ΑΛΕΞΑΝΔΡΕΙΟ Τ.Ε.Ι. ΘΕΣΣΑΛΟΝΙΚΗΣ
ΣΧΟΛΗ ΤΕΧΝΟΛΟΓΙΚΩΝ ΕΦΑΡΜΟΓΩΝ
ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ

ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ

Δημιουργία παιχνιδιών με το Game Maker

Φοιτητής

Φτάκλας Αναστάσιος – Τσαμπίκος

Αρ. Μητρώου: 05/2772

Επιβλέπων καθηγητής

Ράπτης Πασχάλης

Θεσσαλονίκη 2013

ΠΡΟΛΟΓΟΣ

Η πτυχιακή αυτή εργασία εκπονήθηκε στα πλαίσια των σπουδών μου στο Τμήμα Πληροφορικής της Σχολής Σ.Τ.ΕΦ του Αλεξάνδρειου Τ.Ε.Ι. Θεσσαλονίκης. Κύριος σκοπός αυτής της εργασίας είναι να εισάγει τους αναγνώστες της, βήμα βήμα, στην τεχνολογία και την «επιστήμη» της δημιουργίας ηλεκτρονικών παιχνιδιών, με την βοήθεια της εφαρμογής Game Maker.

Αρχικά θέλω να ευχαριστήσω τον κύριο Ράπτη Πασχάλη, καθηγητή πληροφορικής του τμήματος ως επιβλέπων καθηγητή της παρούσας εργασίας, για την βοήθειά του στην συγκρότηση και στην ολοκλήρωση αυτής.

Επίσης, θέλω να δώσω θερμές ευχαριστίες σε όλους τους καθηγητές και καθηγήτριες του τμήματος, οι οποίοι μου έδωσαν τις βάσεις και τα ανάλογα προσόντα ώστε να βρίσκομαι αυτή τη στιγμή αντάξιος στις προσδοκίες μιας τέτοιας εργασίας.

Τέλος, ένα μεγάλο ευχαριστώ στην οικογένειά μου και σε όλους αυτούς τους ανθρώπους οι οποίοι στάθηκαν δίπλα μου όλα αυτά τα χρόνια της θητείας μου στο τμήμα και με βοήθησαν με τον τρόπο τους στο να καταφέρω να ενταχθώ στον κόσμο της Πληροφορικής και των Τεχνολογικών Επιστημών.

ΠΕΡΙΛΗΨΗ

Η πτυχιακή αυτή εργασία έχει ως θέμα την **δημιουργία παιχνιδιών με το Game Maker**. Σε αυτό το project θα αναλύσουμε την έννοια των ηλεκτρονικών παιχνιδιών, πως ξεκίνησε η δημιουργία τους, με ποιο σκοπό και με ποιό τρόπο, τα πρώτα ηλεκτρονικά παιχνίδια και γενικά θα αναζητήσουμε πληροφορίες για την ιστορία των περίφημων «games».

Όπως είναι λογικό, στην πορεία θα επικεντρωθούμε στην εφαρμογή του Game Maker. Μια πολλά υποσχόμενη εφαρμογή με πολλές δυνατότητες και μεγάλες μετοχές στον κόσμο της δημιουργίας παιχνιδιών, απλών και μη. Θα δώσουμε βάση κυρίως στις δυνατότητες που έχει για την δημιουργία δισδιάστατων(2D) παιχνιδιών με μια μικρή αναφορά στις ικανότητές της να εισχωρήσει και στον χώρο των τρισδιάστατων(3D) παιχνιδιών. Θα γίνει ανάλυση ένα προς ένα των εργαλείων που διαθέτει, τις ιδιότητες που έχει το κάθε ένα από αυτά καθώς και τα αποτελέσματα που θα έχουμε με τις διάφορες ενέργειες που έχουμε στην διάθεσή μας ως προγραμματιστές ή ως απλοί χρήστες.

Στην συνέχεια θα αναφερθούμε σχολιαστικά στην επίσημη γλώσσα που χρησιμοποιεί το Game Maker, που δεν είναι άλλη από την GML (Game Maker Language) . Θα μιλήσουμε για το συντακτικό της, τις μεταβλητές, τις συναρτήσεις, τις βιβλιοθήκες που διαθέτει και πολλά ακόμη βασικά στοιχεία που την συνθέτουν.

Τέλος, αφού δημιουργήσουμε ένα δικό μας ηλεκτρονικό παιχνίδι για υπολογιστή με την κονσόλα του Game Maker, θα αναλύσουμε ένα προς ένα τα βήματα που ακολουθήσαμε, με πολλά screenshots έτσι ώστε να γίνει πιο κατανοητή και εύκολη η χρήση της. Το παιχνίδι που θα δημιουργήσουμε θα περιλαμβάνει τις περισσότερες από τις χαρακτηριστικές ιδιότητες του Game Maker και με την παρουσίασή τους θα το κάνει ακόμη πιο προσιτό ακόμα και στους αρχάριους χρήστες του. Να τονίσω πως η εργασία αυτή συνοδεύεται από την ηλεκτρονική μορφή του παιχνιδιού που έχουμε φτιάξει, έτσι ώστε ο κάθε ενδιαφερόμενος να μπορεί να κατανοήσει πιο εύκολα τις ενέργειες που έχουν γίνει, στην πράξη.

Με λίγα λόγια η εργασία αυτή δεν επικεντρώνεται μόνο σε ένα κομμάτι του «gaming». Περιλαμβάνει αρκετά στοιχεία από όλα τα κομμάτια που συνθέτουν το πάζλ της δημιουργίας ηλεκτρονικών παιχνιδιών, με έμφαση όμως στο Game Maker.

ABSTRACT

The subject of this project is the **game production with the Game Maker**. Here we will analyze the concept of the video games, how did start the production of them, for which reason and with which way, the first video games on the world and generally we will search out the information about the history of them.

As is reasonable, we will focus on the Game Maker application. One promising application with many features and great stakes in the world of game creation, simple games and complicated. We will mainly base on the potential to create two-dimensional (2D) games with a brief reference to its abilities to penetrate in the world of three-dimensional (3D) games. Will be analyzed one by one of the tools available, the attributes that each one of them and the results that we have with the various actions that we hold as developers or simple users.

Then we will refer to the official programming language of Game Maker, which is none other than the GML (Game Maker Language). We will talk about the syntax of GML, the variables, the functions, the libraries and many more features key elements that compose it.

Finally, after creating our own video game for PC with Game Maker, we will analyze one by one the steps we followed, with many screenshots, to make it more understandable and easy to use. The game we will create will contain most of the characteristic features of Game Maker and the presentation of them will make it even more accessible even to novice users. The project is accompanied by an electronic version of the game that we've made, so that anyone can understand more easily the steps taken.

In short, this work focuses not only on a piece of «gaming». Includes several elements of all the pieces of the puzzle to create computer games, with emphasis on Game Maker.

ΠΕΡΙΕΧΟΜΕΝΑ

ΕΙΣΑΓΩΓΗ

Η επιστήμη της πληροφορικής για πολλούς φαντάζει ανούσια. Άλλοι την θεωρούν σημαντική και άλλοι την χαρακτηρίζουν ως «η επιστήμη του μέλλοντος». Έχει όμως κάποιες διαφορές από τις υπόλοιπες επιστήμες που έχει αναπτύξει ο άνθρωπος, που την κάνουν μοναδική. Η σημαντικότερη αυτών, κατά την προσωπική μου άποψη, είναι οι τεράστιες διαστάσεις της – δεν έχει μία μόνο κατεύθυνση και δεν επικεντρώνεται σε ένα μόνο στοιχείο. Αυτή η λεπτομέρεια και μόνο κάνει αυτομάτως την πληροφορική να αποκτά τεράστιο ενδιαφέρον.

Σε αυτήν την εργασία λοιπόν θα επικεντρωθούμε σε ένα από τα άπειρα «κλαδιά» της πληροφορικής, αυτό που ίσως κεντρίζει το ενδιαφέρον των περισσότερων. Αυτό που ίσως φέρνει πιο κοντά στην πληροφορική ακόμα και αυτούς που την θεωρούν ανούσια ως επιστήμη. Στην δημιουργία ηλεκτρονικών παιχνιδιών.

Μια εφαρμογή για τα πρώτα βήματα (και όχι μόνο) στον χώρο των ηλεκτρονικών παιχνιδιών είναι το Game Maker. Ας ξεκινήσει λοιπόν η «περιήγησή» μας σε αυτόν τον χώρο.

ΚΕΦΑΛΑΙΟ 1
ΕΓΧΕΙΡΙΔΙΟ ΕΚΜΑΘΗΣΗΣ ΤΟΥ GAME MAKER

ΕΙΣΑΓΩΓΗ

Το **Game Maker** αποτελεί μια από τις πιο γνωστές εφαρμογές που έχει ως κύριο στόχο την δημιουργία παιχνιδιών για τον υπολογιστή. Δημιουργός και εκδότης αυτής της εφαρμογής είναι ο Mark Overmars,ο οποίος διδάσκει ως λέκτορας στο ίδρυμα πληροφοριών και επιστημών πληροφορικής του πανεπιστημίου της Ουτρέχτης.

Το Game Maker μπορούμε να το βρούμε στην εξής διεύθυνση http://www.yoyogames.com/ , η οποία είναι και η επίσημη ιστοσελίδα της εταιρίας.

Εικόνα 1.1

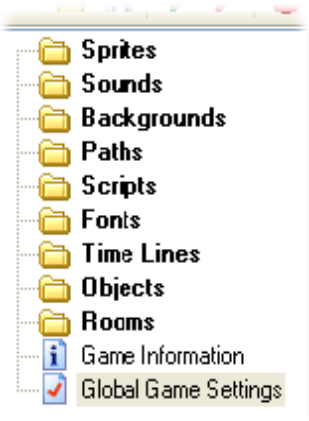
Φυσικά δεν υπάρχει μόνο η έκδοση του Game Maker για τα Microsoft Windows, αλλά και για τα υπόλοιπα κοινώς γνωστά λειτουργικά συστήματα.

Ο σκοπός του **Game Maker** είναι να παράσχει ένα πρόγραμμα κατασκευής δισδιάστατων παιχνιδιών σε ένα συνηθισμένο προσωπικό υπολογιστή. Μπορεί ο οποιοσδήποτε να το χρησιμοποιήσει με την εκμάθηση κάποιων μόνο βασικών λειτουργιών της εφαρμογής. Το πρόγραμμα μπορεί να τρέξει στα περισσότερα λειτουργικά συστήματα ακόμη και σε παλαιότερες εκδόσεις όπως τα WINDOWS 98, εκτός από τα WINDOWS NT 4.0. Με το GAME MAKER είναι δυνατόν κάποιος να δημιουργήσει επίσης και τρισδιάστατα παιχνίδια μαζί με μια πληθώρα άλλων επιλογών, στην Full έκδοσή του .

Σημαντικό και αξιοσημείωτο είναι ότι το Game Maker παρέχει ένα περιβάλλον εύκολα χρησιμοποιούμενο ακόμα και από αρχάριους χρήστες, ενώ ταυτόχρονα, παρέχει τεράστιες δυνατότητες για ευπαρουσίαστα και εξελιγμένα παιχνίδια.

ΥΠΟΚΕΦΑΛΑΙΟ 1.1
ΟΙ ΟΝΤΟΤΗΤΕΣ ΤΟΥ GAME MAKER

Οι «οντότητες» αποτελούν το βασικότερο σκέλος αλλά και το πρώτο, στο οποίο πρέπει να επικεντρωθούμε . Αφού σχεδιάσουμε στο μυαλό μας την βασική ιδέα του παιχνιδιού που θέλουμε να δημιουργήσουμε, τότε ξεκινάμε και την εισαγωγή και δημιουργία των ανάλογων οντοτήτων που θα χρειαστούμε, τις οποίες και θα αναλύσω εκτενέστερα στη συνέχεια. Όταν ανοίξουμε το Game Maker, θα δούμε έναν κατάλογο όλων των διαφορετικών τύπων οντοτήτων που μπορούμε να χρησιμοποιήσουμε, στην αριστερή πλευρά του παραθύρου. Οι οντότητες αυτές είναι τοποθετημένες σε μια λίστα που έχει τη μορφή δέντρου, όπως ακριβώς παρατηρούμε και στο παράθυρο εξερεύνησης αρχείων των Windows, που μπορούν να ανοίξουν και να κλείσουν. Αυτό το μέρος της οθόνης το ονομάζουμε : εξερευνητή των οντοτήτων.



Εικόνα 1.2

Ονομαστικά, οι τύποι οντοτήτων είναι οι εξής:

A. **Sprites,**
B. **Sounds,**
C. **Backgrounds,**
D. **Paths,**
E. **Scripts,**
F. **Fonts,**
G. **Time Lines,**
H. **Objects**
I. **Rooms**

Ας δούμε λοιπόν στην συνέχεια αναλυτικότερα αυτούς τους τύπους οντοτήτων και την λειτουργία τους.

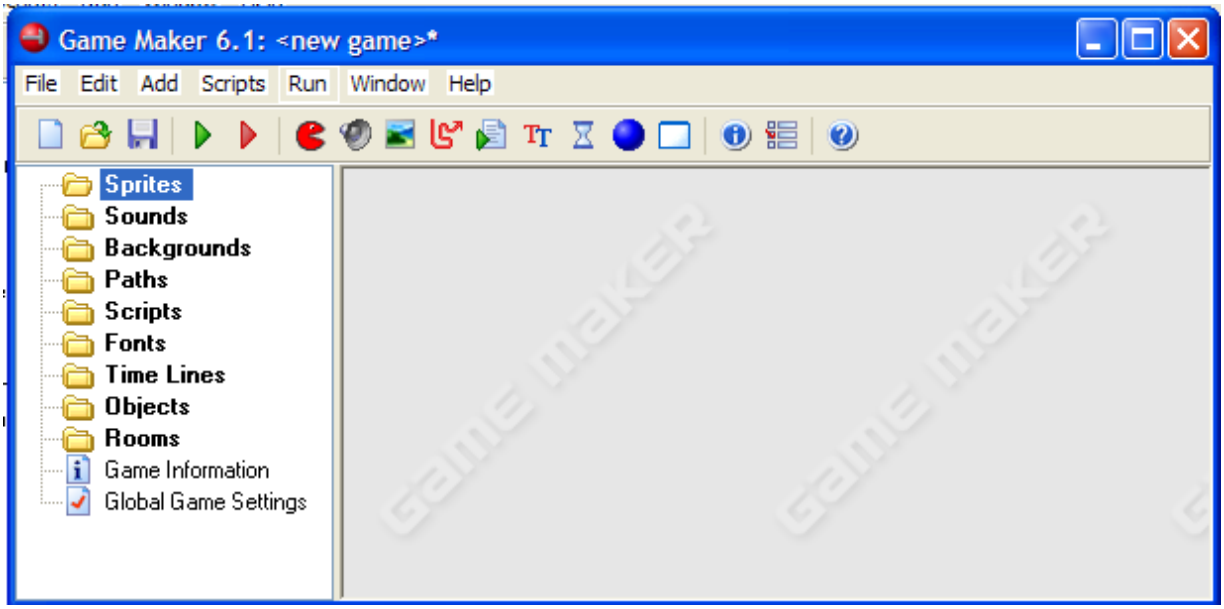
ΥΠΟΚΕΦΑΛΑΙΟ 1.1.1
SPRITES

Η οντότητα αυτή αποτελεί στην ουσία μια εικόνα ή μια σειρά εικόνων (η σειρά εικόνων μας βοηθάει στη δημιουργία κινούμενων χαρακτήρων και αντικειμένων). Ονομάζεται sprite επειδή εκτός από τα δεδομένα για μια εικόνα περιέχει και κάποιες επιπρόσθετες ιδιότητες που αφορούν το πλάτος, το ύψος, τη διαφάνεια, κ.ά.

Το κάθε sprite μπορούμε να το προσθέσουμε στο παιχνίδι είτε με τη φόρτωση αρχείων εικόνας που είναι ήδη αποθηκευμένες στο πρόγραμμα, είτε με τη δημιουργία νέων εικόνων. Να σημειώσουμε πως το format των εικόνων που μπορούμε να φορτώσουμε είναι .ico ή .gif .

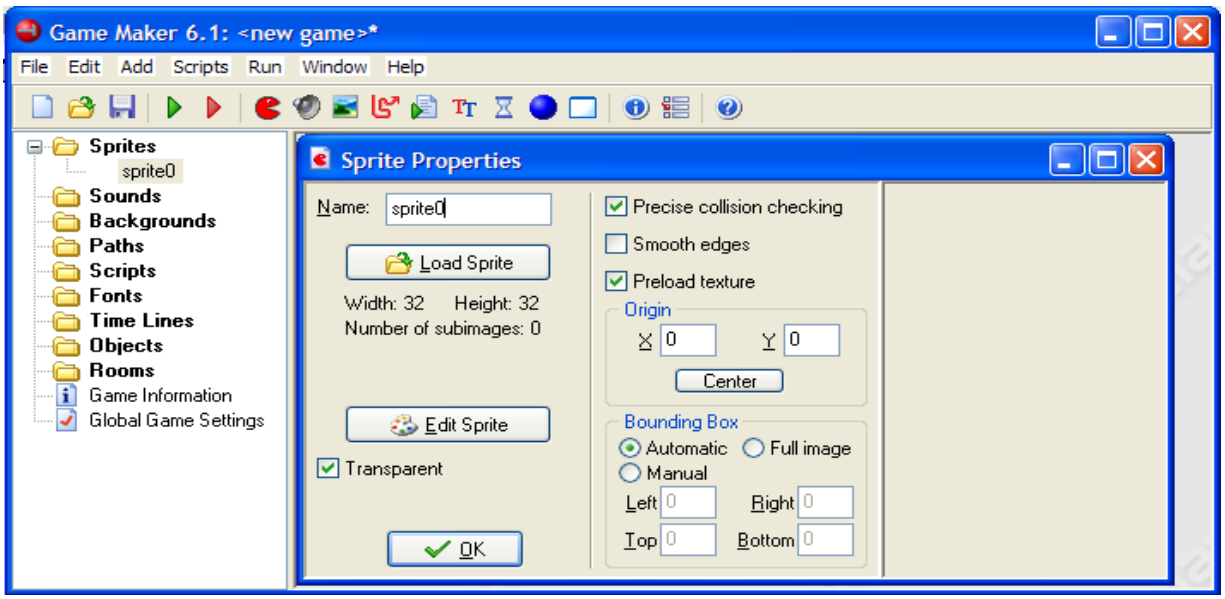
Αρχικά θα δούμε παρακάτω πως γίνεται η δημιουργία μιας νέας οντότητας sprite . Αυτό μπορεί να γίνει με τρεις διαφορετικούς τρόπους:

1. Από το βασικό μενού στην κορυφή της εφαρμογής **Add → Add Sprite**
   (Στις νέες εκδόσεις του Game Maker επιλέγουμε **Resources → Create Sprite**)
2. Με τη βοήθεια του πληκτρολογίου πατώντας ταυτόχρονα **CTRL-SHIFT-S**
3. Με δεξί κλικ του ποντικιού στο **Sprite** στο μενού αριστερά και επιλογή **Create Sprite**



Εικόνα 1.3

Στη συνέχεια εμφανίζεται ένα παράθυρο στο οποίο θέτουμε τα χαρακτηριστικά που θέλουμε να έχει το sprite που θα δημιουργήσουμε



Εικόνα 1.4

Το όνομα που θα δώσουμε στο sprite μπορεί να αποτελείται από λατινικούς χαρακτήρες, μικρούς ή κεφαλαίους (a, b, c, d, e, f … A, B, C, D, E, F,…κ.τ.λ.), από την κάτω παύλα (_) και από αριθμούς (1, 2, 3, 4,…κ.τ.λ.) .

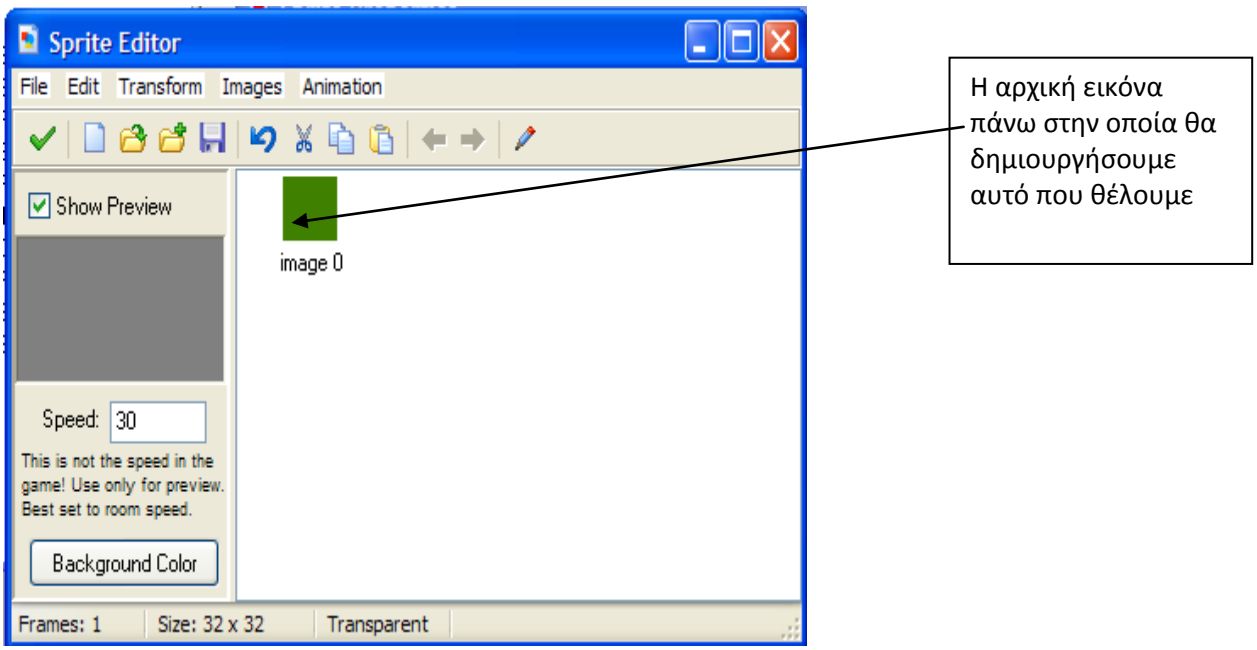
Το όνομα του sprite δεν μπορεί να ξεκινάει από αριθμό, αλλά μόνο από γράμμα της αλφαβήτου. Και να τονίσω, όπως είπαμε και παραπάνω, το μόνο σύμβολο που μπορεί να χρησιμοποιηθεί στο όνομα είναι η κάτω παύλα (_) , κανένα άλλο.

Το όνομα του sprite είναι πάρα πολύ σημαντικό διότι οποιαδήποτε στιγμή θελήσουμε να επεξεργαστούμε το παιχνίδι, θα πρέπει να αναγνωρίζουμε και να ξεχωρίζουμε την κάθε οντότητά μας. Φανταστείτε μια εφαρμογή με 1000 διαφορετικά sprites και άλλα τόσα αντικείμενα κτλ. Συνήθως τα sprites, για μεγαλύτερη ευκολία, ονομάζονται sprXXXX , όπου XXXX κείμενο που μας παραπέμπει στην ιδιότητα του κάθε sprite.

