθεί ο “Golden Buddha” μέσα σε ένα σπίτι. Ο χρήστης μπορεί να μετακινείται μέσα στον κόσμο με τα βέλη, να ανοίγει πόρτες και να πιάνει πράγματα με το κλικ του mouse. Μπορεί όμως να πετύχει κάποιους εχθρούς και να χάσει. Έτσι οι κλάσεις-εχθροί είναι οι Ghost, Skeleton, Ghost-Zombie. Επίσης υπάρχουν και οι DoorRoom, Buddha, Ground.

35. land scapes.a2w

Σε αυτό το παράδειγμα υπάρχει ένας ιππότης και ένα σκηνικό με κάστρα. Κάνοντας κλικ πάνω στον ιππότη μπορεί να μετακινηθεί. Επίσης με τα βέλη μετακινείται η κάμερα. Χρησιμοποιούμενες κλάσεις είναι οι SphereHighPoly, Square, Knight, SmallBuilding, SmallAppartmentBuilding.

36. haunted bedroom.a2w

Το στοιχειωμένο δωμάτιο αποτελείται από διάφορα αντικείμενα. Κάνοντας κλικ πάνω σε κάθε ένα από αυτά εμφανίζεται κάτι ή μεταμορφώνεται το αντικείμενο. Υπάρχουν πολλές κλάσεις, μερικές είναι οι Room, Clock, ArmChair, Mummy, EndTable, Dresser, Doorway, Skeleton, LightBulb.

37. mystery.a2w

Και σε αυτό το πρόγραμμα είναι ένα δωμάτιο με διάφορα αντικείμενα. Κάνοντας κλικ πάνω σε καθένα από αυτά λαμβάνει χώρα ένα γεγονός. Κάνοντας κλικ για παράδειγμα στο γραμμόφωνο αυτό ξεκινάει να παίζει μουσική. Μερικές ενδεικτικές κλάσεις είναι οι: Ground, Cube, Square, Doorway, Couch, Bookcase, DiningTable, Chair, TV και RemoteControl.

38. mystery 2.a2w

Το σκηνικό αποτελείται από ένα εξωτερικό χώρο όπου υπάρχουν διάφορα κατεστραμμένα αντικείμενα. Με τα βέλη αλλάζει κατεύθυνση η κάμερα και κάνοντας κλικ πάνω σε αντικείμενα αυτά κάνουν μια κίνηση. Μερικές κλάσεις που χρησιμοποιούνται είναι οι: Ground, Capture, Tire, TeddyBear, TV, Toilet.

39. mystery 3.a2w

Στο πρόγραμμα αυτό ένας χαρακτήρας που βρίσκεται σε ένα σπίτι μιλάει. Ο χρήστης μπορεί με το mouse να κινήσει τη κάμερα και με ένα κλικ πάνω στην λάμπα να την ρίξει με αποτέλεσμα το σπίτι να πάρει φωτιά. Μερικές κλάσεις που χρησιμοποιούνται για την υλοποίηση του προγράμματος είναι οι Ground, Square, Chair, StageSpotLight, Doorway, Circle, Fire.

40. In-Game Minigame.a2w

Σε αυτό το παράδειγμα είναι ένα τοπίο με διάφορα αντικείμενα. Κάνοντας κλικ σε καθένα από αυτά, είτε αυτό εξαφανίζεται, είτε κάνει μια κίνηση. Με τα βέλη κινείται η κάμερα προς την αντίστοιχη κατεύθυνση. Με τα πλήκτρα “1”,”2”, “3” αλλάζει η κάμερα οπτική γωνία. Μερικές κλάσεις του προγράμματος είναι οι SmallTable, Swings, ChesireCat, Chicken και Square.

41. holiday Puzzle.a2w

Το κλασσικό Puzzle. Υπάρχουν εννέα αριθμημένα κομμάτια. Με την έναρξη του παιχνιδιού αν θέλει ο χρήστης διαβάζει πληροφορίες για το παιχνίδι αλλιώς τις προσπερνάει. Μετά δίνει το επιθυμητό επίπεδο δυσκολίας. Ξεκινώντας το παιχνίδι αφαιρείται ένα κομμάτι και τα υπόλοιπα ανακατεύονται. Κάνοντας ένα κλικ πάνω σε ένα τετραγωνάκι αυτό πηγαίνει στο κενό. Σκοπός του παιχνιδιού είναι να μπουν τα τετραγωνάκια με την αριθμητική τους σειρά. Κλάσεις είναι οι 1-DkWood, 2-LtWood, Blank, LargeBillbard και άλλες.