Σε αυτό το σημείο που πλέον η οντότητα μας έχει κάποιο όνομα, έχει σειρά να της δώσουμε μια μορφή, όπου αυτό θα γίνει με την προσθήκη σε αυτήν μιας εικόνας, που όπως προαναφέραμε αυτό μπορεί να γίνει με δύο διαφορετικούς τρόπους.

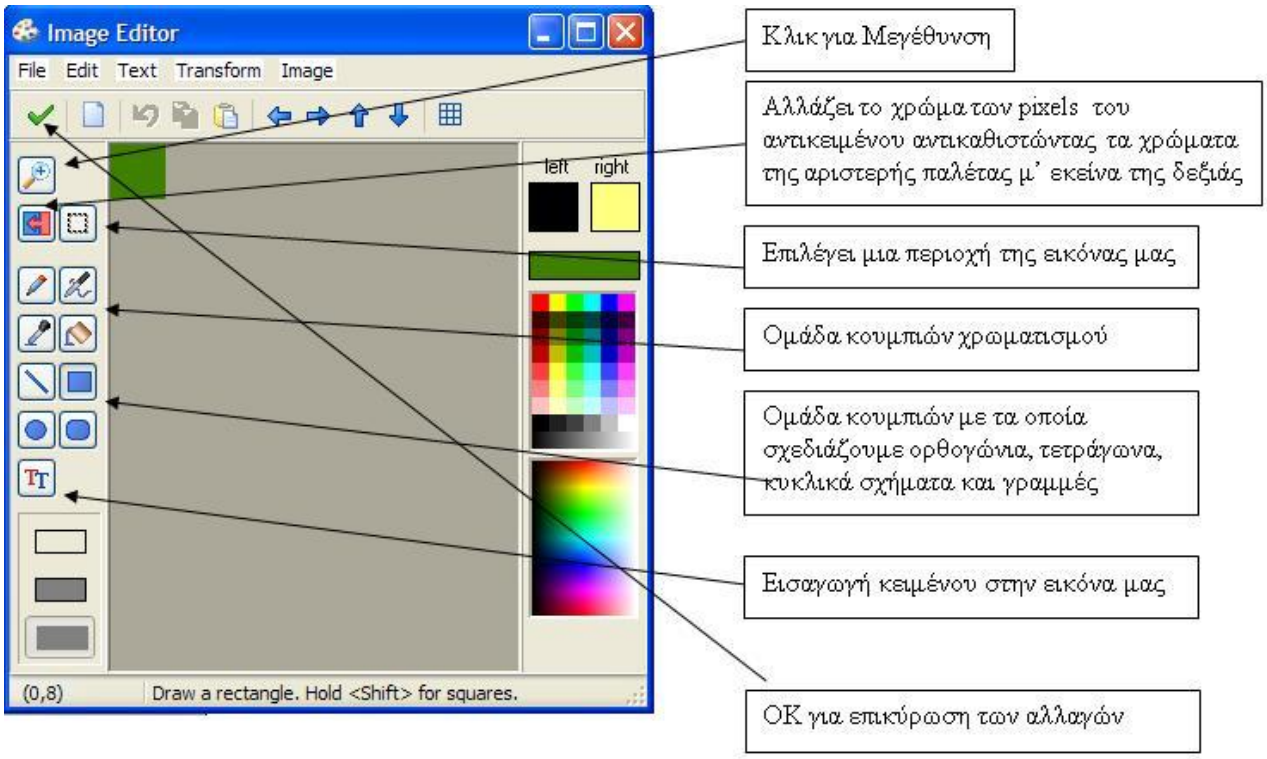
1. Κατά τον πρώτο τρόπο, γίνεται με τη φόρτωση αρχείων εικόνας που είναι ήδη αποθηκευμένες στο πρόγραμμα μέσω της επιλογής : **Load Sprite**
2. Κατά τον δεύτερο τρόπο, γίνεται δημιουργώντας κάποια δικιά μας εικόνα εκείνη τη στιγμή στο Game Maker με τη βοήθεια του κουμπιού : **Edit Sprite.**

Στην δεύτερη περίπτωση, όταν κάνουμε αριστερό κλικ στο **Edit Sprite** θα εμφανιστεί το παρακάτω παράθυρο:



Εικόνα 1.4

Στη συνέχεια επιλέγουμε από το μενού στην κορυφή : **Edit →Edit** ή κάνουμε διπλό κλικ πάνω στην εικόνα, κι έτσι προκύπτει το παράθυρο μορφοποίησης της εικόνας μας : **Image Editor**, στο οποίο θα επεξεργαστούμε τα χαρακτηριστικά της.



Εικόνα 1.5

Αφού προσθέσουμε το sprite στη συνέχεια πάμε στην επιλογή **Backgrounds.** Από εδώ θα προσθέσουμε ένα Background (φόντο) για το παιχνίδι μας από τα διαθέσιμα του Game Maker, παραδείγματος χάριν το Sky.gif, η μπορούμε και πάλι να δημιουργήσουμε το δικό μας φόντο, προσθέτοντας μία εικόνα . Για τα Backgrounds θα μιλήσουμε εκτενέστερα παρακάτω.

ΥΠΟΚΕΦΑΛΑΙΟ 1.1.2
SOUNDS

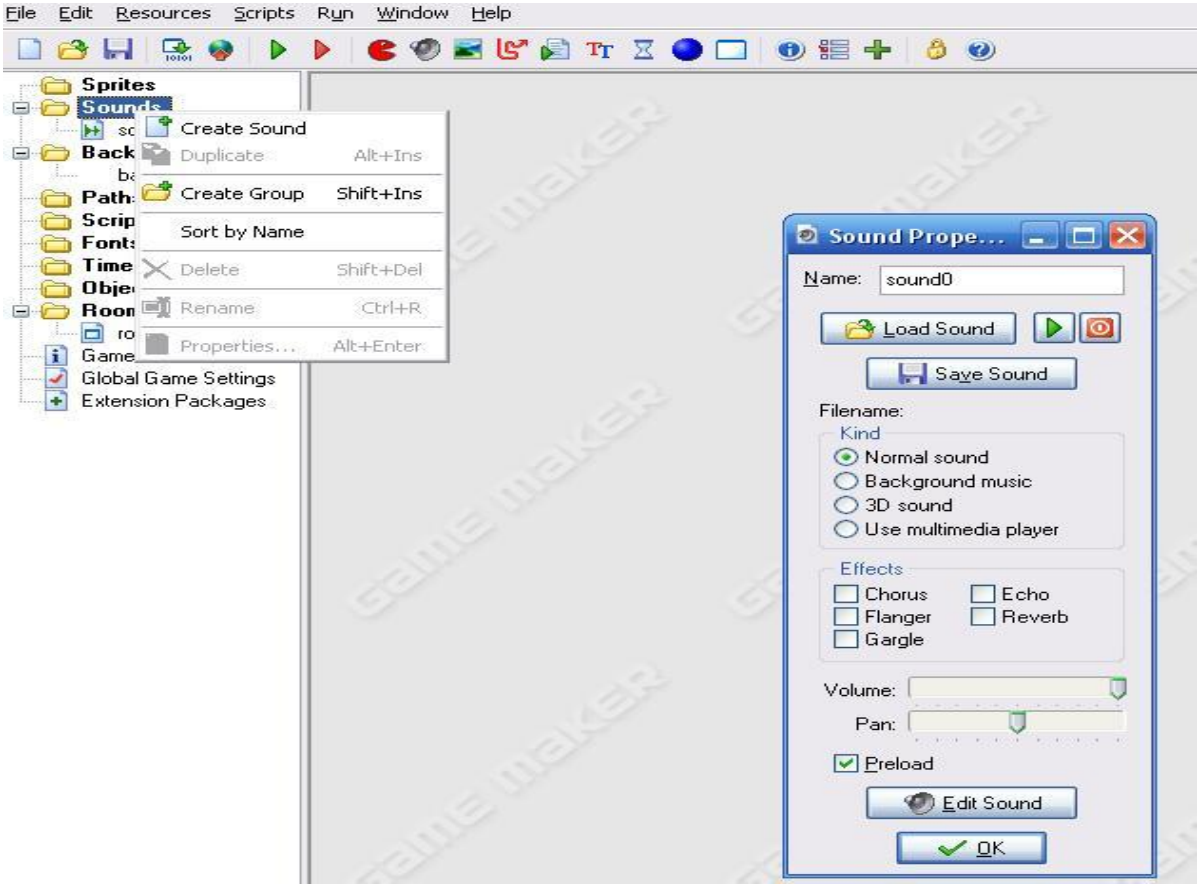
Οι ήχοι και η μουσική (Sounds) αποτελούν επίσης ένα πάρα πολύ σημαντικό κομμάτι για τη δημιουργία ενός παιχνιδιού. Φανταστείτε να παίζετε ένα παιχνίδι, το οποίο δεν έχει καμία μουσική υπόκρουση, δεν ακούγεται κανένας ήχος στις αλληλεπιδράσεις που γίνονται και είναι τελείως βουβό . Επίσης, σημασία δεν έχει απλά να ακούμε κάποιον ή κάποιους ήχους, και αυτοί οι ήχοι να είναι τελείως άσχετοι με το παιχνίδι. Οι ήχοι οι οποίοι θα εισάγουμε θα πρέπει να είναι οι κατάλληλοι και οι ανάλογοι των περιστάσεων, σύμφωνα με το παιχνίδι και το είδος παιχνιδιού που έχουμε σκοπό και στόχο να δημιουργήσουμε. Δεν γίνεται , παραδείγματος χάριν , να παίζει κάποιος ένα τρομακτικό παιχνίδι πολέμου και όταν σκοτώνει κάποια οντότητα του παιχνιδιού να ακούγεται κάποιος χαρούμενος ήχος, ή σαν μουσική υποβάθρου να ακούγεται ένα χαρούμενο τραγούδι.

Μπορούμε αν θέλουμε να προσθέσουμε κάποια δικά μας ηχητικά Clips, ή κάποια δικά μας τραγούδια που έχουμε αποθηκευμένα στον υπολογιστή στον οποίο εργαζόμαστε. Αν θέλουμε όμως, έχουμε την δυνατότητα να επιλέξουμε κάποιους από τους ήχους που υπάρχουν στη επίσημη διαδικτυακή σελίδα του Game Maker, ή και σε άλλες ιστοσελίδες (δίνεται η δυνατότητα της ανάκτησης ολόκληρων φακέλων με ήχους, ώστε να επιλέξουμε τους κατάλληλους για το εκάστοτε παιχνίδι που δημιουργούμε).

Για να εισάγουμε έναν ήχο στο παιχνίδι μας, πρέπει να ακολουθήσουμε έναν από τους παρακάτω τρόπους :

1. Από το βασικό μενού στην κορυφή της εφαρμογής **Add → Add Sound**
   (Στις νέες εκδόσεις του Game Maker επιλέγουμε **Resources → Create Sound**)
2. Με τη βοήθεια του πληκτρολογίου πατώντας ταυτόχρονα **CTRL-SHIFT-U**
3. Με δεξί κλικ του ποντικιού στο **Sounds** στο μενού αριστερά και επιλογή **Create Sound**

Κατά την δημιουργία (εισαγωγή) ενός ήχου, υπάρχουν διάφοροι παράμετροι και ιδιότητες, που μπορούμε να επεξεργαστούμε , ανάλογα με τις απαιτήσεις του παιχνιδιού που δημιουργούμε. Πολύ σημαντική παρατήρηση, είναι η επιλογή που μας δίνεται, ώστε να ορίσουμε , για τον εκάστοτε ήχο , να φορτώνεται στο παιχνίδι μας την χρονική στιγμή που έχουμε ορίσει, και να μην φορτώνεται από την αρχή, κατά την εκκίνηση του παιχνιδιού. Αυτό βοηθάει στην μνήμη του υπολογιστή στο οποίο πρόκειται να τρέξει το παιχνίδι, στους πόρους του συστήματος που θα χρειαστεί να διατεθούν, και στο μέγεθος του αποθηκευτικού χώρου στο δίσκο που θα χρειαστεί να δαπανηθεί .



Εικόνα 1.6

Name: Στο πεδίο αυτό θα εισάγουμε το όνομα του **Sound**. Και εδώ, έχει πάρα πολύ μεγάλη σημασία το όνομα που θα δώσουμε στον κάθε ήχο , για προφανείς λόγους όπως προείπαμε και για τα **Sprites**

Στο Game Maker, καλώς η κακώς, δεν υπάρχει κάποιος Editor για την δημιουργία ήχων. Μπορούμε όμως να χρησιμοποιήσουμε άλλους επεξεργαστές ήχων, να δημιουργήσουμε τους ήχους που χρειαζόμαστε και να τους μεταφορτώσουμε στην εφαρμογή μας. Όταν μεταφορτώσουμε έναν ήχο, μπορούμε να τον αποθηκεύσουμε στο Game Maker, ώστε σε περίπτωση που χάσουμε το πρωτότυπο αρχείο ήχου, να μην δημιουργηθεί κάποιο πρόβλημα.

Μας δίνεται η δυνατότητα μεταφόρτωσης τριών (3) διαφορετικών τύπων ηχητικών αρχείων , οι οποίοι είναι οι εξής : MP3 , Midi , Wave .

Υπάρχουν δύο buttons , ένα πράσινο (play), με το οποίο μπορούμε να τεστάρουμε τον ήχο που φορτώσαμε, κι ένα κόκκινο (stop), με το οποίο σταματάμε την αναπαραγωγή του συγκεκριμένου ήχου. Εάν δεν πατήσουμε το stop, ο ήχος συνεχίζει να παίζει ασταμάτητα και επαναλαμβανόμενα.

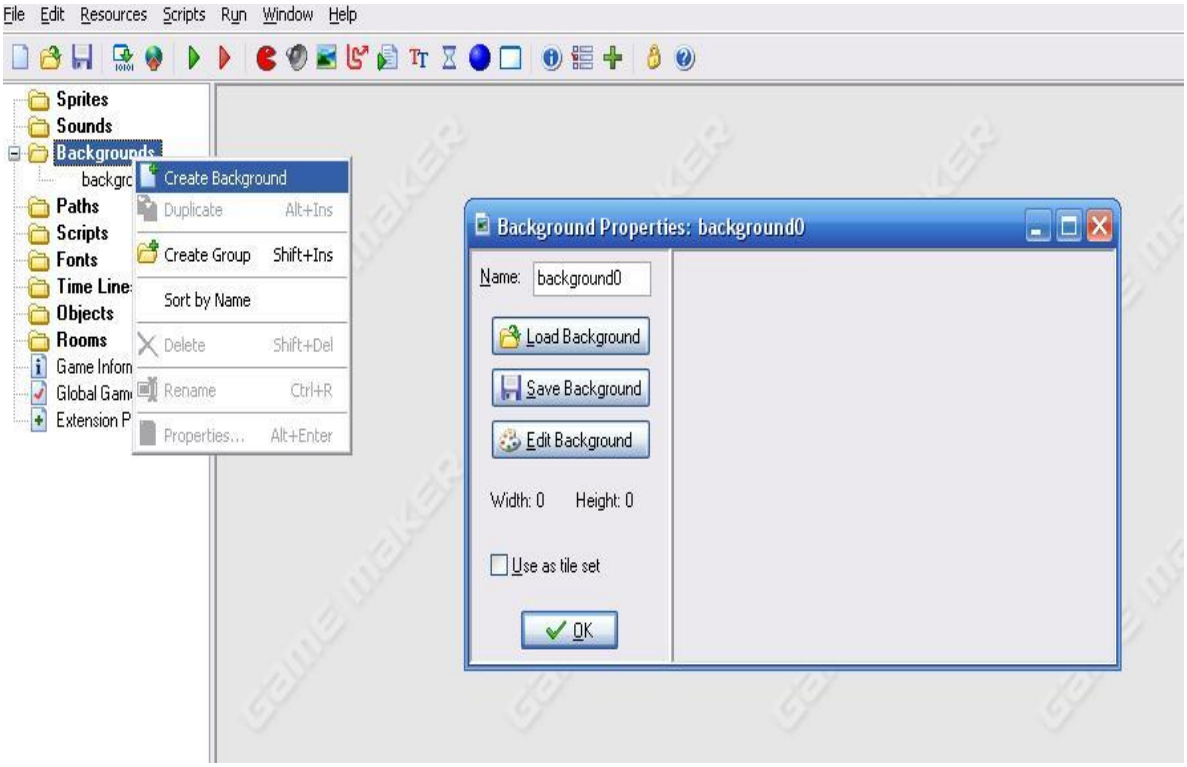
ΥΠΟΚΕΦΑΛΑΙΟ 1.1.3
BACKGROUDS

Το κάθε δωμάτιο (η κάθε πίστα) του παιχνιδιού που έχουμε ξεκινήσει να δημιουργούμε, θα πρέπει να έχει και ένα φόντο, ή αλλιώς ένα υπόβαθρο ή όπως είναι ευρέως γνωστό στο χώρο της πληροφορικής, ένα Background. Το φόντο αυτό, είναι στην ουσία μια εικόνα, η οποία προφανώς θα είναι μονίμως πίσω από τους χαρακτήρες και τα αντικείμενα του παιχνιδιού μας. Αλλά έχουμε και τη δυνατότητα να επιλέξουμε εάν θέλουμε η εικόνα αυτή να έχει κάποια διαφάνεια ή όχι. Το Game Maker, μας δίνει τη δυνατότητα να επιλέξουμε ως φόντο μία από τις εικόνες που έχει ήδη εγκατεστημένες εξ αρχής στις βιβλιοθήκες του. Επίσης, στο διαδίκτυο υπάρχουν πάρα πολλές , δωρεάν ή μη, βιβλιοθήκες, από επίσημες ιστοσελίδες του παιχνιδιού είτε όχι, από τις οποίες μπορούμε να αντλήσουμε διάφορα φόντα, αφού τις εγκαταστήσουμε πρώτα στην εφαρμογή μας. Τέλος, μας δίνεται η δυνατότητα να δημιουργήσουμε εξ αρχής τις δικιές μας εικόνες, ώστε να τις χρησιμοποιήσουμε στην συνέχεια ως φόντο σε κάποιο δωμάτιο (Room) του παιχνιδιού που σχεδιάζουμε. Αυτό μπορεί να γίνει είτε στην ειδική πλατφόρμα που μας παρέχει το Game Maker, είτε απλά μεταφορτώνοντας κάποιες από τις εικόνες που έχουμε ήδη αποθηκευμένες στον υπολογιστή στον οποίο εργαζόμαστε .

Για να εισάγουμε ένα φόντο (Background) στο παιχνίδι μας ώστε να μπορέσουμε στη συνέχεια να το χρησιμοποιήσουμε σε κάποιο δωμάτιο (Room) , πρέπει να ακολουθήσουμε έναν από τους παρακάτω τρόπους :

1. Από το βασικό μενού στην κορυφή της εφαρμογής **Add → Add Background** (Στις νέες εκδόσεις του Game Maker επιλέγουμε **Resources → Create Background**)
2. Με τη βοήθεια του πληκτρολογίου πατώντας ταυτόχρονα **CTRL-SHIFT-B**
3. Με δεξί κλικ του ποντικιού στο **Backgrounds** στο μενού αριστερά και επιλογή **Create Background**

Για να τροποποιήσουμε ένα ήδη υπάρχον φόντο ή για να δημιουργήσουμε ένα νέο στην πλατφόρμα του παιχνιδιού, εκτελούμε το εξής : **Edit Background** στη φόρμα “Background Properties” .

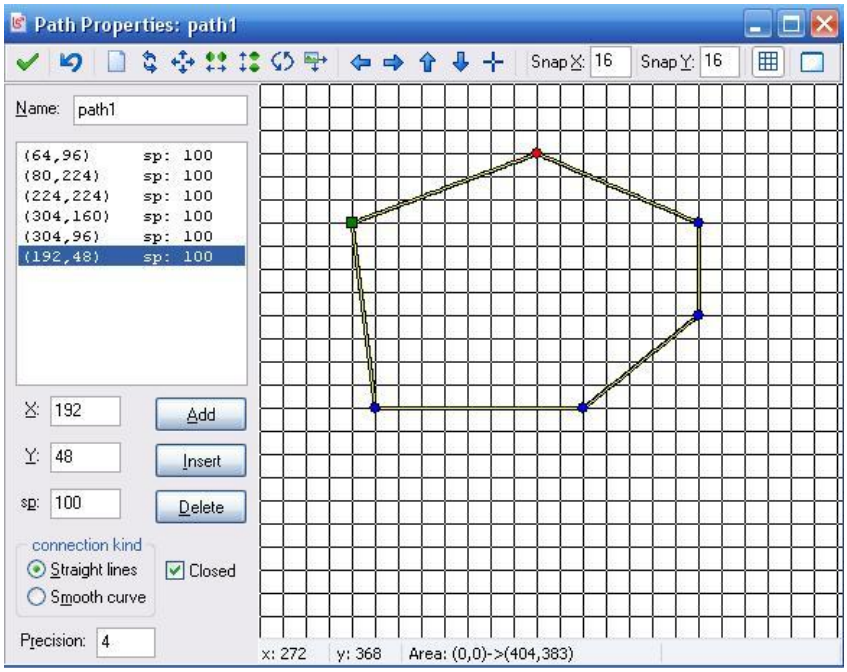


Εικόνα 1.7

Name: Στο πεδίο αυτό θα εισάγουμε το όνομα του **Background** που δημιουργήσαμε, που εισάγαμε ή που επιλέξαμε. Και πάλι η ονομασία που θα δώσουμε έχει τεράστια σημασία ,για προφανείς λόγους όπως προείπαμε και για τα **Sprites** .

ΥΠΟΚΕΦΑΛΑΙΟ 1.1.4
PATHS

Πολλές φορές θέλουμε τα βασικά μέλη του παιχνιδιού που δημιουργούμε να ακολουθήσουν κάποιο συγκεκριμένο «μονοπάτι». Είναι πολύ απλό. Δημιουργούμε ένα μονοπάτι σχεδιάζοντάς το και στην συνέχεια τοποθετούμε και εφαρμόζουμε τα ανάλογα events που θέλουμε να λάβουν χώρα στην πορεία που θα ακολουθήσει κάποιο αντικείμενό μας.



Εικόνα 1.8

Για να δημιουργήσουμε ένα νέο Path στο παιχνίδι μας, πρέπει να ακολουθήσουμε έναν από τους παρακάτω τρόπους :

1. Κλικ στο κουμπί «**Create a Path»**
2. Κλικ στην επιλογή **Resources → Create Path** από το μενού πάνω
3. Πληκτρολογώντας ταυτόχρονα από το πληκτρολόγιο **CTRL- SHIFT -P**
4. Με δεξί κλικ στο **Paths** από το μενού αριστερά και επιλογή **Create Path**

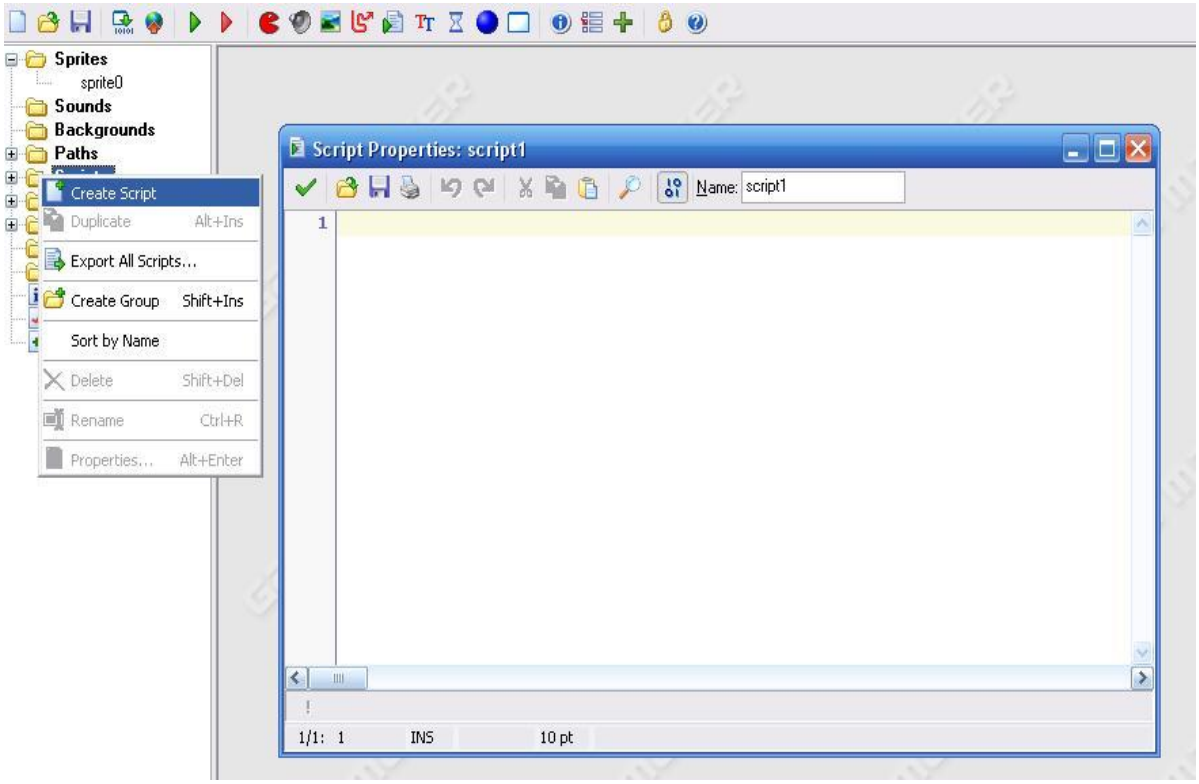
ΥΠΟΚΕΦΑΛΑΙΟ 1.1.5
SCRIPTS

Η οντότητα αυτή υπάρχει, ώστε να μπορούμε να γράψουμε ολόκληρα κομμάτια κώδικα από μόνοι μας, να τα αποθηκεύσουμε και να τα χρησιμοποιήσουμε όποτε και όπου θέλουμε στο παιχνίδι που φτιάχνουμε. Όποιος γνωρίζει και έχει ασχοληθεί με τον προγραμματισμό, θα γνωρίζει πως σε εφαρμογές όπως το Game Maker που το περιβάλλον εργασίας του είναι εύκολο και προσιτό σχεδόν για όλους, (δεν χρειάζονται γνώσεις προγραμματισμού για την χρήση του), η δημιουργία δικών μας κομματιών κώδικα πέρα από αυτά που έχει, βοηθάει στην «διείσδυση» στη λεπτομέρεια. Αυτό φυσικά μας δίνει πολύ καλύτερα αποτελέσματα. Συνήθως αποθηκεύουμε κομμάτια κώδικα σε Scripts τα οποία αποτελούν «ρουτίνες» τις οποίες πρόκειται να επαναλάβουμε πολλές φορές στο παιχνίδι μας.

Η γλώσσα που χρησιμοποιείται στο Game Maker είναι η **GML** (**Game Maker Language**) . Με τον τρόπο σύνταξης της **GML** θα ασχοληθούμε εκτενέστερα στο ανάλογο κεφάλαιο.

Για να εισάγουμε Script στο παιχνίδι μας ώστε να μπορέσουμε στη συνέχεια να το χρησιμοποιήσουμε όπου το χρειαζόμαστε , πρέπει να ακολουθήσουμε έναν από τους παρακάτω τρόπους :

1. Κλικ στην επιλογή **Add → Add Script** από το μενού πάνω
(Στις νέες εκδόσεις του Game Maker επιλέγουμε **Resources → Create Script**)
2. Πληκτρολογώντας ταυτόχρονα από το πληκτρολόγιο **CTRL- SHIFT -C**
3. Με δεξί κλικ στο **Scripts** από το μενού αριστερά και επιλογή **Create Script**



Εικόνα 1.9

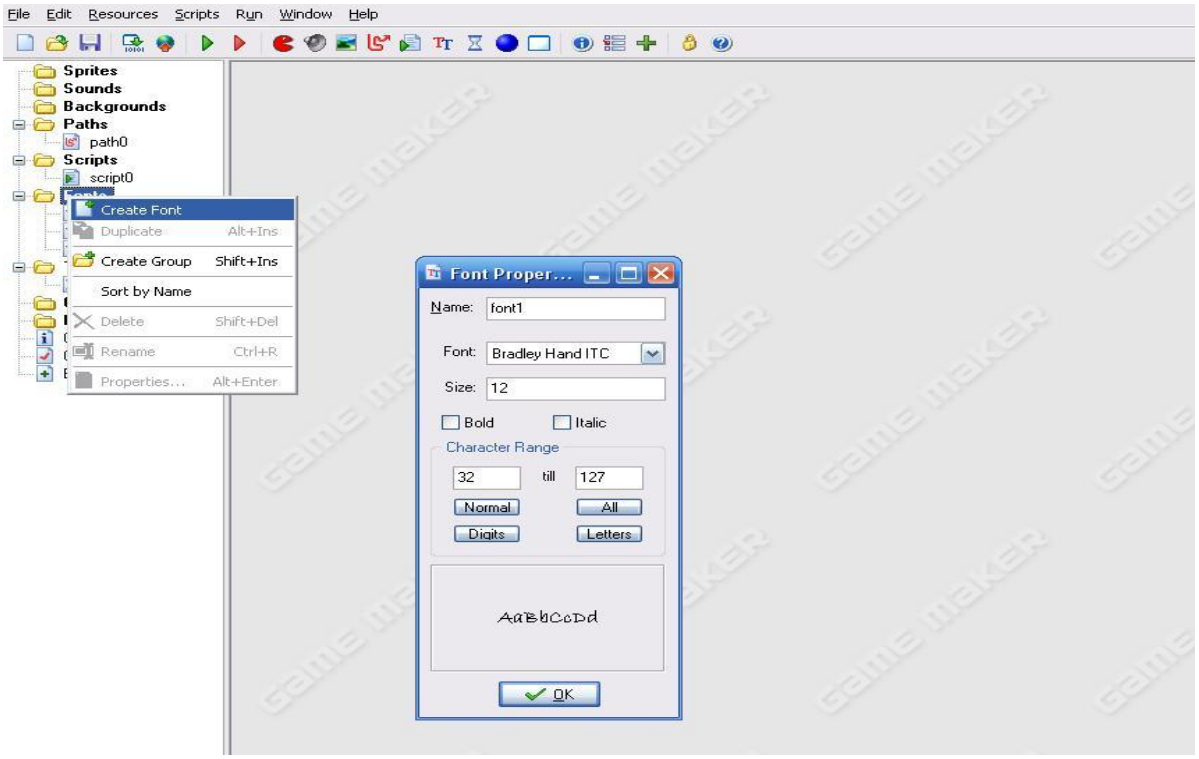
Name: Στο πεδίο αυτό θα εισάγουμε το όνομα του **Script** που δημιουργήσαμε. Και πάλι η ονομασία που θα δώσουμε έχει τεράστια σημασία ,για προφανείς λόγους όπως προείπαμε και για τα **Sprites** .

ΥΠΟΚΕΦΑΛΑΙΟ 1.1.6
FONTS

Σε αυτήν την οντότητα, εισάγουμε και αποθηκεύουμε μια γραμματοσειρά, συμπεριλαμβάνοντας όλες τις παραμέτρους , όπως για παράδειγμα το Bolt , το Italic Style , το μέγεθος της γραμματοσειράς και τα λοιπά. Έτσι, εάν σε κάποιο σημείο του παιχνιδιού θέλουμε να αναγράφεται κάτι, έχουμε ήδη έτοιμη και αποθηκευμένη την γραμματοσειρά που θέλουμε και την εφαρμόζουμε στο εκάστοτε κείμενο. Αυτό γίνεται κυρίως και για ευνόητους λόγους ταχύτητας, διότι μια αποθηκευμένη γραμματοσειρά μπορούμε να την χρησιμοποιήσουμε σε περισσότερα από ένα σημεία στο παιχνίδι μας, κι έτσι δεν χρειάζεται να επαναλαμβάνουμε την διαδικασία ρύθμισης της γραμματοσειράς, κάθε φορά που την χρειαζόμαστε.

Για να εισάγουμε μια γραμματοσειρά (Font) ώστε να μπορέσουμε στη συνέχεια να τη χρησιμοποιήσουμε σε κάποιο ή κάποια σημεία του παιχνιδιού , πρέπει να ακολουθήσουμε έναν από τους παρακάτω τρόπους :

1. Κλικ στην επιλογή **Add → Add Font** από το μενού πάνω
   (Στις νέες εκδόσεις του Game Maker επιλέγουμε **Resources → Create Font**)
2. Πληκτρολογώντας ταυτόχρονα από το πληκτρολόγιο **CTRL- SHIFT -F**
3. Με δεξί κλικ στο **Fonts** από το μενού αριστερά και επιλογή **Create Font**

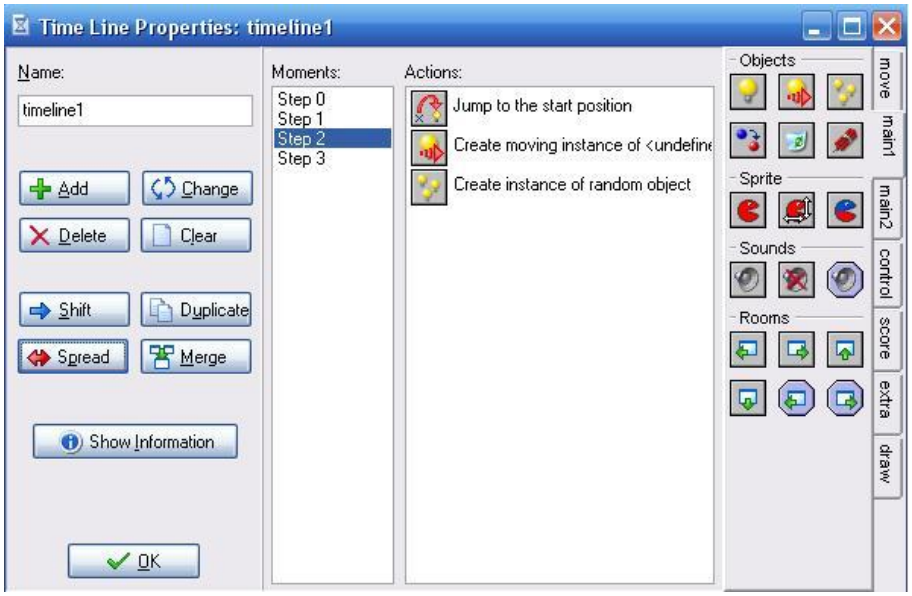


Εικόνα 1.10

Name: Στο πεδίο αυτό θα εισάγουμε το όνομα του **Font** που θέλουμε να αποθηκεύσουμε. Και πάλι η ονομασία που θα δώσουμε έχει τεράστια σημασία ,για προφανείς λόγους όπως προείπαμε και για τα **Sprites** .

ΥΠΟΚΕΦΑΛΑΙΟ 1.1.7
TIME LINES

Πολλές φορές θέλουμε κάποια πράγματα να συμβούν σε κάποιες συγκεκριμένες χρονικές στιγμές. Αυτό κάποιες φορές είναι εφικτό με το Alarm(για το οποίο θα μιλήσουμε εκτενέστερα στην πορεία). Όταν όμως τα δεδομένα δεν είναι τόσο απλά, η μέθοδος του Alarm περιπλέκει τα πράγματα και δεν έχουμε το ανάλογο επιθυμητό αποτέλεσμα. Ας δούμε ένα μικρό παράδειγμα :



Εικόνα 1.11

Για να εισάγουμε μια γραμματοσειρά (Font) ώστε να μπορέσουμε στη συνέχεια να τη χρησιμοποιήσουμε σε κάποιο ή κάποια σημεία του παιχνιδιού , πρέπει να ακολουθήσουμε έναν από τους παρακάτω τρόπους :

1. Κλικ στην επιλογή **Resources → Create Time Line**από το μενού πάνω
2. Πληκτρολογώντας ταυτόχρονα από το πληκτρολόγιο **CTRL- SHIFT -T**
3. Με δεξί κλικ στο **Time Lines** από το μενού αριστερά και επιλογή **Create Time line**

ΥΠΟΚΕΦΑΛΑΙΟ 1.1.8
OBJECTS

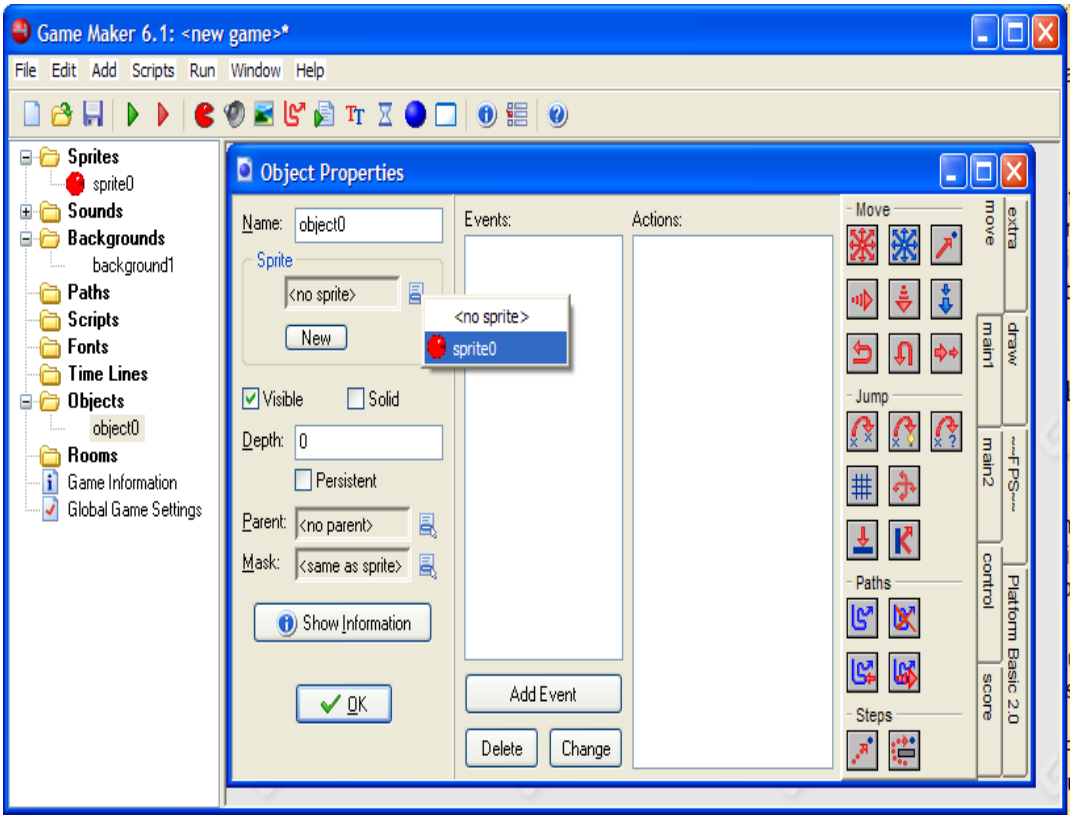
Τα **Objects** (αντικείμενα) αποτελούν τη σημαντικότερη οντότητα του Game Maker. Είναι τα έμβια, δηλαδή τα ενεργά αντικείμενα του κάθε παιχνιδιού που θα δημιουργήσουμε. Το κάθε object συνήθως αντιπροσωπεύεται και από ένα sprite, αλλά όχι πάντα.

Για να εισάγουμε ένα object στο παιχνίδι μας, πρέπει να ακολουθήσουμε έναν από τους παρακάτω τρόπους :

1. Κλικ στο κουμπί «**Add an Object»**
2. Κλικ στην επιλογή **Add → Add Object** από το μενού πάνω
   (Στις νέες εκδόσεις του Game Maker επιλέγουμε **Resources → Create Object**)

3. Πληκτρολογώντας ταυτόχρονα από το πληκτρολόγιο **CTRL- SHIFT -O**
4. Με δεξί κλικ στο **Objects** από το μενού αριστερά και επιλογή **Create Object**

Παρατηρούμε πως εμφανίζεται ένα παράθυρο όπου θα επιλέξουμε τις ιδιότητες που θέλουμε να έχει το αντικείμενό μας



Εικόνα 1.12

Θα εξηγήσουμε λοιπόν τι ακριβώς σημαίνουν οι ιδιότητες αυτές που μας ζητάει να προσαρμόσουμε στο αντικείμενο που θέλουμε να δημιουργήσουμε :

Name: Στο πεδίο αυτό θα εισάγουμε το όνομα του **Object**. Ξεκινούμε την ονομασία των **Objects** συνήθως με **obj**XXXX, για προφανείς λόγους όπως προείπαμε και για τα **Sprites** .

Sprite: Στο πεδίο αυτό επιλέγουμε κάποιο από τα **Sprite** που έχουμε ήδη κατασκευάσει προηγουμένως. (Να σημειώσουμε εδώ πως ένα **Sprite** δεν είναι τίποτα απολύτως από μόνο του, δεν μας χρησιμεύει πουθενά για το παιχνίδι που έχουμε ξεκινήσει να φτιάχνουμε, παρά μόνο όταν το «συνδέσουμε» με κάποιο **Object** (αντικείμενο) .

Μας δίνεται η δυνατότητα να τσεκάρουμε ή όχι δύο επιλογές για το αντικείμενό μας :

Visible: Όταν είναι επιλεγμένο σημαίνει ότι το **Sprite** θα είναι ορατό καθ' όλη τη διάρκεια του παιχνιδιού (Υπάρχουν περιπτώσεις όπου θέλουμε σε κάποιο σημείο του παιχνιδιού, να υπάρχουν απτά αντικείμενα τα οποία θα έχουν την ανάλογη αλληλεπίδραση με τα υπόλοιπα, αλλά θέλουμε να μην φαίνονται, να μην είναι ορατά, π.χ. ένας αόρατος τοίχος, τον οποίο ο βασικός μας κινούμενος χαρακτήρας να μην μπορεί να τον διαπεράσει).

Solid: Όταν είναι επιλεγμένο σημαίνει ότι το sprite είναι συμπαγές και επομένως αδιαπέραστο από οποιοδήποτε αντικείμενο που κινείται μέσα στο παιχνίδι, για παράδειγμα ένας τοίχος ή το δάπεδο ή κάποιο εμπόδιο.

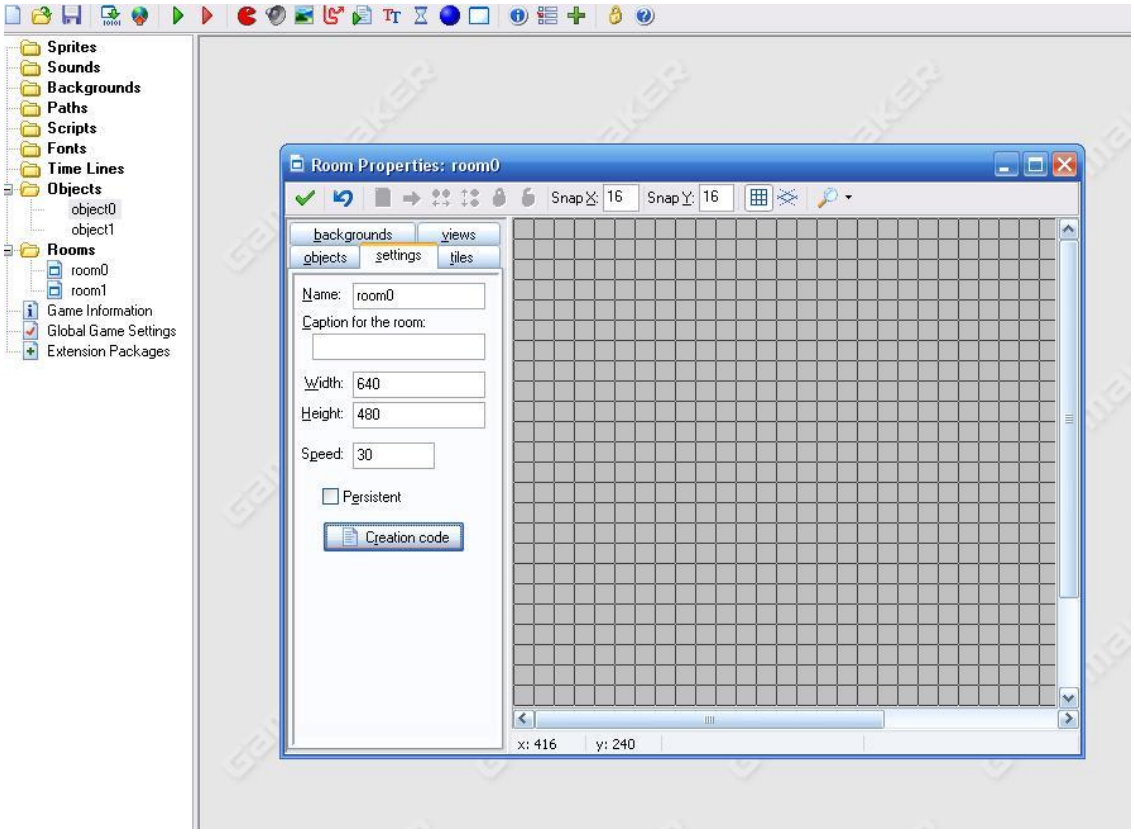
ΥΠΟΚΕΦΑΛΑΙΟ 1.1.9
ROOMS

Το κάθε παιχνίδι που δημιουργούμε διαδραματίζεται σε ένα ή περισσότερα δωμάτια («σκηνές») τα οποία στο Game Maker ονομάζονται **rooms**. Τα δωμάτια (rooms) αυτά αποτελούν διάφορα επίπεδα του παιχνιδιού μας, τα οποία είναι μη τρισδιάστατα, αλλά μπορούν να περιέχουν γραφικά που μοιάζουν με 3D. Τα rooms αντιστοιχούν στις λεγόμενες πίστες ή οθόνες ή επίπεδα των ηλεκτρονικών παιχνιδιών.