42. memory Game.a2w

Το κλασσικό παιχνίδι εύρεσης όμοιων καρτών. Ξεκινώντας ζητάει από το χρήστη των αριθμό των επιθυμητών προσπαθειών. Ανακατεύονται οι κάρτες και ο χρήστης πρέπει να βρει τις όμοιες. Χρησιμοποιούμενες κλάσεις είναι οι: Anchor, Title_Screen, Counter, Win_Loss_Text.

43. light Demonstration.a2w

Στο πρόγραμμα αυτό τα χρησιμοποιούμενα πλήκτρα είναι τα “M”, “Enter”, “Space”, “1”, “2”, “3” και “4”. Με το πάτημα του κάθε πλήκτρου γίνεται ένας συνδυασμός χρωμάτων στην οθόνη. Ενδεικτικές κλάσεις είναι οι SphereHighPoly, lightBulb και άλλες.

44. Strasnick - Robot Defence.a2w

Στο πρόγραμμα αυτό είναι ο χαρακτήρας που κινεί ο χρήστης και ένα ρομπότ που τον πυροβολεί. Σκοπός του παιχνιδιού είναι, με ένα σπαθί, ο χαρακτήρας να αποφύγει και να στείλει πίσω τους πυροβολισμούς ώστε να χάσει ενέργεια το ρομπότ. Η κίνηση του σπαθιού γίνεται με το spacebar. Οι κλάσεις που χρησιμοποιούνται για την υλοποίηση του προγράμματος είναι οι BadGuyRobot, Sphere, Sword.

45. flight Simulator.a2w

Το πρόγραμμα αυτό είναι ένας εξομοιωτής πτήσης. Με τα βέλη το αεροπλάνο κινείται μπρος, πίσω, δεξιά, αριστερά και με το spacebar κάνει

μια περιστροφή. Χρησιμοποιούμενες κλάσεις είναι οι Ground, Temple, Biplane και Carrier.

46. flySeaPlane5.a2w

Το πρόγραμμα αυτό μοιάζει με το παραπάνω απλά αλλάζει το σκηνικό και το αεροπλάνο δεν κάνει περιστροφή, απλά πατώντας το spacebar επιταχύνει. Εδώ υπάρχει και η περίπτωση σύγκρουσης του αεροπλάνου. Χρησιμοποιούμενες κλάσεις είναι οι SmallIsland, BigIsland, SeaPlane, SeaBoat, Smoke, Explosion.

47. abduction.a2w

Στο παράδειγμα αυτό υπάρχουν οι κλάσεις Robot, Alice, Chicken, Turtle, Bunny. Πατώντας τα βέλη του πληκτρολογίου το διαστημόπλοιο-ρομπότ κινείται και όταν φτάσει πάνω από κάποιο αντικείμενο με το πάτημα του spacebar “μαζεύει” το αντικείμενο. Σκοπός του παιχνιδιού είναι να μαζευτούν όλα τα αντικείμενα.

48. asteroids.a2w

Σε αυτό το παράδειγμα υπάρχουν σε ένα τοπίο διαστήματος κλάσεις όπως GreenJumpJet, Asteroids, Spawn. Με τα βέλη κινείται το διαστημόπλοιο και με το πάτημα του spacebar πυροβολεί ώστε να αποφευχθούν οι αστέρες. Σε περίπτωση που κάποιος αστεροειδής πέσει πάνω στο διαστημόπλοιο τότε το παιχνίδι τερματίζεται.

49. bungie_smarty_test.a2w

Εδώ υπάρχουν οι κλάσεις Monkey, Banana, HappyTree, PineTree, Box. Η αλληλεπίδραση μεταξύ χρήστη και προγράμματος γίνεται με το spacebar και με αριστερό κλικ του mouse. Το αντικείμενο Monkey κάνει ένα τεστ στο χρήστη του προγράμματος και σκοπός είναι να απαντηθούν σωστά οι ερωτήσεις.

50. carrierlanding.a2w

Στο πρόγραμμα αυτό υπάρχουν οι κλάσεις: NavyJet, FireAnimation, SmokeAnimation, BeachTerrain. Στην αρχή του προγράμματος υπάρχουν οδηγίες για τα πλήκτρα χειρισμού και επιλογή παιχνιδιού. Ο χρήστης κινεί με τα βέλη το τζετ και πυροβολεί.