Για να εισάγουμε ένα δωμάτιο στο παιχνίδι μας, πρέπει να ακολουθήσουμε έναν από τους παρακάτω τρόπους :

1. Κλικ στην επιλογή **Add → Add Room** από το μενού πάνω
   (Στις νέες εκδόσεις του Game Maker επιλέγουμε **Resources → Create Room**)
2. Πληκτρολογώντας ταυτόχρονα από το πληκτρολόγιο **CTRL- SHIFT -R**
3. Με δεξί κλικ στο **Rooms** από το μενού αριστερά και επιλογή **Create Room**

Δημιουργώντας ένα νέο δωμάτιο (Η επιλέγοντας ένα από αυτά που έχουμε ήδη δημιουργήσει), μας δίνεται η δυνατότητα να επεξεργαστούμε κάποιες ιδιότητες



Εικόνα 1.13

Name: Στο πεδίο αυτό θα εισάγουμε το όνομα του **Room**. Ξεκινούμε την ονομασία των δωματίων συνήθως με **room**XXXX, για προφανείς λόγους όπως προείπαμε και για τα **Sprites** .

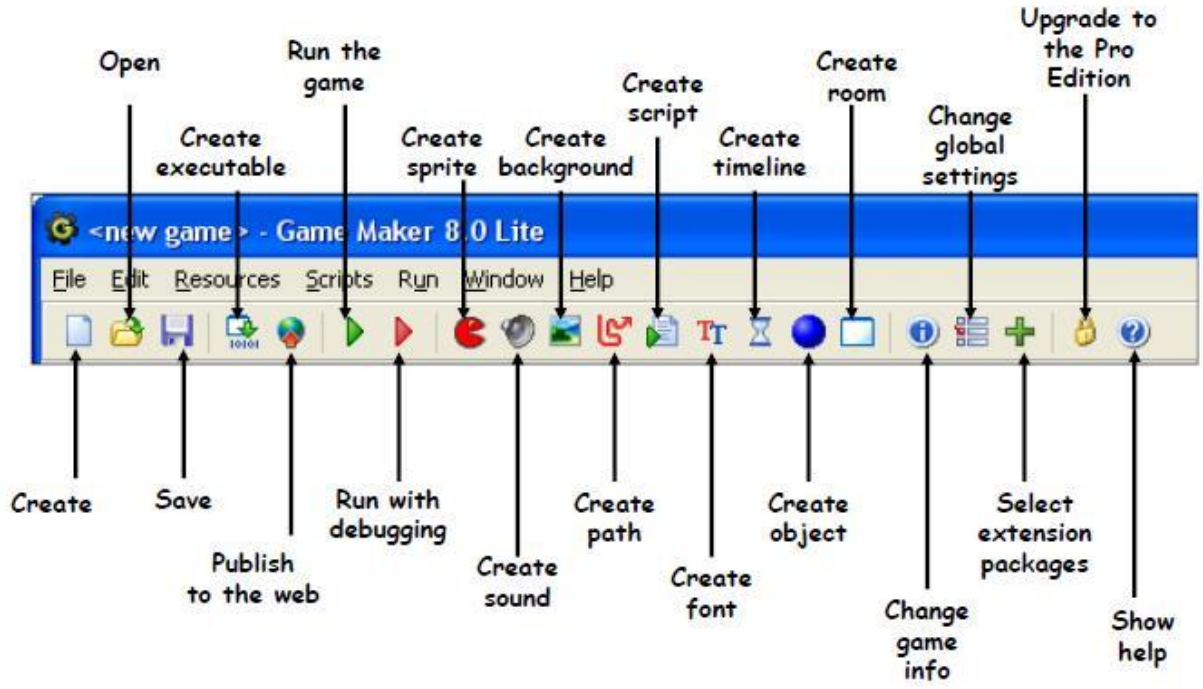
Επίσης μας δίνεται η δυνατότητα να επιλέξουμε το μήκος και το πλάτος που θέλουμε να έχει το κάθε δωμάτιο που έχουμε δημιουργήσει . Στη συνέχεια θέτουμε το Background του δωματίου και στη συνέχεια αρχίζουμε να τοποθετούμε όσα και όποια αντικείμενα θέλουμε, από αυτά φυσικά που έχουμε ήδη δημιουργήσει .

ΥΠΟΚΕΦΑΛΑΙΟ 1.2
ΔΗΜΙΟΥΡΓΙΑ ΠΑΙΧΝΙΔΙΩΝ / EVENTS ΚΑΙ ACTIONS

Αφού δημιουργήσουμε κάποιο δωμάτιο (room), και αφού βάλουμε και κάποιο φόντο (background), αρχίζουμε να τοποθετούμε τα αντικείμενα (objects) που θέλουμε με την απλή μέθοδο του Drag And Drop. Έτσι χτίζουμε σιγά σιγά την «πίστα» μας. Θα βάλουμε τα ανάλογα αντικείμενα, ανάλογα με το παιχνίδι που δημιουργούμε, αφού φυσικά όπως προείπαμε τα έχουμε ταυτίσει με κάποιο Sprite όταν αυτό χρειάζεται. Συνήθως βάζουμε κάποιους τοίχους, κάποια αντικείμενα, κάποιον ή κάποιους χαρακτήρες, τα οποία στην πορεία θα έχουν και την ανάλογη αλληλεπίδραση μεταξύ τους που θέλουμε. Πολλές φορές χρειάζεται να τοποθετήσουμε και κάποια κείμενα όπως τα score ή τις ζωές, τα οποία αλλάζουν τιμές κατά τη διάρκεια του παιχνιδιού ανάλογα με τις παραμέτρους που θα ορίσουμε στη συνέχεια. Και αυτό μπορεί κάποιες φορές να επηρεάζει την πορεία του παιχνιδιού. Για να γίνει αυτό εφικτό, θα πρέπει να ορίσουμε και κάποιες «αόρατες» μεταβλητές κατά την εκκίνηση του παιχνιδιού, και οι οποίες θα αλλάζουν τιμές ανάλογα με τις διάφορες αλληλεπιδράσεις που θα υπάρξουν στο παιχνίδι, και για τις οποίες θα μιλήσουμε στην συνέχεια. Πολλές φορές υπάρχουν αντικείμενα (objects) τα οποία τοποθετούμε, αλλά είναι αόρατα, δηλαδή δεν φαίνονται κατά την διάρκεια που κάποιος παίζει το παιχνίδι. Για παράδειγμα, αυτά μπορεί να είναι κάποιες νοητές γραμμές, κάποιες μεταβλητές συνιστώσες και φυσικά μην ξεχνάμε πως μπορεί να είναι και κάποιοι ήχοι, μουσικές υποκρούσεις που θέλουμε να εκτελούνται, αλλά φυσικά χωρίς να φαίνονται.

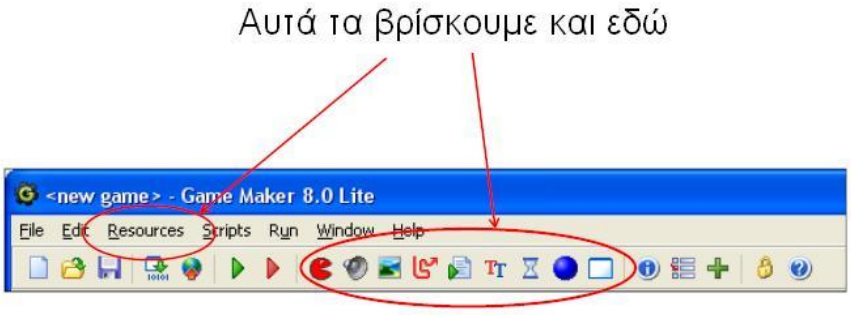
Ας ξεκινήσουμε λοιπόν από τα πολύ βασικά πράγματα που πρέπει να γνωρίζει κάποιος ώστε να ξεκινήσει την υλοποίηση ενός παιχνιδιού στο Game Maker. Έστω ότι έχουμε στήσει το Room στο οποίο θέλουμε να δουλέψουμε, με όλα αυτά που προείπαμε παραπάνω. Κι έτσι ξεκινάμε να δημιουργούμε τις διάφορες αλληλεπιδράσεις μεταξύ τους. Να τονίσουμε πως το Game Maker μας δίνει την δυνατότητα είτε του προγραμματισμού των εντολών που θέλουμε να δώσουμε με την εξ’ αρχής γραφή κώδικα στη γλώσσα του Game Maker, για την οποία θα μιλήσουμε στην συνέχεια, είτε με την χρήση έτοιμων στην ουσία εντολών, με την μέθοδο του Drag And Drop, χάρη στο γραφικό περιβάλλον που μας δίνει η εφαρμογή.

Αυτή τη στιγμή λοιπόν θα ασχοληθούμε με την δεύτερη μέθοδο. Θα εξηγήσουμε αρχικά το Toolbar που υπάρχει σε οριζόντια θέση, στην κορυφή της εφαρμογής.



Εικόνα 1.14

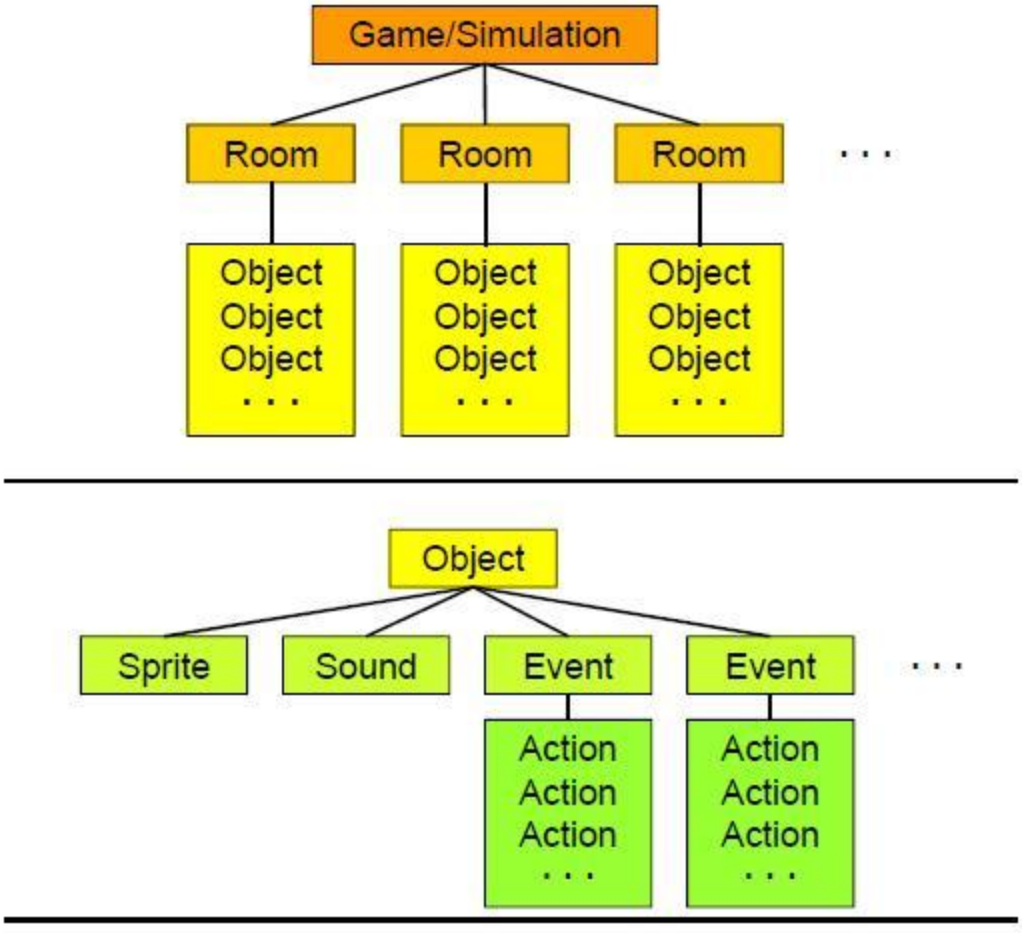
Όπως και στα περισσότερα προγράμματα, έτσι και στο Game Maker, το Toolbar αυτό είναι καθαρά βοηθητικό, ώστε να μπορεί ο χρήστης να κάνει αυτό που λέμε «πιο εύκολη τη ζωή του». Όλα τα στοιχεία της βρίσκονται και στα μενού των Drop Down λιστών του προγράμματος, που βρίσκονται ακριβώς από πάνω. Στην ουσία, υπάρχουν τρεις με τέσσερις τρόποι για την εισαγωγή παραδείγματος χάρη ενός Sprite, όπως επεξηγούμε και στο ανάλογο κεφάλαιο.



Εικόνα 1.15

Και τώρα θα εξηγήσουμε τη βασική δομή με την οποία γίνεται η υλοποίηση ενός παιχνιδιού στη εφαρμογή αυτή. Όπως είπαμε παραπάνω, μεταξύ των οντοτήτων που έχουμε εισάγει σε ένα δωμάτιο, υπάρχουν κάποιες αλληλεπιδράσεις, τις οποίες αποκαλούμε **Events**. Για παράδειγμα αλληλεπίδραση υπάρχει ανάμεσα σε έναν χαρακτήρα και σε ένα αντικείμενο, όταν αυτά έρχονται σε επαφή. Όταν λοιπόν έχουμε κάποιο Event, τότε γίνονται και οι ανάλογες αντιδράσεις όπως εμείς θα τις ορίσουμε, για παράδειγμα αύξηση του score, να μην μπορεί να προχωρήσει ο χαρακτήρας και τα λοιπά. Οι αντιδράσεις αυτές καλούνται στο Game Maker, **Actions**. Actions μπορούμε να δηλώσουμε κατά την εκκίνηση του παιχνιδιού, κατά την είσοδο σε ένα Room και τα λοιπά, όπου όπως είναι κατανοητό, και αυτά «βαπτίζονται» και θεωρούνται ως Events.

Η γενική δομή λοιπόν της υλοποίησης ενός παιχνιδιού στην πλατφόρμα μας παρουσιάζεται κάπως έτσι



Εικόνα 1.16

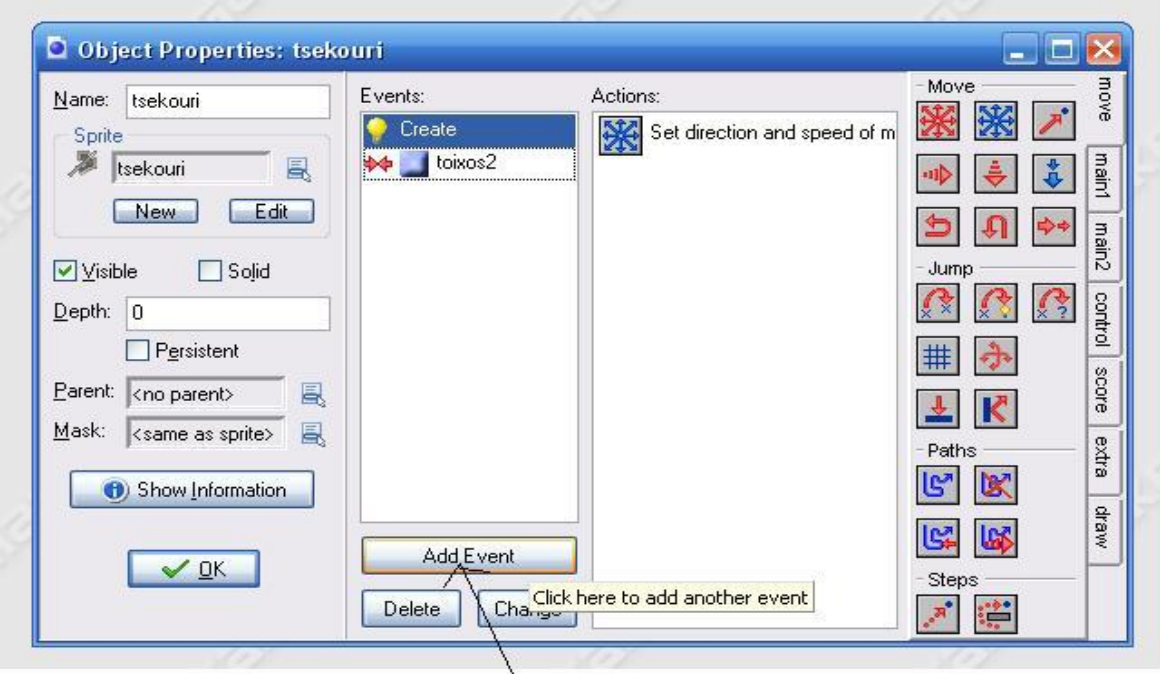
Πιο πριν αναφέραμε πως για την υλοποίηση και την δημιουργία ενός παιχνιδιού στο Game Maker, πρέπει να ακολουθήσουμε μια συγκεκριμένη σειρά από ενέργειες που πρέπει να κάνουμε. Τα βήματα λοιπόν που ακολουθούμε βασίζονται κυρίως στην εξής σειρά με δυνατότητα κάποιων αλλαγών στη σειρά εκτέλεσής τους, όπου αυτό φυσικά μπορεί να γίνει :

1. Περιγραφή του παιχνιδιού που σχεδιάζουμε να υλοποιήσουμε. Τι είναι αυτό που θέλουμε να δημιουργήσουμε, με τι έχει σχέση και πως το φανταζόμαστε
2. Ορίζουμε τα Sprites του παιχνιδιού μας
3. Ορίζουμε τους ήχους (Sounds) που θα εισάγουμε κατά το «τρέξιμο» του παιχνιδιού
4. Ορίζουμε όλα τα Objects, αλλά όχι ακόμη τα Events και τα Actions που θα λάβουν χώρα στην εκτέλεση του παιχνιδιού
5. Στη συνέχεια λοιπόν καθορίζουμε τα Events και τα Actions του κάθε αντικειμένου
6. Ορίζουμε και δημιουργούμε το δωμάτιο ή τα δωμάτια που θα χρησιμοποιήσουμε για τις πίστες του παιχνιδιού
7. Τοποθετούμε τα διάφορα αντικείμενα που έχουμε ήδη δημιουργήσει, στα δωμάτια.

Εδώ να σημειώσουμε πως το κάθε αντικείμενο μπορεί να εισαχθεί περισσότερες από μία φορές στο παιχνίδι, στο ίδιο δωμάτιο ή όχι. Απλά να τονίσουμε πως τα Actions που θα ορίσουμε θα εκτελούνται για όλα να «ίδια» αντικείμενα. Εάν δεν θέλουμε να συμβαίνει κάτι τέτοιο, αυτό που μπορούμε να κάνουμε είναι να δημιουργήσουμε διαφορετικά αντικείμενα, με διαφορετική ονομασία φυσικά, αντιστοιχώντας τα όμως με το ίδιο Sprite, ώστε να έχουν την ίδια μορφή (για παράδειγμα, ο ίδιος χαρακτήρας, σε διαφορετικά δωμάτια, να έχει διαφορετικές αλληλεπιδράσεις με παρόμοια αντικείμενα).

Εννοείται πως εάν κατά τη διάρκεια της δημιουργίας μας ή μετά το πέρας της θέλουμε να προσθέσουμε και κάτι άλλο ή να τροποποιήσουμε κάτι, για παράδειγμα ένα αντικείμενο, φυσικά και δεν μας το απαγορεύει κανείς, μπορούμε.

Επομένως ας ξεκινήσουμε με τη δημιουργία των Events. Όταν ανοίξουμε την καρτέλα επιλογών ενός αντικειμένου κάνοντας διπλό κλικ πάνω του, επιλέγουμε το **Add Events** .



Εικόνα 1.17

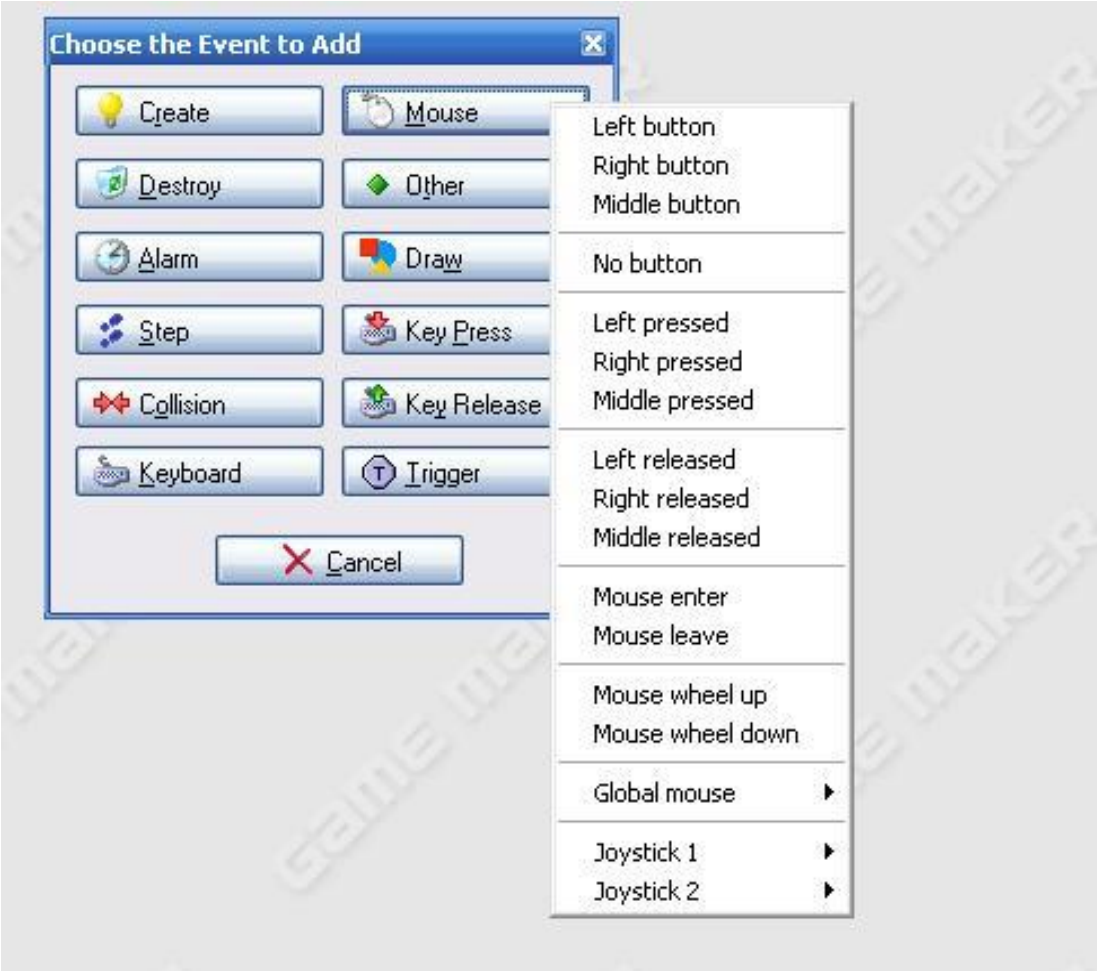
Και στη συνέχεια μας εμφανίζεται η καρτέλα με τα Events που μπορούμε να επιλέξουμε. Κάθε μία από αυτές τις επιλογές, όταν κάνουμε κλικ πάνω τους μας δίνει και μια λίστα από άλλες, ανάλογες επιλογές που μπορούμε να κάνουμε.

Εικόνα 1.18

ΥΠΟΚΕΦΑΛΑΙΟ 1.2.1
EVENTS

Create : Με αυτό το Event μας δίνεται η δυνατότητα να επιλέξουμε κάποια Actions, τα οποία θα λάβουν χώρα κατά τη δημιουργία του αντικειμένου, δηλαδή στην αρχή του παιχνιδιού ή στην είσοδο της συγκεκριμένης πίστας ή γενικά τη χρονική στιγμή που θα δημιουργηθεί και θα εμφανιστεί το αντικείμενο στο παιχνίδι.

Mouse : Με αυτό το Event μας δίνεται η δυνατότητα να επιλέξουμε κάποια Actions, να οποία θα πραγματοποιηθούν ανάλογα με την επιλογή ή τις επιλογές που θα κάνουμε, από τις διάφορες ενέργειες που μπορούν να γίνουν από το ποντίκι (mouse) του υπολογιστή μας, κατά τη διάρκεια του παιχνιδιού. Οι επιλογές αυτές, οι οποίες αποτελούν διαφορετικά μεμονωμένα Events, φαίνονται παρακάτω



Εικόνα 1.19

- **Left button :** Όταν κάνουμε αριστερό κλικ
- **Right button :** Όταν κάνουμε δεξί κλικ
- **Middle button :** Όταν κάνουμε κλικ με το μεσαίο κουμπί του ποντικιού
- **No button :** Όταν δεν πατήσουμε κανένα κουμπί του ποντικιού
- **Left Pressed :** Όταν πατήσουμε το αριστερό κουμπί, οι ενέργειες που θα γίνουν πριν το αφήσουμε
- **Right pressed :** Όταν πατήσουμε το δεξί κουμπί, οι ενέργειες που θα γίνουν πριν το αφήσουμε
- **Middle pressed :** Όταν πατήσουμε το μεσαίο κουμπί, οι ενέργειες που θα γίνουν πριν το αφήσουμε
- **Left released :** Όταν αφήσουμε το αριστερό κουμπί του ποντικιού, που έχουμε ήδη κάνει κλικ
- **Right released :** Όταν αφήσουμε το δεξί κουμπί του ποντικιού, που έχουμε ήδη κάνει κλικ
- **Middle released :** Όταν αφήσουμε το μεσαίο κουμπί του ποντικιού, που έχουμε ήδη κάνει κλικ
- **Mouse enter :** Όταν το ποντίκι μας περάσει και βρίσκεται πάνω από το αντικείμενό μας
- **Mouse leave :** Όταν το ποντίκι φύγει από πάνω από το αντικείμενό μας
- **Mouse wheel up :** Όταν κάνουμε scroll την ρόδα του ποντικιού μας προς τα πάνω
- **Mouse wheel down :** Όταν κάνουμε scroll την ρόδα του ποντικιού μας προς τα κάτω
- **Global mouse :** Ενέργειες που θα κάνουμε εάν χρησιμοποιούμε κάποιο σφαιρικό ποντίκι. Γενικά, όταν πατάμε κάποια πλήκτρα του ποντικιού, χωρίς να μας ενδιαφέρει η θέση του στην οθόνη
- **Joystick 1 και Joystick 2 :** Ενέργειες π[ου θα κάνουμε εάν χρησιμοποιούμε κάποιο ή κάποια χειριστήρια

Destroy : Εδώ επιλέγουμε τις ενέργειες που θα γίνουν, όταν καταστραφεί το αντικείμενό μας κατά την διάρκεια του παιχνιδιού. Η καταστροφή ενός αντικειμένου αποτελεί ένα από τα προκατασκευασμένα Actions που μας δίνει η εφαρμογή, το οποίο και θα συναντήσουμε στην συνέχεια

Alarm : Με αυτό το Event, ορίζουμε τις ενέργειες που θέλουμε να γίνουν (Actions), κάποια συγκεκριμένη χρονική στιγμή του παιχνιδιού. Δηλαδή, για παράδειγμα, μπορούμε να ορίσουμε ότι σε δέκα λεπτά (10') αφού ξεκινήσει το παιχνίδι, κάνε αυτό και αυτό. Επίσης, στον προγραμματισμό έχουμε μάθει να χρησιμοποιούμε κάποιες μεταβλητές οι οποίες ανάλογα με κάποιες ενέργειες που γίνονται, αλλάζουν τιμή. Έτσι, με το Alarm Event, ορίζουμε και τα Actions που θα λάβουν χώρα όταν μια μεταβλητή πάρει κάποια συγκεκριμένη τιμή. Για παράδειγμα όταν η μεταβλητή που έχουμε ορίσει για τις ζωές του χαρακτήρα μας πάρει την τιμή 0 , τότε να εμφανίζεται το Game Over και να τελειώνει το παιχνίδι εκεί.

Draw : Στο παιχνίδι, μπορεί κάποια στιγμή να ορίσουμε μια ενέργεια η οποία θα εμφανίζει κάτι καινούριο στο δωμάτιό μας, για παράδειγμα μια εικόνα, ή θα αλλάζει στο σκορ, τις ζωές π[ου μας έχουν απομείνει και τα λοιπά. Έτσι με αυτό το Event, αφού γίνει κάτι από τα παραπάνω που είπαμε, τότε ζητάμε την εκτέλεση κάποιων Actions.

Step : Σε πολλά παιχνίδια, όλοι μας έχουμε συναντήσει κάποιον παίχτη, συνήθως κάποιο ανθρωπάκι, το οποίο μπορούμε να ελέγξουμε εμείς, συνήθως με το πληκτρολόγιο ή με κάποιο Joystick. Επομένως κινούμε τον χαρακτήρα αυτόν, με κάποια βήματα, δεξιά αριστερά και τα λοιπά. Μπορούμε λοιπόν να ορίσουμε κάποια Actions, τα οποία θα συμβούν όταν κινούμε αυτό το Object, όταν ξεκινάει την κίνησή του ή όταν την τελειώνει.

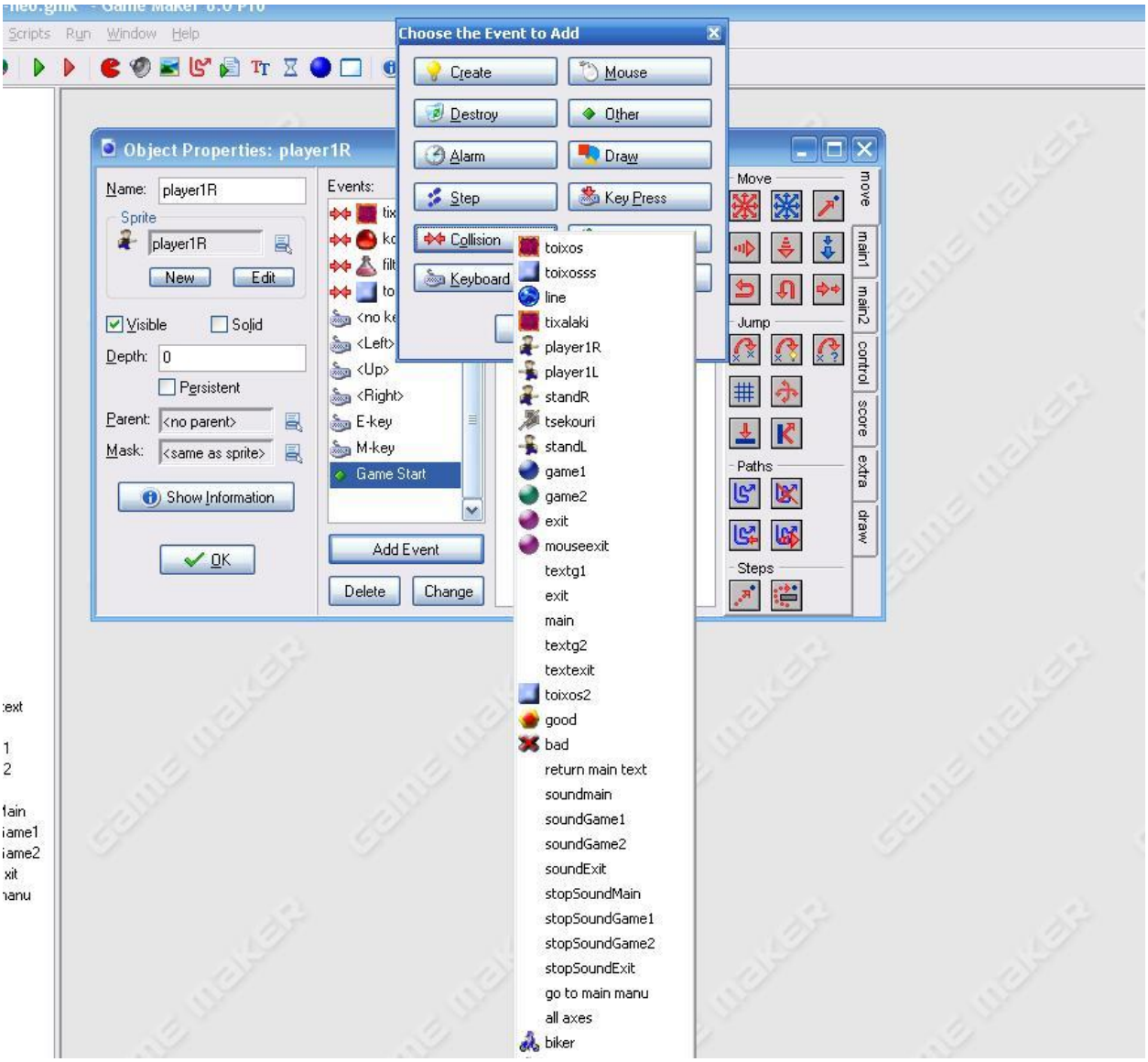
Εικόνα 1.20

Key Press : Φυσικά, σχεδόν σε όλα τα παιχνίδια, υπάρχουν Actions τα οποία πραγματοποιούνται όταν πατήσουμε κάποιο ή κάποια πλήκτρα του πληκτρολογίου. Δηλαδή, μπορεί πατώντας κάποιο κουμπί να γίνεται κάτι στο παιχνίδι που να αλλάζει ή όχι την τροπή του, αλλά μπορεί φυσικά να γίνονται και άλλου είδους ενέργειες, όπως η επιστροφή στο αρχικό μενού, η προσωρινή παύση του παιχνιδιού (pause), η αποθήκευση (save) και τα λοιπά. Όπως είναι κατανοητό, όλα αυτά πρέπει να τα ορίσουμε εμείς.

Εικόνα 1.21

Μας δίνεται η δυνατότητα να επιλέξουμε το πλήκτρο που θέλουμε να γίνουν οι ανάλογες ενέργειες με το που το πατήσουμε, από όλα τα διαθέσιμα πλήκτρα του πληκτρολογίου μας. Από τα γράμματα, τους αριθμούς και τα λοιπά, όπως βλέπουμε και στην εικόνα. Η πιο κλασική περίπτωση χρήσης αυτού του Event, είναι η χρήση των Arrow Keys, για την καθοδήγηση κάποιου χαρακτήρα ή γενικά κάποιου αντικειμένου.

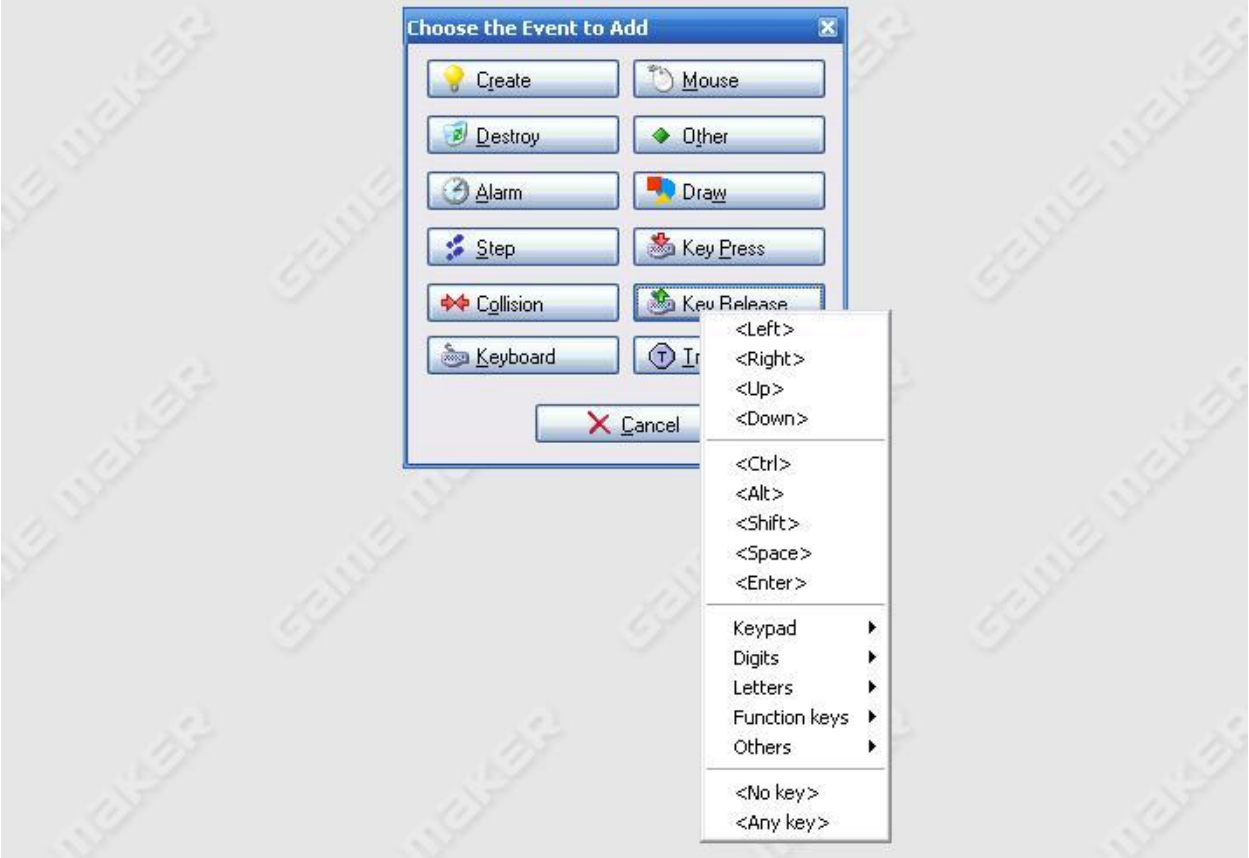
Collision : Με αυτό το Event, δημιουργούμε Actions, στην περίπτωση που το αντικείμενό μας έρθει σε επαφή (σύγκρουση) με κάποιο άλλο αντικείμενο του παιχνιδιού μας, που βρίσκεται και αυτό μέσα στην πίστα μας. Για παράδειγμα, ένας παίχτης, όταν έρθει σε επαφή με κάποιο συγκεκριμένο αντικείμενο, χάνει μία ζωή. Ή, το πιο βασικό παράδειγμα που πρέπει να δώσουμε, είναι αυτό που όταν ο χαρακτήρας μας έρθει σε επαφή με κάποιον τοίχο. Αυτό που θα γίνει, είναι να σταματήσει ο χαρακτήρας και να μην μπορεί να προχωρήσει κι άλλο προς αυτήν την κατεύθυνση. Με το πατήσουμε το κουμπί «Collision» από το μενού των Events, μας ανοίγει μια λίστα με όλα τα αντικείμενα που έχουμε ήδη δημιουργήσει, ώστε να επιλέξουμε αυτό που θέλουμε.



Εικόνα 1.21

Εδώ βλέπουμε ένα παράδειγμα με τη λίστα που ανοίγει, στο παιχνίδι που έχουμε φτιάξει .

Key Release : Με αυτό το Event, δηλώνουμε τα Actions που θέλουμε να συμβούν όταν, αφού έχουμε πατήσει κάποιο κουμπί του πληκτρολογίου ή το έχουμε πατημένο, το αφήσουμε. Για παράδειγμα, μπορεί να χειριζόμαστε έναν χαρακτήρα, κινώντας τον προς τα δεξιά, έχοντας πατημένο το ανάλογο Arrow Key. Με το που σταματήσουμε να έχουμε πατημένο αυτό το κουμπί, θέλουμε να γίνεται κάποια συγκεκριμένη ενέργεια, για παράδειγμα ο χαρακτήρας να αλλάζει μορφή. Αυτό θα το πραγματοποιήσουμε, θέτοντας αυτό το Event. Όπως και με το Event «Key Press», έτσι κι εδώ υπάρχει το ανάλογο μενού.

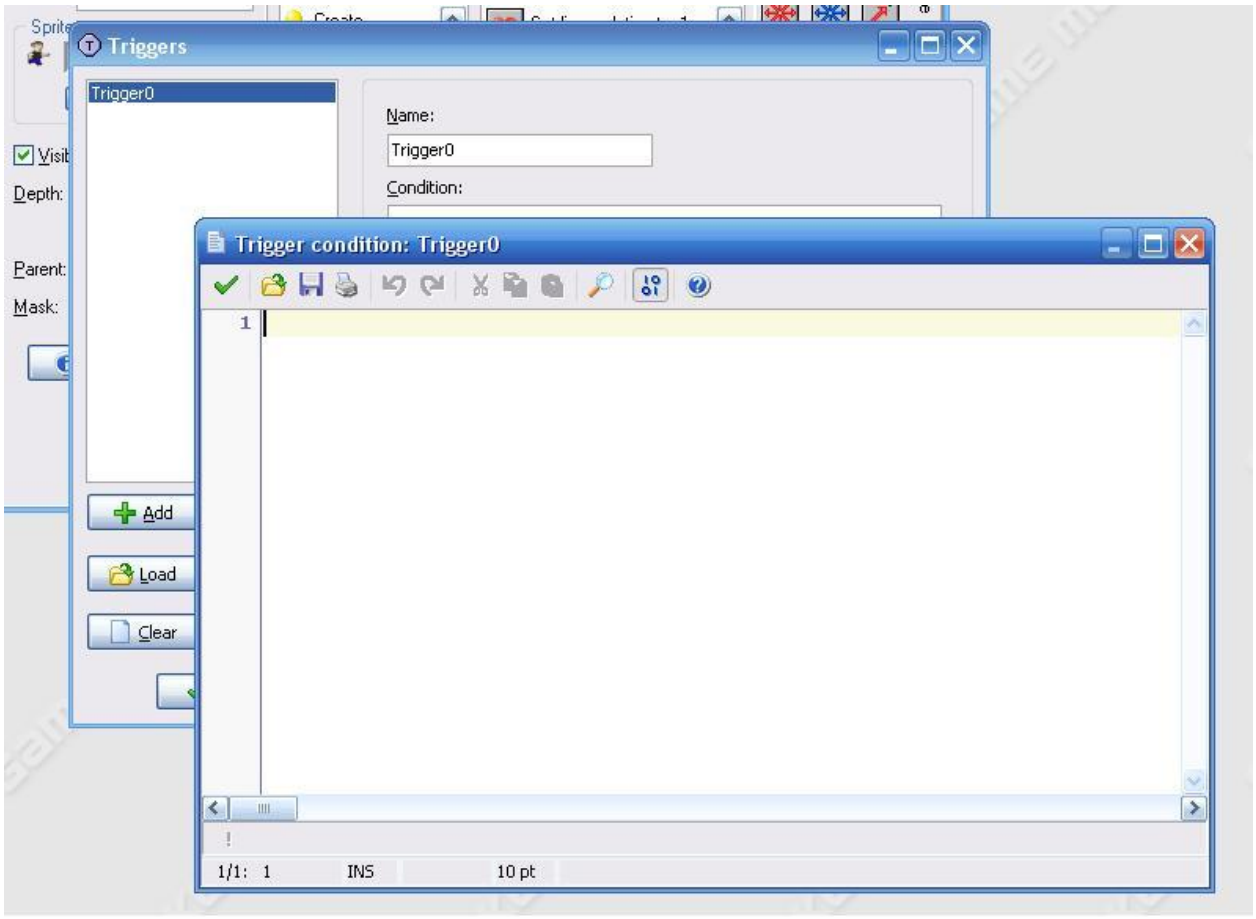


Εικόνα 1.22

Keyboard : Με αυτό το Event, επιλέγουμε τα Actions που θα ενεργοποιηθούν, όταν πατήσουμε κάποιο κουμπί του πληκτρολογίου, γενικά. Η διαφορά του με το Key Press Event, είναι ότι στο δεύτερο, ενεργοποιούνται κάποιες ενέργειες, όταν απλά πατήσουμε μία φορά κάποιο πλήκτρο, και μόνο για τη στιγμή που το πατάμε. Όπως είναι λογικό, η λίστα με τις επιλογές που μας δίνει είναι η ίδια ακριβώς, όπως φαίνεται παρακάτω.

Εικόνα 1.23

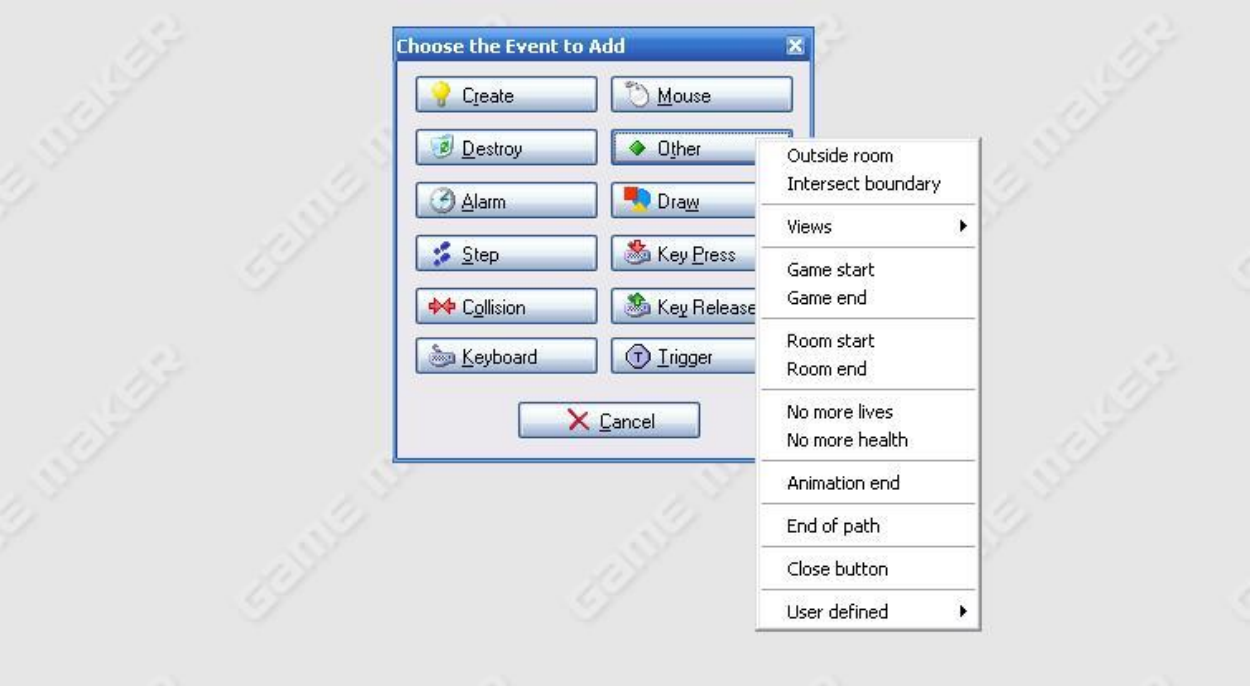
Trigger : Εάν επιλέξουμε το Trigger, στην ουσία δημιουργούμε μόνοι μας κάποιο δικό μας Event, με τη χρήση κώδικα σε γλώσσα GML (Game Maker Language) . Όπως έχουμε αναφέρει ξανά, μπορεί το Game Maker να μας δίνει ένα εύκολο εικονικό μενού επιλογών, φυσικά όμως μπορούμε και να δημιουργήσουμε τα πάντα από μόνοι μας. Μην ξεχνάμε πως είναι ένα προγραμματιστικό εργαλείο που απλά μας παρέχει στην ουσία κάποια έτοιμα κομμάτια κώδικα εικονογραφημένα, απλά για να μας διευκολύνει. Μας παρέχει με αυτόν τον τρόπο τα κομμάτια που ίσως χρησιμοποιήσουμε οι περισσότεροι και περισσότερο. Όχι όμως φυσικά τα πάντα. Και όποιος έχει ασχοληθεί με προγραμματισμό, θα ξέρει, πως όσο πιο πολύ «εισβάλλουμε» στη λεπτομέρεια, τόσο καλύτερο αποτέλεσμα έχουμε. Και φυσικά εδώ η λεπτομέρεια είναι ο κώδικας.