51. space invaders.a2w

Το πρόγραμμα αποτελείται από ένα σκηνικό διαστήματος, ένα διαστημόπλοιο το οποίο χειρίζεται ο χρήστης και μερικά εχθρικά διαστημόπλοια. Σκοπός του παιχνιδιού είναι να καταστραφούν τα εχθρικά διαστημόπλοια. Ο χειρισμός γίνεται με τα βέλη του πληκτρολογίου και με το space το διαστημόπλοιο ανοίγει πυρ προς τα αντίπαλα διαστημόπλοια. Οι κλάσεις που χρησιμοποιούνται είναι οι Ground, SmokeAnimation, Alliens_Row_1, Alliens_Row_2, Ship.

52. SpaceInvaders2.a2w

Το πρόγραμμα αυτό είναι ίδιο με το παραπάνω με την διαφορά ότι αλλάζουν τα γραφικά.

53. guitar surfer.a2w

Σε αυτό το πρόγραμμα είναι μια μαϊμού πάνω σε μια κιθάρα. Η μαϊμού μπορεί να κινηθεί προς όλες τις κατευθύνσεις με τα βέλη, μπορεί να πηδήξει με το spacebar, να κάνει απότομη κίνηση μπροστά με το πλήκτρο “A” και απότομη κίνηση πίσω με το πλήκτρο “X”. Ο χρήστης πρέπει να κάνει κλικ πάνω σε ένα κλειδί ώστε να βρει το σωστό που ανοίγει την κλειδαριά. Σε

αυτό το πρόγραμμα υπάρχουν κλάσεις όπως Ground, Guitar, Monkey, Pier, BalletBar, Padlock, WindupKey, Key και cucooClock.

54. guitar_villains.a2w

Ο χρήστης έχει πλήρη αλληλεπίδραση με αυτό το παιχνίδι. Τα πλήκτρα χειρισμού είναι τα “F”, “G”, “J”, “K” και “L”. Υπάρχουν πέντε διάδρομοι με πέντε “αντλίες” διαφορετικών χρωμάτων. Κάθε φορά περνάει από ένα διάδρομο ένας δίσκος. Πατώντας το κατάλληλο πλήκτρο ο χρήστης πρέπει να μαζέψει δίσκους. Κλάσεις που χρησιμοποιούνται είναι οι GreenGuitarAmp, YellowGuitarAmp, Green_Disc, Red_Disc, GreenEndCube, RedEndCube, GreenFlame και Score.

55. lunar Finish Collection.a2w

Το σκηνικό αποτελείται από μερικά διαστημόπλοια, ένα τζετ, ένα κρατήρα, ένα πλανήτη. Με τα βέλη του πληκτρολογίου κινείται το τζετ. Κάνοντας κλικ σε ορισμένα αντικείμενα το εκτελούνται ορισμένες ενέργειες, για παράδειγμα γίνεται μια έκρηξη. Οι κλάσεις που χρησιμοποιούνται είναι οι GreenJumpJet, LunarLander, Crate, Box, Bump.

56. gun game.a2w

Σκοπός του παιχνιδιού είναι να σκοτωθούν όλοι οι πιγκουΐνοι με 10 μόνο σφαίρες. Με τα βέλη κινείται το όπλο και κάνοντας κλικ πυροβολεί. Υπάρχουν κλάσεις όπως Revolver, Bullet, Penguin, No_ammo, AmmoBox και άλλες.

57. city Man Dead End world 1.a2w

Πρόκειται για ένα πρόγραμμα-παιχνίδι είναι ένας άνθρωπος σε ένα πλούσιο τοπίο από γραφικά. Έτσι υπάρχουν αντικείμενα όπως CornerBuilding, Square, GoomedTrees, BirchTrees, Fence, Hotel και πολλά άλλα. Ο χρήστης μπορεί να πυροβολήσει μέσω του ήρωα του παιχνιδιού και πατώντας το κουμπί του mouse να κάνει διάφορες άλλες ενέργειες.

ΚΕΦΑΛΑΙΟ 4
ΣΥΜΠΕΡΑΣΜΑΤΑ

Συμπεράσματα

Στην διπλωματική εργασία αυτή παρουσιάστηκε το εκπαιδευτικό προγραμματιστικό εργαλείο Alice. Η έκδοση που μελετήθηκε είναι το Alice v2.0, παρόλα αυτά το περιβάλλον είναι παρόμοιο και για τις προηγούμενες εκδόσεις. Αρχικά έγινε μια μελέτη και αξιολόγηση του Alice και στην συνέχεια αναπτύχθηκε διδακτικό υλικό το οποίο αποτελεί ένα ολοκληρωμένο εγχειρίδιο χρήσης για το πρόγραμμα Alice για οποιονδήποτε χρήστη, είτε αυτός είναι προγραμματιστής, είτε ένας απλός ενδιαφερόμενος που θέλει να προγραμματίσει με το Alice.

Για τους τελευταίους, τους απλούς ενδιαφερόμενους οι οποίοι δεν έχουν προγραμματιστική εμπειρία το Alice είναι το κατάλληλο εργαλείο. Η δημιουργία ενός προγράμματος με το Alice είναι τόσο εύκολη και διασκεδαστική που κάθε απλός χρήστης θα μπορούσε να κάνει το δικό του πρόγραμμα. Γι αυτό το λόγο το Alice έχει εξελιχθεί σε ένα από τα κυριότερα εργαλεία εκμάθησης προγραμματισμού σε διάφορα σχολεία και γνωστά Πανεπιστήμια της Αμερικής. Οι μαθητές πλέον βρίσκουν ευχάριστο το μάθημα του προγραμματισμού αφού κάθε κόσμο που δημιουργούν μπορούν να τον δουν να εκτελείται άμεσα, να τον τροποποιήσουν και να πειραματιστούν πάνω σε αυτόν. Εξάλλου οι έρευνες που έγιναν σε σχολεία και πανεπιστήμια και τα στοιχεία έδειξαν ότι όχι μόνο αυξήθηκαν οι μαθητές που επιλέγουν το μάθημα του προγραμματισμού, όταν αυτό διδάσκεται με το Alice, αλλά και οι επιδόσεις τους στο μάθημα είναι πολύ καλύτερες κατά μέσο όρο από αυτές άλλων μαθητών που διδάσκονται προγραμματισμό με ένα άλλο εργαλείο.

Η απήχηση του προγραμματιστικού εργαλείου Alice στο ευρύ κοινό έγινε αρωγός στο να θελήσουν οι ιδρυτές του να το αναβαθμίσουν. Για το λόγο αυτό προέκυψε η συνεργασία του Πανεπιστημίου Carnegie Mellon με την EA (Electronic Arts), εταιρία δημιουργίας και παραγωγής videogames. Με την συνεργασία αυτή το Alice θα αναβαθμίσει τις βιβλιοθήκες του με την προσθήκη των χαρακτήρων του παιχνιδιού Sims 2 της EA. Η αναβάθμιση αυτή των χαρακτήρων θα οδηγήσει στη δημιουργία του προγραμματιστικού εργαλείου Alice v3.0.

Το προγραμματιστικό εργαλείο Alice v3.0 αναμένεται να έχει μεγαλύτερη απήχηση στους μαθητές διότι οι χαρακτήρες του Sims 2, λόγω του παιχνιδιού, είναι οικείοι στα περισσότερα άτομα νεαρής ηλικίας. Αν το Alice v3.0 έχει την απήχηση που αναμένεται τότε θα καθιερωθεί ως το βασικό εκπαιδευτικό εργαλείο εκμάθησης προγραμματισμού στα σχολεία της Αμερικής. Θα ήταν χρήσιμο και ενδιαφέρον να ερευνηθούν στο μέλλον οι αλλαγές που θα γίνουν στο πρόγραμμα Alice 3.0 και κατά πόσο αυτό θα έχει την αναμενόμενη επιτυχία.

Βιβλιογραφία

1. Adams J., *Alice in Action with Java*, Thomson Course Technology, 2007.
2. Adams J., "Alice, Middle Schoolers and the Imaginary World Camps", *38th SIGCSE technical symposium on Computer science education*, 307-311, 2007.
3. Cooper S., Dann W., Pausch R., "Teaching objects-first in Introductory Computer Science", *34th SIGCSE technical symposium on Computer science education*, 191-195, 2003.
4. Dann W., Cooper S., Pausch R., *Learning to Program with Alice*, Pearson Prentice Hall, 2006.
5. Herbert C., *An Introduction to Programming Using Alice*, Thomson Course Technology, 2006.
6. Kelleher C., Pausch R., Kiesler S., "Storytelling Alice motivates middle school girls to learn computer programming", *SIGCHI conference on Human factors in computing systems*, 1455-1464, 2007.
7. Moskal B., Lurie D., Cooper S., "Evaluating the Effectiveness of a New Instructional Approach", *35th SIGCSE technical symposium on Computer science education*, 75-79, 2004.
8. Shelly G., Cashman T., Herbert C., *Alice 2.0: Introductory Concepts and Techniques*, Thomson Course Technology, 2006.
9. http://www.acm.org/
10. http://www.alice.org
11. http://www.alicetik2o.com/rpp/index.htm
12. http://www.cs.cmu.edu/~stage3/publications/95/journals/IEEEcomputer/CGandA/paper.html