Εικόνα 1.24

Για την δημιουργία του κώδικα, το Game Maker μας παρέχει, όπως προαναφέραμε, έναν απλό και εύχρηστο Code Editor.

Other : Αυτό το Event, μας δίνει μια λίστα από διάφορες επιλογές, τις οποίες θα δούμε πιο κάτω.



Εικόνα 1.25

- **Outside room :** Εδώ δίνουμε Actions για το αντικείμενό μας, όταν αυτό βρίσκεται εκτός του δωματίου. Πιθανόν εδώ να δίνουμε τα Actions για την καταστροφή του (Destroy).

- **Intersect boundary :** Εδώ δίνουμε Actions όταν το αντικείμενό μας φτάσει στα «σύνορα», δηλαδή στο τέλος του δωματίου.
- **Views :** Σε περίπτωση που στο παιχνίδι μας χρησιμοποιούμε κάποια Views, κάποιες οπτικές γωνίες δηλαδή, μπορούμε να ορίσουμε κάποια Actions, που θα πραγματοποιηθούν όταν το αντικείμενο, για το οποίο ορίζουμε αυτό το Event, βγει εκτός κάποιου View, φτάσει στα σύνορά του, ή εισέλθει σε αυτό.
- **Game start :** Εδώ ορίζουμε κάποια Actions που θα συμβούν μόλις το παιχνίδι ξεκινήσει. Συνήθως ορίζουμε κάποιες αρχικές τιμές για τις μεταβλητές μας όπως μεταβλητές για το χρόνο και το Score, αρχικοποιούμε κάποιες θέσεις αντικειμένων, και τα λοιπά.
- **Game end :** Εδώ ορίζουμε τα Actions που θα συμβούν μόλις το παιχνίδι μας φτάσει στο τέλος του. Συνηθισμένο και κλασσικό παράδειγμα της χρήσης αυτού του Event, είναι η αποθήκευση του Score που έχουμε κάνει σε κάποιο αρχείο, σε κάποια βάση δεδομένων ας πούμε.
- **Room start :** Εδώ ορίζουμε τα Actions που θα θέσουμε, όταν ξεκινάει κάποια συγκεκριμένη πίστα, κάποιο Room, και όχι όταν ξεκινάει γενικά το παιχνίδι.
- **Room end :** Και εδώ, αυτά τα Actions που θέλουμε να ορίσουμε όταν τελειώνει η «δράση» μας σε κάποιο συγκεκριμένο Room.
- **No more lives :** Η εφαρμογή διαθέτει ένα προκατασκευασμένο σύστημα για τις ζωές που έχει ο χαρακτήρας μας μέσα στο παιχνίδι. Αφού έχουμε ορίσει τις

αυξομειώσεις που θα έχει η μεταβλητή για τις ζωές ανάλογα με τα διάφορα Collisions που θα συμβούν, όταν αυτή η μεταβλητή έχει τιμή μικρότερη ή ίση με το μηδέν, τότε με τη βοήθεια αυτού του Event, πραγματοποιούνται τα ανάλογα Actions που θα ορίσουμε. Για παράδειγμα, Restart του παιχνιδιού, ή Game Over.

- **No more health :** Το Game Maker, όπως και για τις ζωές, έχει ένα προκατασκευασμένο σύστημα για το «Health» ενός Object, ενός χαρακτήρα στο παιχνίδι. Υπάρχουν κάποια συγκεκριμένα Actions με τα οποία μπορούμε να ορίσουμε την τιμή του Health κατά τη διάρκεια του παιχνιδιού (την αυξομείωσή της). Όταν λοιπόν το Health φτάσει σε τιμή μικρότερη ή ίση του μηδενός, τότε με τη βοήθεια αυτού του Event, ενεργούμε ανάλογα. Μπορεί λοιπόν να τερματίσουμε εδώ το παιχνίδι, ή απλά να μειώσουμε τις ζωές, όπως επίσης μπορούμε να μεταφέρουμε τον χαρακτήρα μας στην αρχή του παιχνιδιού και γενικά να κάνουμε ότι θέλουμε εμείς.
- **Animation end :** Γνωρίσουμε πως στον ψηφιακό κόσμο και τον κόσμο των γραφικών, μια κίνηση ενός αντικειμένου, στην πράξη είναι η πολλών εικόνων, η μία μετά την άλλη, πάρα πολύ γρήγορα. Και μετά ξανά η πρώτη, και τα λοιπά. Έτσι για παράδειγμα εμείς πιστεύουμε πως ένα αντικείμενο κινείται, ενώ στην πραγματικότητα βλέπουμε πολλές «σταθερές» εικόνες, την μία μετά την άλλη. Αυτό ονομάζεται ANIMATION. Επομένως, αφού τελειώσει η αλληλουχία αυτών των εικόνων, και πριν ξεκινήσει πάλι από την αρχή, σε αυτό το σημείο εμφανίζονται τα Actions αυτού του Event, όπως εμείς τα έχουμε ορίσει.
- **End of path :** Εδώ μπαίνουν τα Actions, στην περίπτωση που το αντικείμενό μας ακολουθεί κάποιο μονοπάτι (Path) και αυτό φτάσει στο τέλος του.
- **Close button :** Εδώ ορίζουμε Actions προς εκτέλεση, στην περίπτωση που ο χρήστης πατήσει το κουμπί «Close», το **X** δηλαδή. Πιθανόν να μην θέλουμε να τερματίζεται απότομα το παιχνίδι, αλλά να γίνονται οι ανάλογες ενέργειες, όπως αποθήκευση, ερώτηση για τερματισμό και τα λοιπά.
- **User defined :** Εδώ ορίζουμε Events και Actions, δικά μας, τα οποία θα συμβούν μόνο όταν τα καλέσουμε από μόνοι μας, με τη χρήση κώδικα.

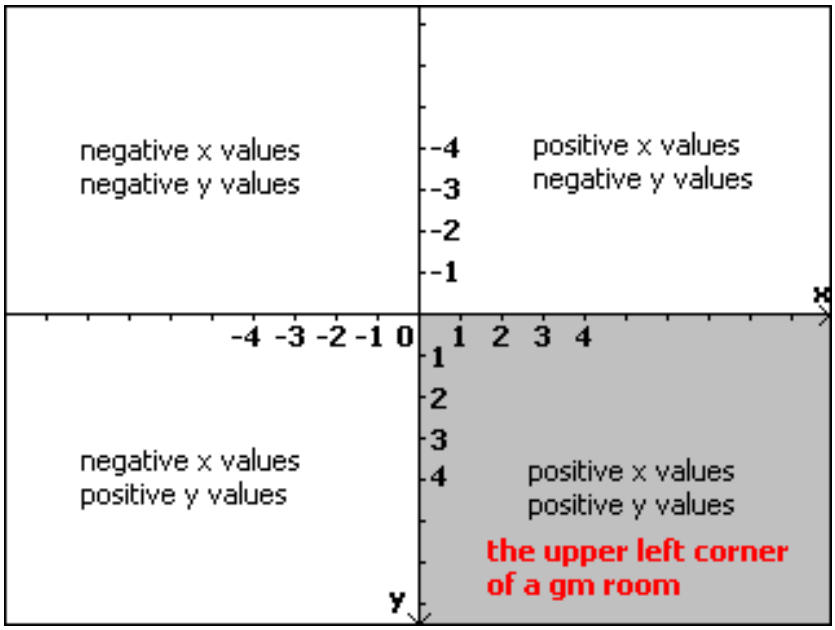
ΥΠΟΚΕΦΑΛΑΙΟ 1.2.2
ACTIONS

Αφού λοιπόν έχουμε ήδη επιλέξει το αντικείμενό μας, και αφού επιλέξαμε και το Event, προχωράμε στο επόμενο βήμα, που είναι η εισαγωγή των Actions.

Τα Actions, είναι χωρισμένα σε κατηγορίες, για λόγους ευκολίας του χρήστη. Οι κατηγορίες αυτές ονομαστικά, είναι οι εξής :

Move, main1, main2, control, score, extra και draw

Αρχικά θα σας δείξω το παρακάτω πινακάκι, το οποίο προσδιορίζει και μας ξεκαθαρίζει τις θέσεις που ορίζουμε να παίρνει το αντικείμενό μας, όταν χρησιμοποιούμε κυρίως τα Move Actions, τα οποία βασίζονται στον γνωστό σε όλους μας άξονα X και Y.

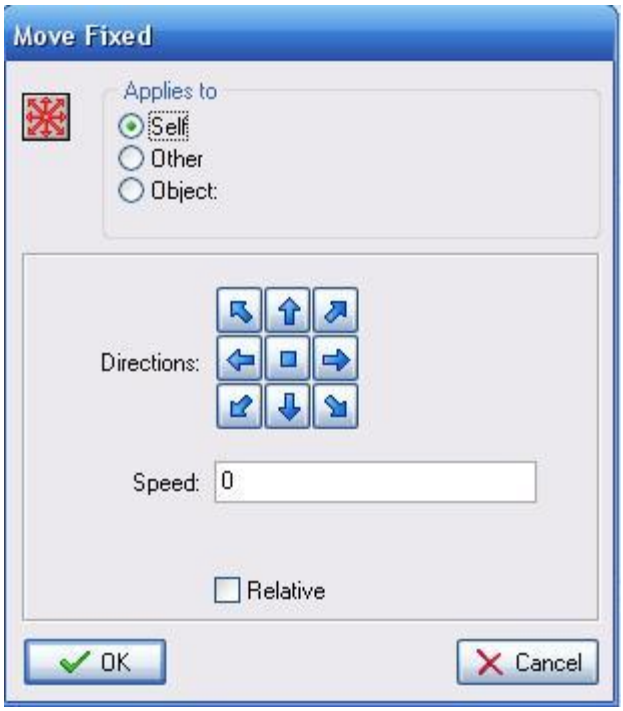


Εικόνα 1.26

Και παρακάτω θα επεξηγήσουμε ένα προς ένα όλα τα Actions που μας παρέχει το Game Maker, αναλυτικά.

ΥΠΟΚΕΦΑΛΑΙΟ 1.2.2.1
MOVE ACTIONS

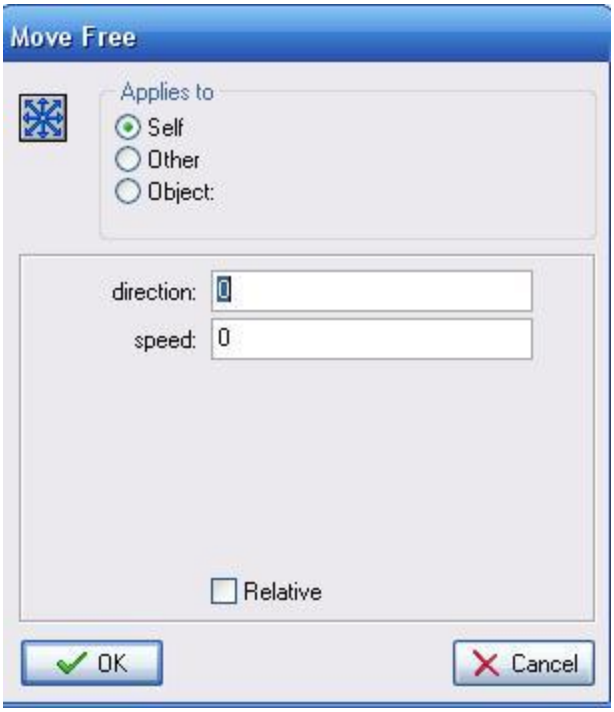
Move Fixed : Με το Action αυτό, καθορίζουμε την κίνηση ενός αντικειμένου, προς μία συγκεκριμένη κατεύθυνση, όπως δεξιά, αριστερά, και τα λοιπά.. Συνήθως το ορίζουμε σε κάποιο από τα Arrow Keys. Επίσης ορίζουμε την ταχύτητα με την οποία θέλουμε να κινηθεί το αντικείμενό μας. Επιλέγουμε εάν αυτό το Action θέλουμε να «λειτουργήσει» στο ίδιο το αντικείμενο πάνω στο οποίο δουλεύουμε, σε κάποιο άλλο, ή στο αντικείμενο με το οποίο έρχεται σε επαφή, εάν το Event που έχουμε επιλέξει είναι το Collision. Να τονίσουμε πως η ταχύτητα η οποία θα ορίσουμε, είναι σε Pixels. Εάν δεν το ορίσουμε, έχει default τιμή το 8. Καλό θα ήταν, να μην βάλουμε αρνητική τιμή στην ταχύτητα, διότι πιθανόν να έχουμε διαφορετικά αποτελέσματα από αυτά που θέλουμε.



Εικόνα 1.27

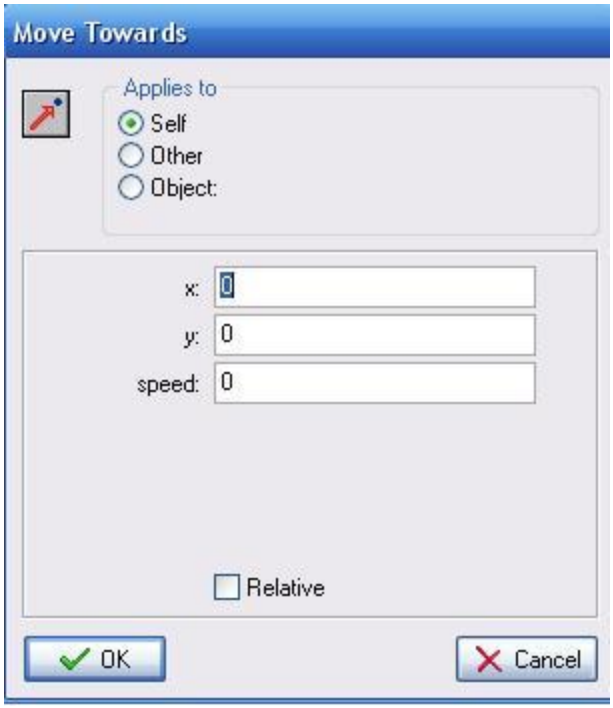
ΠΡΟΣΟΧΗ : Σε πολλά Actions θα παρατηρήσετε ένα Check Box, με την ονομασία **Relative**. Εάν το τσεκάρουμε, τότε η αριθμητική τιμή που δίνουμε παραπάνω, θα προστίθεται κάθε φορά στην προηγούμενη. Δηλαδή, εάν για παράδειγμα δώσουμε την τιμή 5 στην ταχύτητα, πατώντας το ανάλογο πλήκτρο, το αντικείμενό μας στην αρχή θα έχει ταχύτητα 5 pixels, στην συνέχεια 10, μετά 15 και ούτω καθεξής. Εάν δεν το τσεκάρουμε, θα έχει μονίμως την τιμή 5. Αυτό ισχύει σε όλα τα Actions στα οποία συναντάμε αυτήν την επιλογή.

Move Free : Με αυτό το Action, δίνουμε κατεύθυνση στο αντικείμενό μας, με την βοήθεια των μοιρών. Δηλαδή όχι κάποια προτετελεσμενή πορεία όπως δεξιά, αριστερά και τα λοιπά. Στην ουσία, του δίνουμε μια γωνία, μεταξύ 0 και 360 μοιρών. Εάν δώσουμε 0, τότε κατευθύνεται ευθύγραμμα προς τα δεξιά. Οι 90 μοίρες, του δίνουν κάθετη ανοδική πορεία. Εάν θέλουμε να πάρει μια τυχαία πορεία, τότε πληκτρολογούμε στο πλαίσιο : random(360) . Αυτή η εντολή, επιλέγει έναν τυχαίο αριθμό, μικρότερο ή ίσο του αριθμού που δίνουμε, που σε αυτήν την περίπτωση είναι το 360.



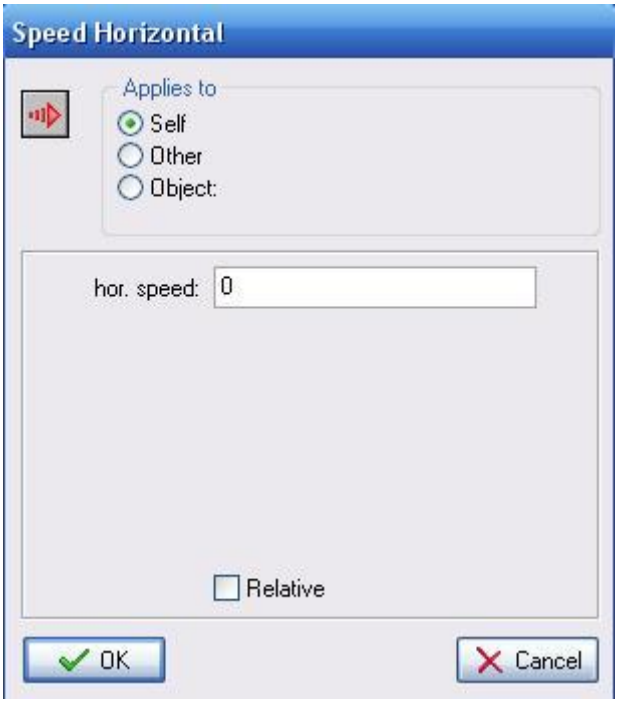
Εικόνα 1.28

Move Towards : Εδώ ορίσουμε μια συγκεκριμένη θέση του δωματίου προς την οποία θέλουμε να κατευθυνθεί το αντικείμενό μας. Αυτό το ορίζουμε σύμφωνα με το καρτεσιανό σύστημα των X και Y. Επίσης, εάν για παράδειγμα θέλουμε το αντικείμενό μας να πάει σε μια θέση, όπου βρίσκεται κάποιο άλλο αντικείμενο, του οποίου την θέση δεν γνωρίζουμε με αριθμούς, τότε αντί νούμερα μπορούμε να βάλουμε το όνομα του αντικειμένου , τελεία x και τελεία y. Για παράδειγμα : kakos.x και kakos.y . Και φυσικά και πάλι θα ορίσουμε την ταχύτητα την οποία θέλουμε να έχει καθώς κατευθύνεται προς το σημείο που θέλουμε, και το Relative εάν το επιθυμούμε.



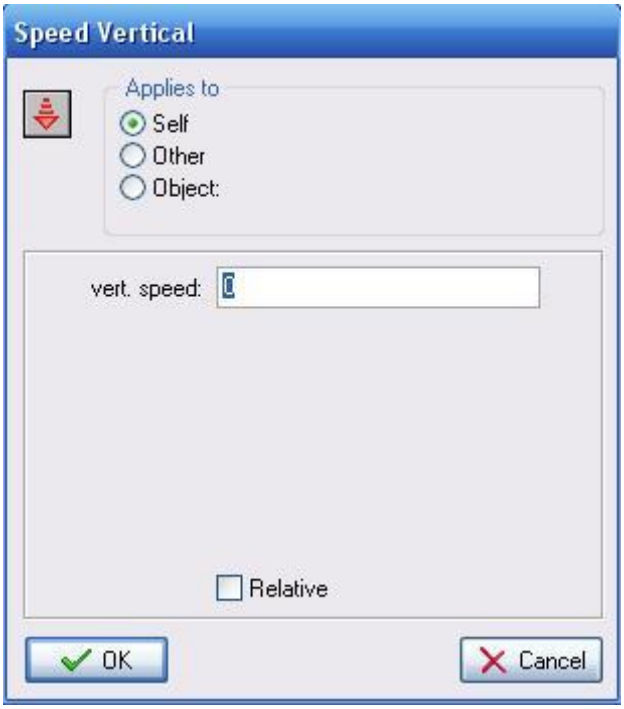
Εικόνα 1.29

Speed Horizontal : Με αυτό το Action, μας δίνεται η δυνατότητα να αλλάξουμε την οριζόντια ταχύτητα του αντικειμένου. Εάν αφήσουμε την τιμή στο 0, τότε η ταχύτητα δεν αλλάζει. Εάν βάλουμε θετικό αριθμό, τότε το αντικείμενο παίρνει ταχύτητα προς τα δεξιά. Εάν βάλουμε αρνητικό, τότε, παίρνει ταχύτητα αντίθετα, δηλαδή προς τα αριστερά. Και στις δύο περιπτώσεις, παίρνει ταχύτητα όσο λέει η τιμή που θα δώσουμε. Εάν τσεκάρουμε το Relative, η ταχύτητα θα αυξηθεί ή θα μειωθεί, όσο ήταν συν ή πλην όσο έχουμε δώσει στη τιμή.



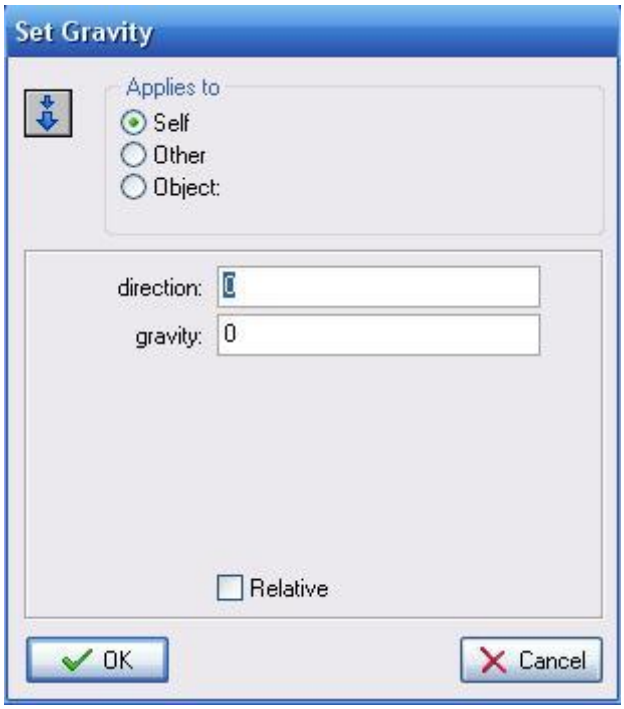
Εικόνα 1.30

Speed Vertical : Το ίδιο ακριβώς με το Speed Horizontal, μόνο που εδώ έχουμε να κάνουμε με την κάθετη ταχύτητα. Και επομένως, αντί για δεξιά και αριστερά, έχουμε να κάνουμε με την ταχύτητα προς τα πάνω και προς τα κάτω.



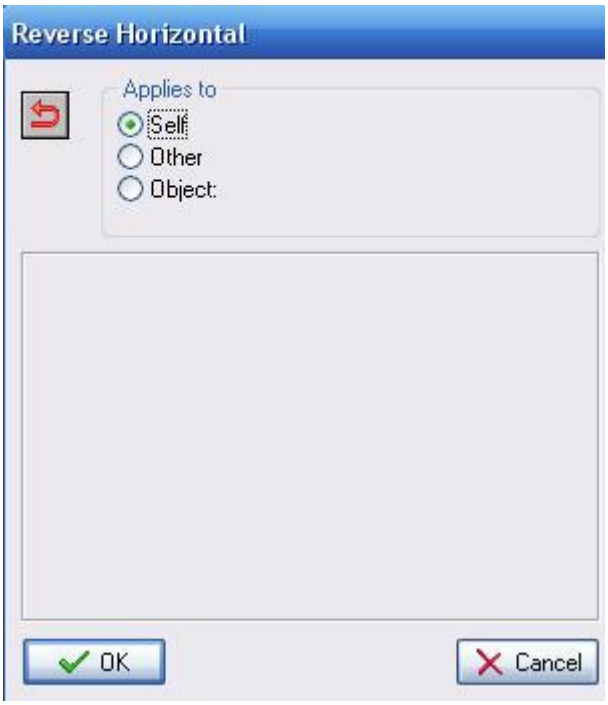
Εικόνα 1.31

Set Gravity : Με αυτό το Action,ορίζουμε την βαρύτητα που θέλουμε να έχει το αντικείμενό μας. Δηλαδή, εάν για παράδειγμα θέσουμε ένα πλήκτρο ώστε το αντικείμενό μας να κάνει κάποιο άλμα, θέλουμε στην συνέχεια να αποκτήσει ταχύτητα προς τα κάτω, ώστε να προσγειώνεται ξανά. Σε αυτήν την περίπτωση θέτουμε το direction σε 270 και στο gravity την ταχύτητα με την οποία θέλουμε να κινείται προς τα κάτω, μετά το άλμα. Να προσέξουμε πως κάθε αντικείμενο, ανάλογα με το είδος και το μέγεθός του, θα πρέπει να έχει διαφορετικό gravity.



Εικόνα 1.32

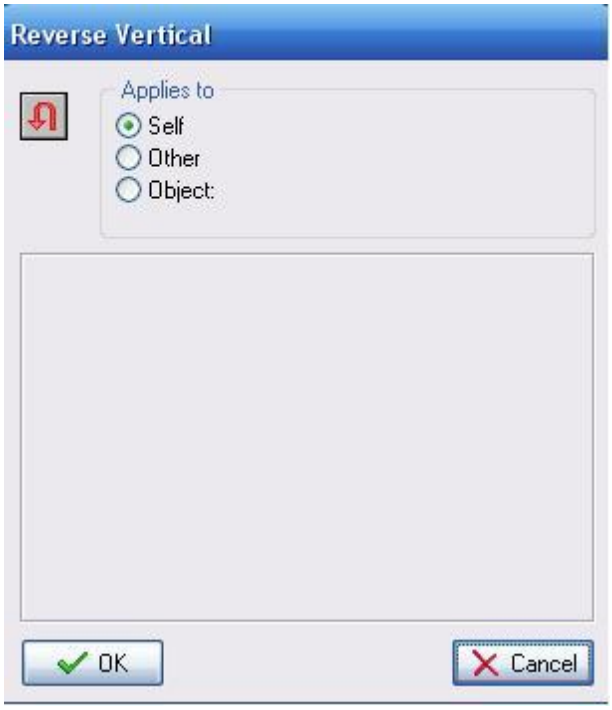
Reverse Horizontal : Με αυτό το Action, αντιστρέφουμε την οριζόντια κίνηση του αντικειμένου μας. Αν για παράδειγμα το αντικείμενο είχε κίνηση προς τα δεξιά, τότε αυτομάτως αρχίζει και κατευθύνεται προς τα αριστερά.



Εικόνα 1.33

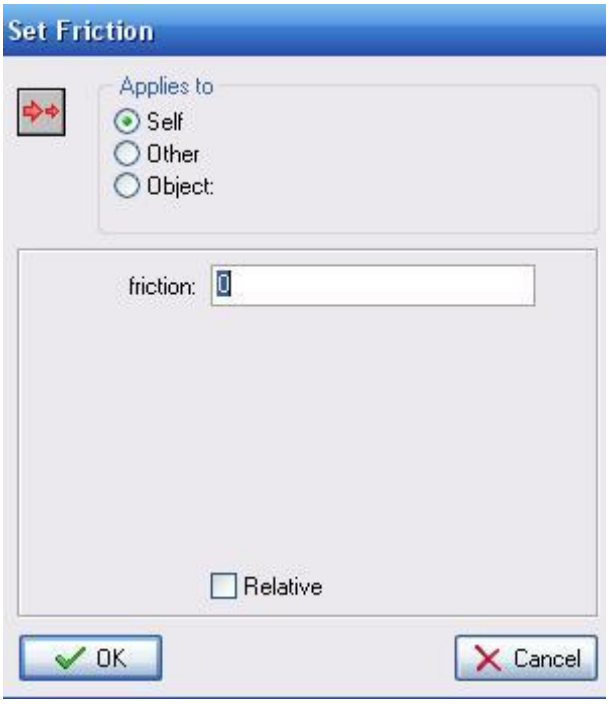
Εδώ δεν υπάρχουν κάποιοι παράμετροι να ορίσουμε, πέρα από το για ποιοι αντικείμενο θέλουμε να ενεργοποιηθεί το Action αυτό. Η ταχύτητα είναι αντιστρόφως ανάλογη με αυτήν που είχαμε ήδη θέσει με το ανάλογο Action.

Reverse Vertical : Το ίδιο ακριβώς όπως και με το Reverse Horizontal, μόνο που σε αυτήν την περίπτωση αλλάζει κατεύθυνση η κάθετη πορεία του αντικειμένου.



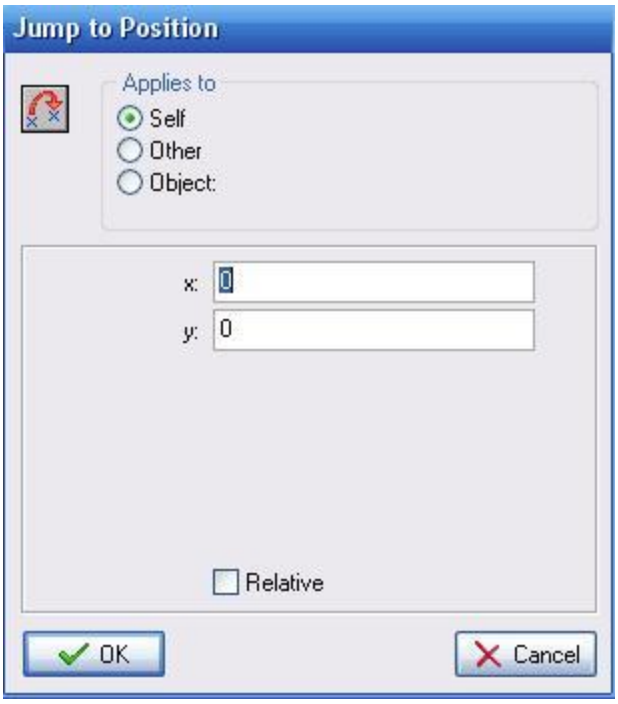
Εικόνα 1.34

Set Friction : Εδώ έχουμε να κάνουμε με ένα Action που αφορά την τριβή του αντικειμένου μας. Εάν το αντικείμενο μας έχει μια κίνηση με τιμή X, και δώσουμε μια τιμή στο Fiction Y, τότε το X μειώνεται συνεχώς κατά Y, μέχρι η τιμή του να γίνει 0, όπου το αντικείμενό μας σταματάει. Εάν το X είναι αρνητικός αριθμός, τότε σε αυτό προστίθεται κάθε φορά το Y, μέχρι και πάλι να γίνει 0. Για το Relative, ισχύει ότι ήδη γνωρίσουμε .



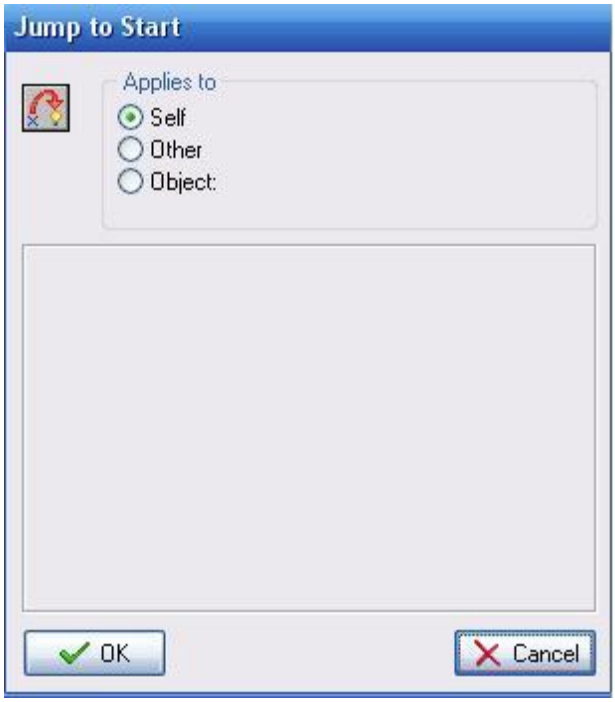
Εικόνα 1.35

Jump to Position : Με αυτό το Action, ορίζουμε ένα σημείο μέσα στο Room, στο οποίο θέλουμε να πάει το αντικείμενό μας, με τη χρήση και πάλι σημείων x και y. Η διαφορά αυτού του Action με το Move Towards, είναι ότι το αντικείμενο απλά θα εμφανιστεί στη θέση που έχουμε προσδιορίσει, χωρίς να φανεί στο παιχνίδι κάποια κίνηση. Για αυτό άλλωστε και δεν ορίζουμε κάποια ταχύτητα εδώ .



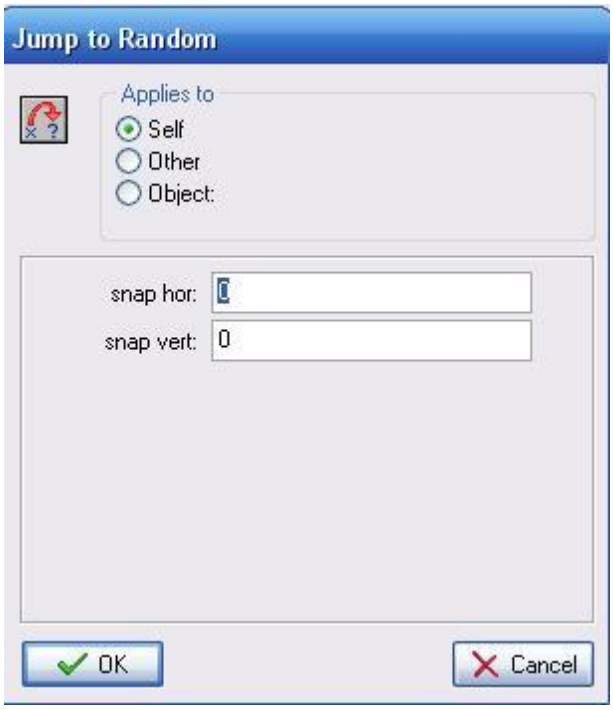
Εικόνα 1.36

Jump To Start : Αυτό το Action, κάνει ότι ακριβώς και το Jump to Position, μόνο που εδώ δεν χρειάζεται να ορίσουμε x και y, διότι το αντικείμενο μεταφέρεται αυτομάτως στην αρχική θέση στην οποία δημιουργήθηκε. Αυτό συνήθως το χρησιμοποιούμε, εάν θέλουμε ο χαρακτήρας μας, αφού έχει χάσει μια ζωή, να αρχίζει ξανά την πίστα μας.



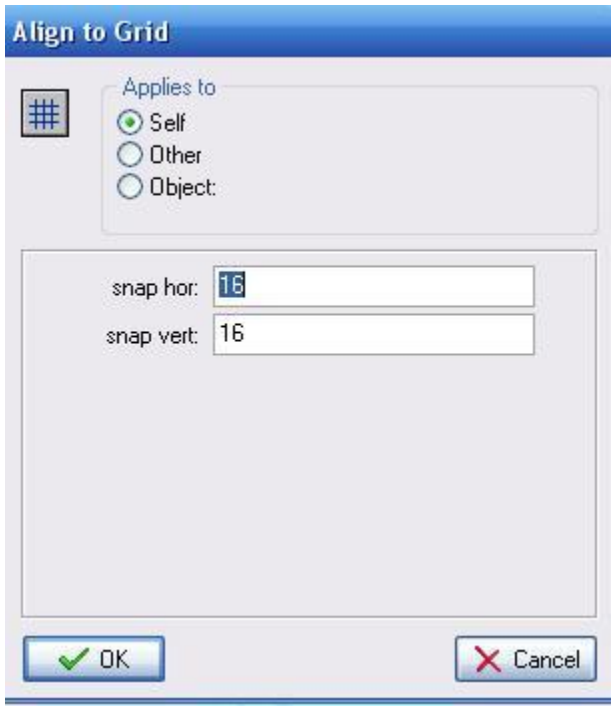
Εικόνα 1.37

Jump To Random : Με αυτό το Action, κάνουμε ότι και στα προηγούμενα δύο, δηλαδή αλλάζουμε την θέση του αντικειμένου, μόνο που σε αυτήν την περίπτωση δίνουμε εντολή να μεταφερθεί σε μια τυχαία θέση (random). Το αντικείμενό μας λοιπόν θα μεταφερθεί σε κάποια «κενή» θέση, δηλαδή σε θέση στην οποία δεν θα έρχεται σε επαφή με κάποιο άλλο αντικείμενο του παιχνιδιού. Δηλώνουμε όμως κάποιες τιμές για την οριζόντια και κάθετη θέση που θέλουμε, στην περίπτωση που θέλουμε να είναι ευθυγραμμισμένο με κάποιο ή με κάποια άλλα αντικείμενα. Για παράδειγμα, ένας παίχτης ο οποίος κινείται σε μια ευθεία, θέλουμε να τον εμφανίσει σε μια τυχαία θέση, αλλά πάλι πάνω σε αυτήν την ευθεία, και όχι για παράδειγμα στον ουρανό.



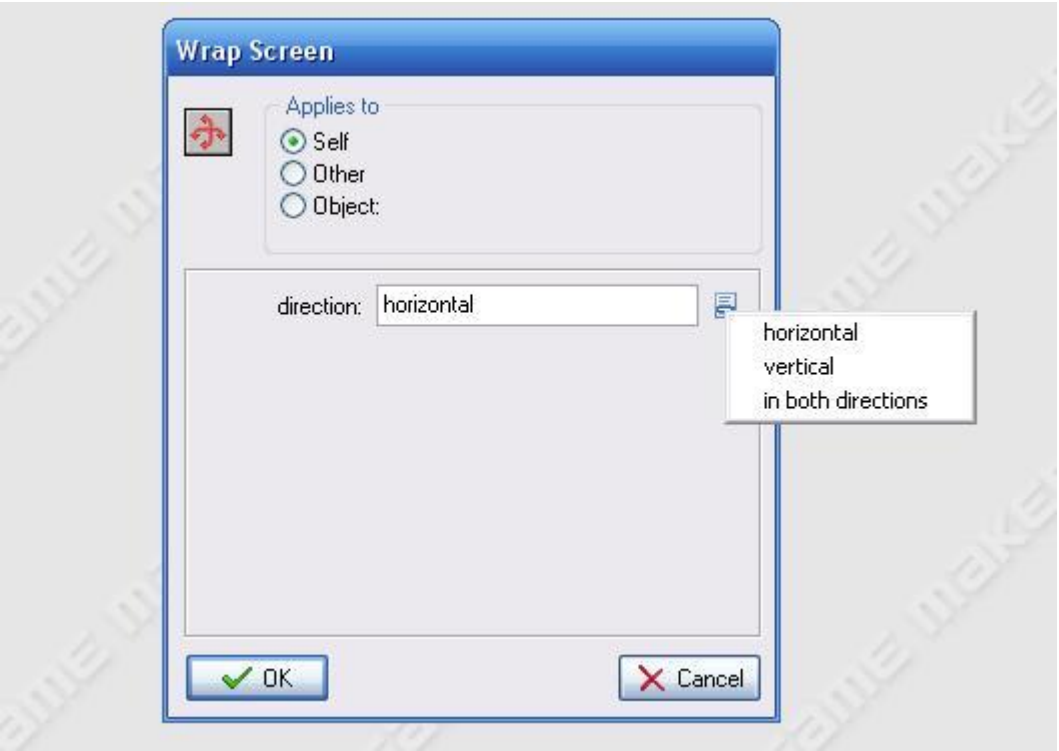
Εικόνα 1.38

Align To Grid : Με αυτό το Action, μπορούμε να ορίσουμε ένα «πλέγμα», με μέγεθος κελιών που εμείς θέτουμε, μέσα στο οποίο θα βρίσκεται το αντικείμενό μας .



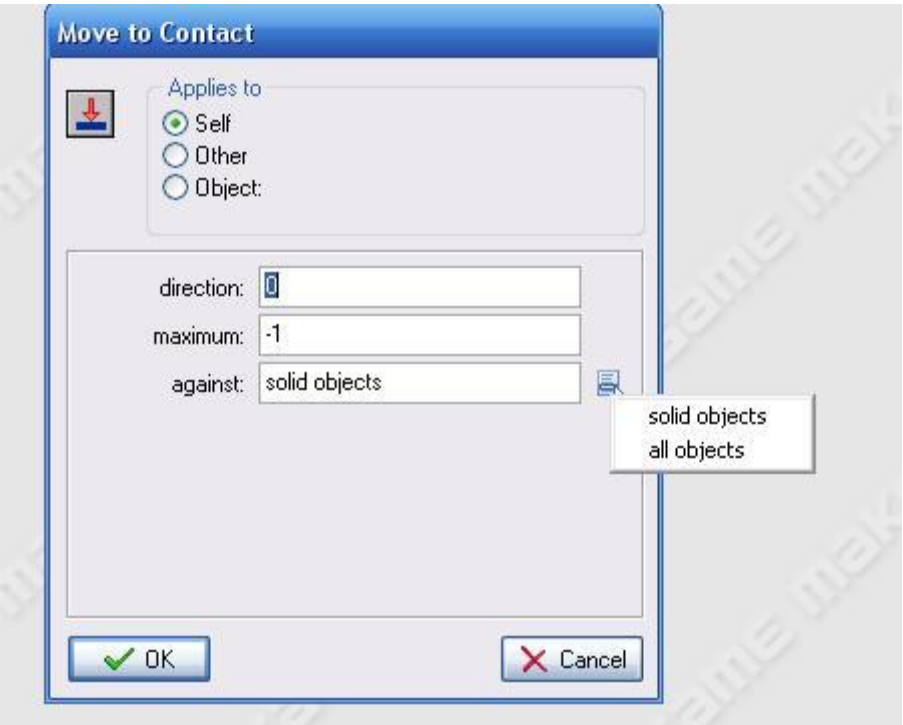
Εικόνα 1.39

Wrap To Screen : Εδώ μας δίνεται η δυνατότητα, όταν το αντικείμενό μας φεύγει από τη μία πλευρά του δωματίου (room), να εμφανίζεται στην άλλη. Αυτό μπορεί να γίνει, για την οριζόντια θέση του, για την κάθετη ή και για τις δύο.



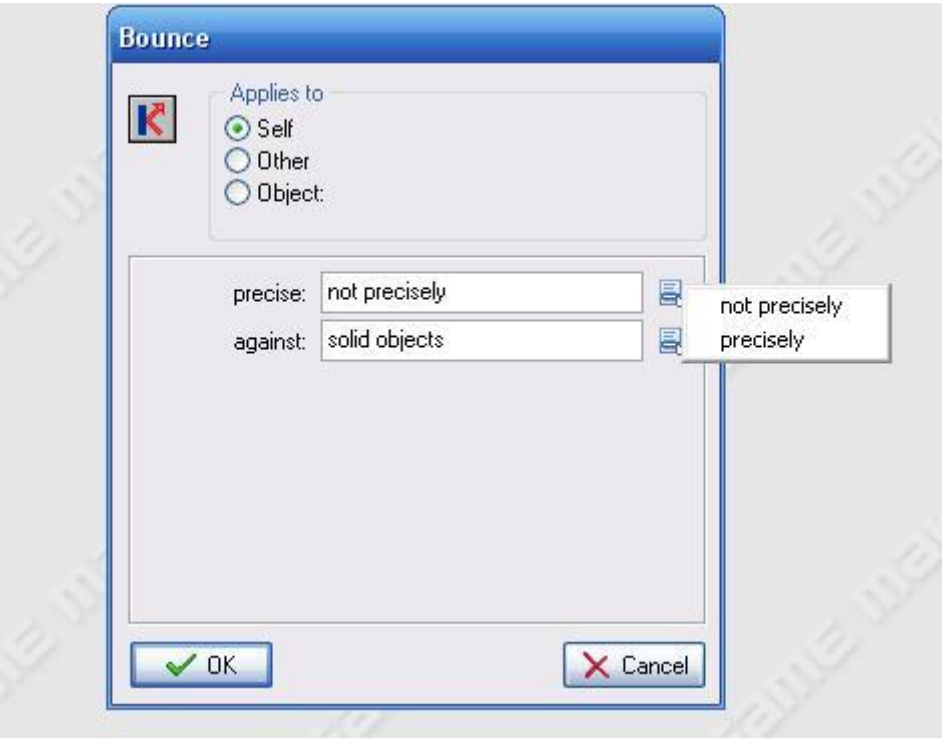
Εικόνα 1.40

Move To Contact : Με αυτό το Action, λέμε στο αντικείμενό μας, να διανύσει μια απόσταση, της οποίας την μέγιστη τιμή προκαθορίσουμε, μέχρι να συναντήσει ένα στερεό αντικείμενο ή γενικά κάποιο αντικείμενο του δωματίου.



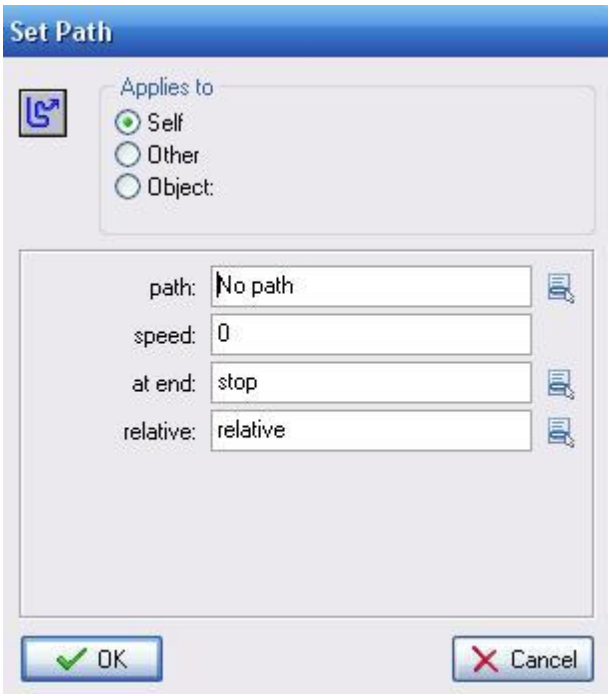
Εικόνα 1.41

Bounce : Αυτό το Action, το δίνουμε σε ένα Collision Event με κάποιο άλλο αντικείμενο. Όταν συναντήσει λοιπόν αυτό το άλλο αντικείμενο, του λέμε να το αναπηδήσει. Υπάρχει μια παράμετρος, precise. Εάν την θέσουμε ως αρνητική, τότε το Action, έχει επίδραση μόνο σε αντικείμενα με οριζόντια ή κάθετη θέση. Αλλιώς θα πρέπει να την θέσουμε ως θετική. Και εδώ επιλέγουμε εάν θέλουμε να ισχύει μόνο σε στερεά (αδιαπέραστα) ή σε όλα τα αντικείμενα (solid objects, all objects)



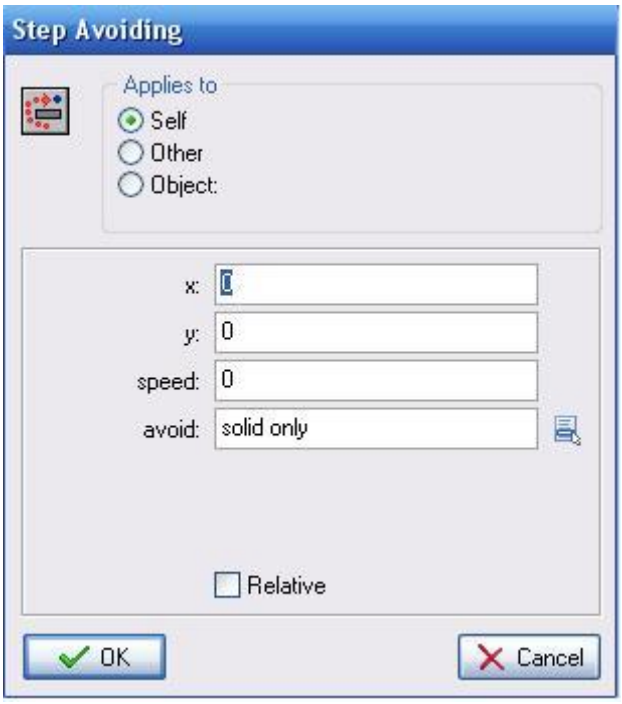
Εικόνα 1.42

Paths : Επιπλέον υπάρχουν τέσσερα (4) Actions που αφορούν τα Paths (μονοπάτια). Την δημιουργία τους, το τέλος τους, τη θέση τους και την ταχύτητά τους, τα οποία όμως σπάνια μας είναι χρήσιμα, αλλά φυσικά καλό είναι να τα γνωρίζουμε.



Εικόνα 1.43

Steps : Και τέλος υπάρχουν ακόμη δύο (2) Actions που αφορούν τα Steps (βήματα). Το ένα έχει να κάνει με την δημιουργία κάποιου βήματος που θα εκτελέσει το αντικείμενό μας προς μία κατεύθυνση, και το δεύτερο μας βοηθάει, όταν το αντικείμενό μας συναντήσει κάποιο άλλο συγκεκριμένο αντικείμενο, απλά να το αποφύγει και να το προσπεράσει.



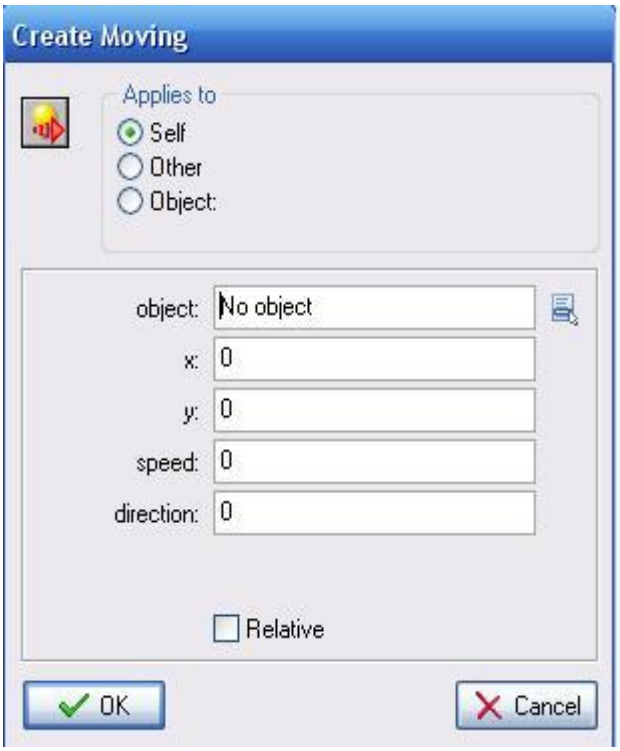
Εικόνα 1.44

ΥΠΟΚΕΦΑΛΑΙΟ 1.2.2.2
MAIN 1 ACTIONS

Create Instance : Με αυτό το Action, μπορούμε να δημιουργήσουμε και να «εμφανίσουμε» ένα αντικείμενο μέσα στο δωμάτιο. Επιλέγουμε φυσικά κάποιο από τα αντικείμενα που έχουμε δημιουργήσει για τις ανάγκες του παιχνιδιού. Και φυσικά επιλέγουμε και σε ποια θέση του δωματίου (room) θέλουμε να τοποθετηθεί. Εάν τσεκάρουμε το Relative, τότε οι συντεταγμένες που θα δώσουμε, προστίθενται σε αυτές που έχει εκείνη τη στιγμή το αντικείμενο στο οποίο εφαρμόζουμε το Action. Επομένως είναι μια πάρα πολύ χρήσιμη επιλογή που μας δίνεται. Για παράδειγμα, με την χρήση κάποιου πλήκτρου που έχουμε ορίσει, εμφανίζεται ένα σπαθί μπροστά στο χαρακτήρα μας.

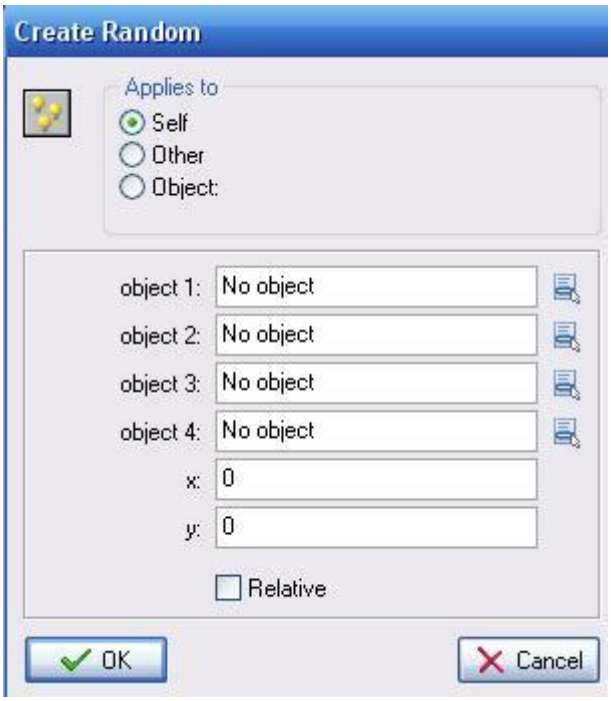
Εικόνα 1.45

Create Moving : Εδώ μας δίνεται η δυνατότητα να δημιουργήσουμε ένα αντικείμενο, με τη διαφορά ότι σε αυτήν την περίπτωση, το αντικείμενο αυτό θα έχει και κάποια κίνηση προς μια συγκεκριμένη κατεύθυνση. . Για παράδειγμα, μας βοηθάει, σε ένα παιχνίδι που χρησιμοποιεί κάποιο όπλο, να εμφανίζεται μια σφαίρα μπροστά από το όπλο, μόλις πατήσουμε κάποιο πλήκτρο που έχουμε ορίσει. Και η σφαίρα αυτή κινείται προς την φορά που έχει ο χαρακτήρας μας. Επομένως, θα πρέπει να ορίσουμε, εκτός από την θέση του, την ταχύτητα με την οποία θα κινείται αλλά και την κατεύθυνση που θα έχει.



Εικόνα 1.46

Create Random : Ισχύει ότι και στο Create Instance, με μια διαφορά. Επιλέγομε τέσσερα (4) από τα αντικείμενα που έχουμε ήδη δημιουργήσει στο παιχνίδι, και το Action αυτό μας εμφανίζει ένα από αυτά τα τέσσερα, τυχαία. Και το εμφανίζει κι εδώ στο σημείο που θα ορίσουμε, με την βοήθεια των x και y.

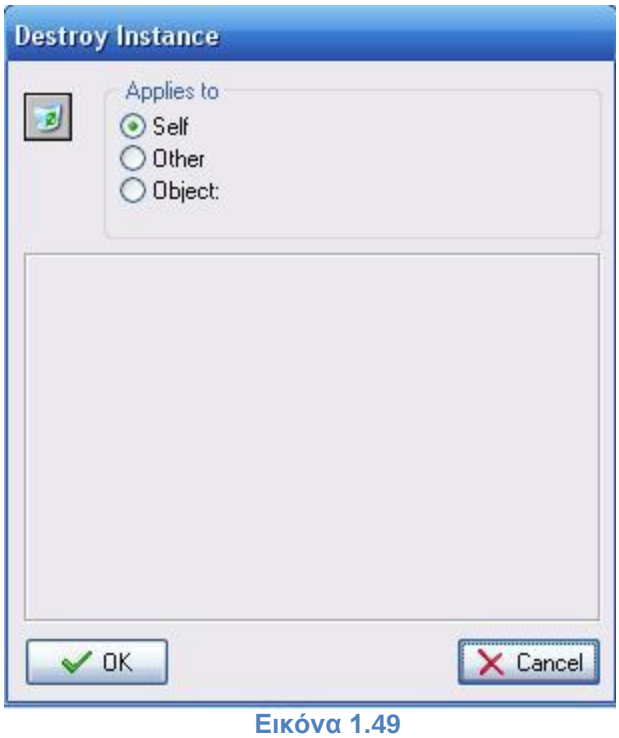


Εικόνα 1.47

Change Instance : Με αυτό το Action, αντικαθιστάμε το αντικείμενό μας με κάποιο άλλο. Δηλώνουμε ποίο αντικείμενο θέλουμε να αντικατασταθεί (το ίδιο ή κάποιο άλλο), αλλά και το πότε θέλουμε να γίνει αυτό, μετά το Event ή όχι και τα λοιπά.

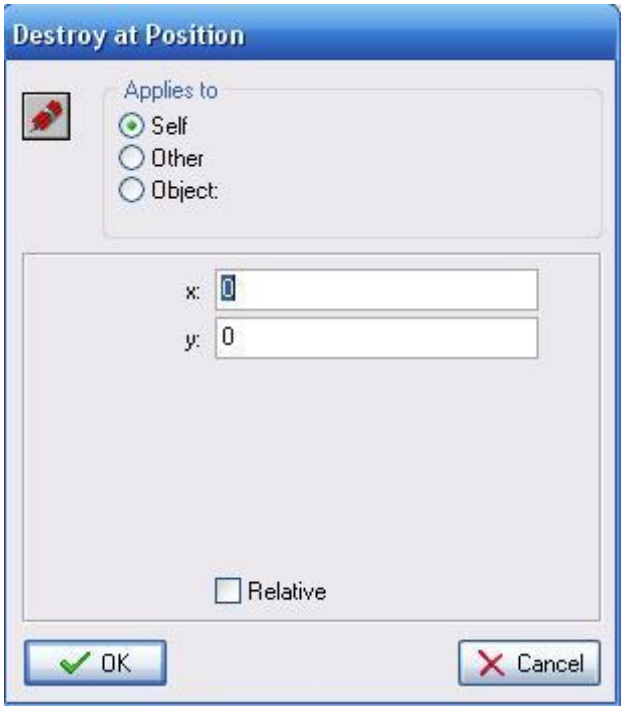
Εικόνα 1.48

Destroy Instance : Και συνεχίζουμε με ένα Action, το οποίο μας δίνει την δυνατότητα να καταστρέψουμε ένα αντικείμενο και να το αφαιρέσουμε στην ουσία από το δωμάτιο (room) .



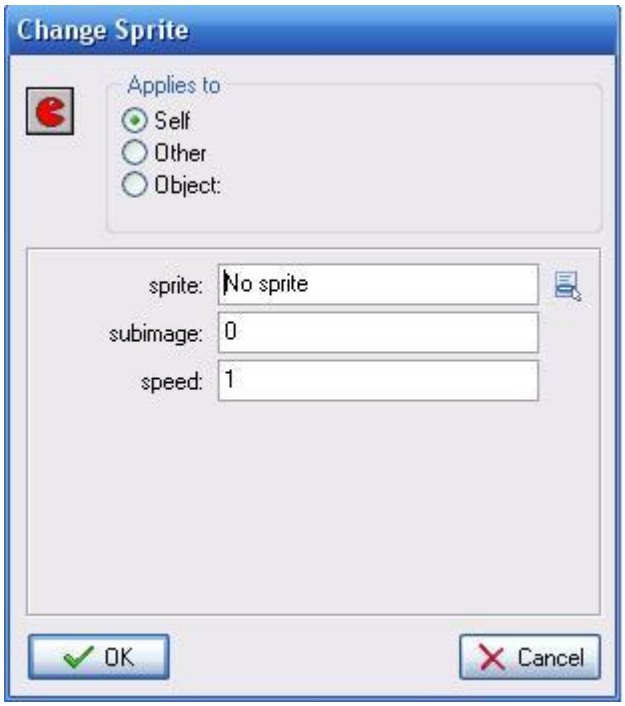
Εικόνα 1.49

Destroy At Position : Καταστρέφει το αντικείμενο ή τα αντικείμενα τα οποία βρίσκονται σε κάποια συγκεκριμένη θέση, που εμείς ορίζουμε. Θέτουμε το x και το y, καθώς εάν θέλουμε τσεκάρουμε το Relative, με τα αποτελέσματα αυτής της επιλογής που ήδη έχουμε προαναφέρει.



Εικόνα 1.50

Change Sprite : Με αυτό το Action, μπορούμε να αντικαταστήσουμε το υπάρχον Sprite που αντιπροσωπεύει το αντικείμενό μας, με κάποιο άλλο. Στην ουσία , απλά αλλάζει η εικόνα, η «εμφάνιση» που θα έχει το αντικείμενό μας.



Εικόνα 1.51

Transform Sprite : Με αυτό το Action, μπορούμε να τροποποιήσουμε και να αλλάξουμε το μέγεθος που έχει το Sprite, το οποίο αντιπροσωπεύει το αντικείμενό μας. Επομένως μπορούμε να προσδιορίσομε το ύψος και το πλάτος του, την φορά που θα έχει και την κατεύθυνσή του.

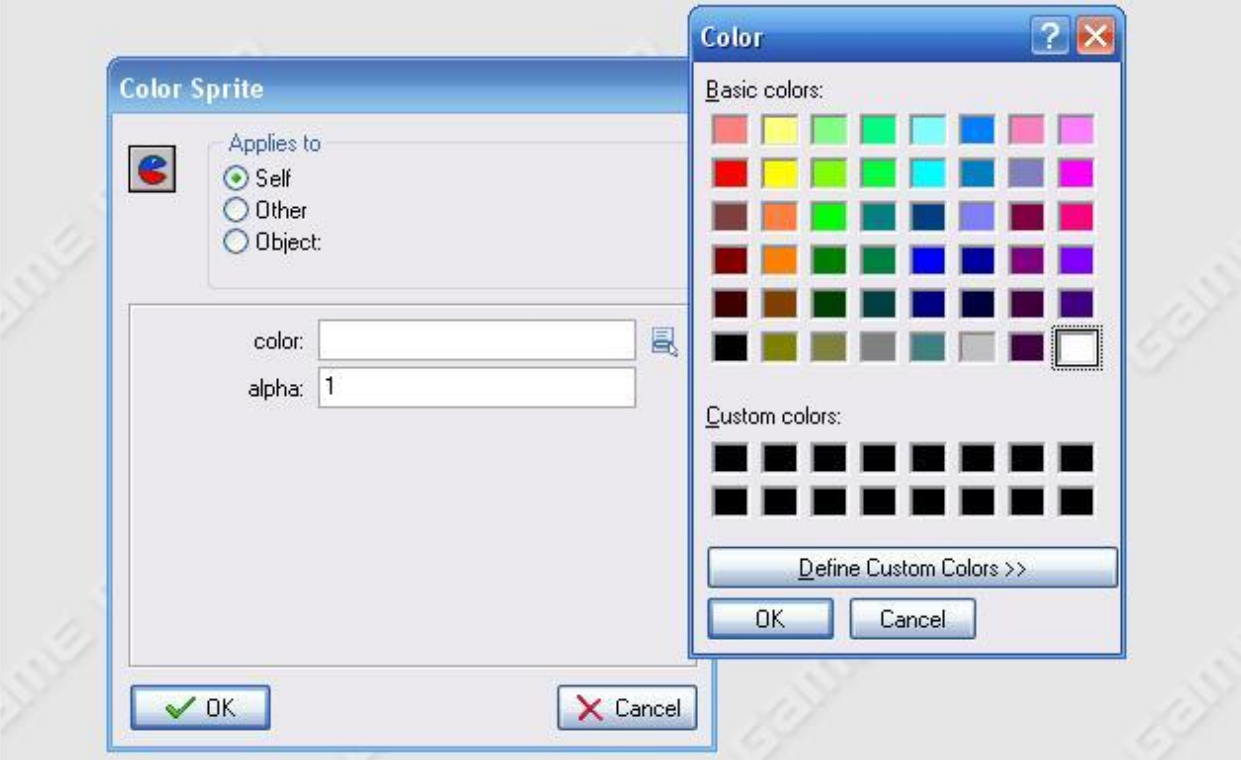
Εικόνα 1.52

Το παρακάτω πινακάκι πιστεύω πως μας είναι πολύ χρήσιμο, όσο αφορά τις αλλαγές που θα γίνουν στο Sprite, σε σχέση με την τιμή του Scale που θα δώσουμε.

Scale	Width/Height change	Mirrored
Less than -1	Larger	Yes
-1	Normal	Yes
0 to -1	Smaller	Yes
0	None	No
0 to 1	Smaller	No
1	Normal	No
More than 1	Larger	No

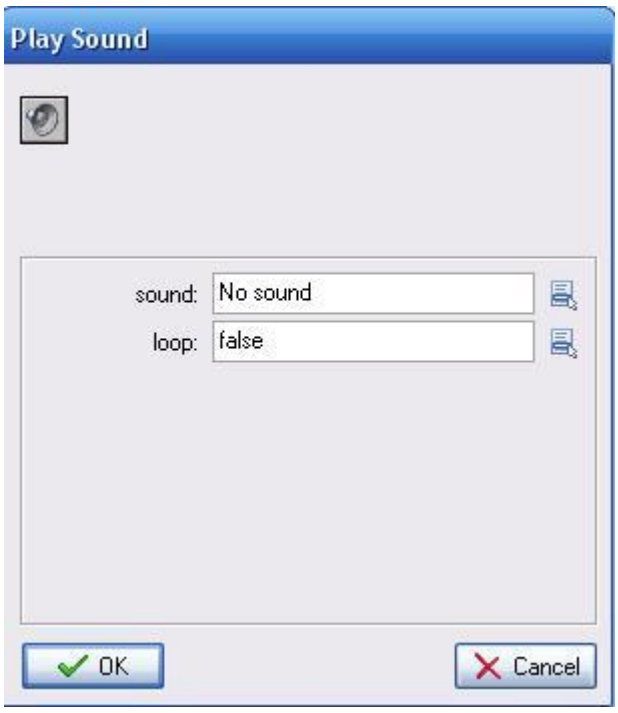
Πίνακας 1.1

Color Sprite : Με αυτό το Action, αλλάζουμε το χρώμα του Sprite. Στην ουσία δηλώνουμε με ποιο χρώμα θέλουμε να αναμιχθούν τα υπάρχον χρώματα του Sprite, ώστε να έχουμε το επιθυμητό αποτέλεσμα. Επίσης με το «alpha», δηλώνουμε την διαφάνεια που θέλουμε να έχει το Sprite. Εάν αυτή η μεταβλητή έχει τιμή 1, τότε το Sprite είναι πλήρες ορατό. Αυτή είναι και default τιμή της. Εάν το θέσουμε ως 0 (μηδέν), τότε γίνεται αόρατο (invisible).



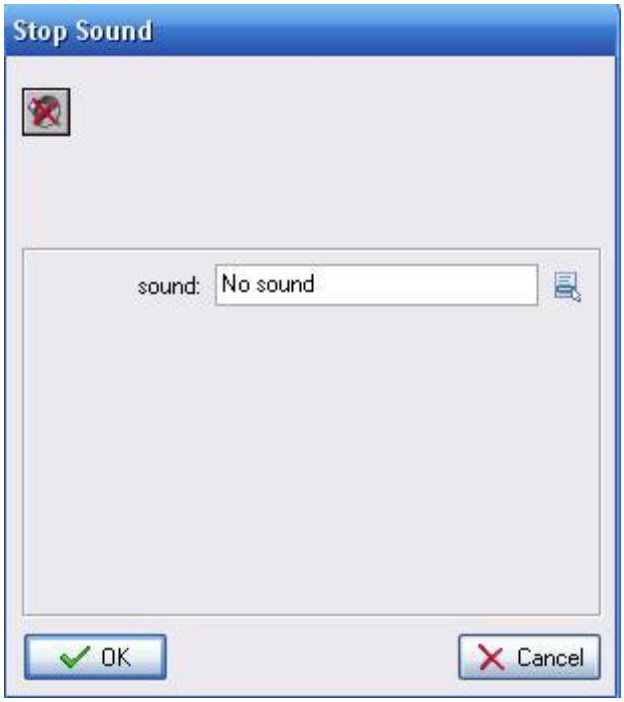
Εικόνα 1.53

Play Sound : Με αυτό το Action, ορίζουμε έναν ήχο, που έχουμε ήδη εισάγει στο Sounds, και ζητάμε την αναπαραγωγή του, μία φορά ή καθ' επανάληψη.



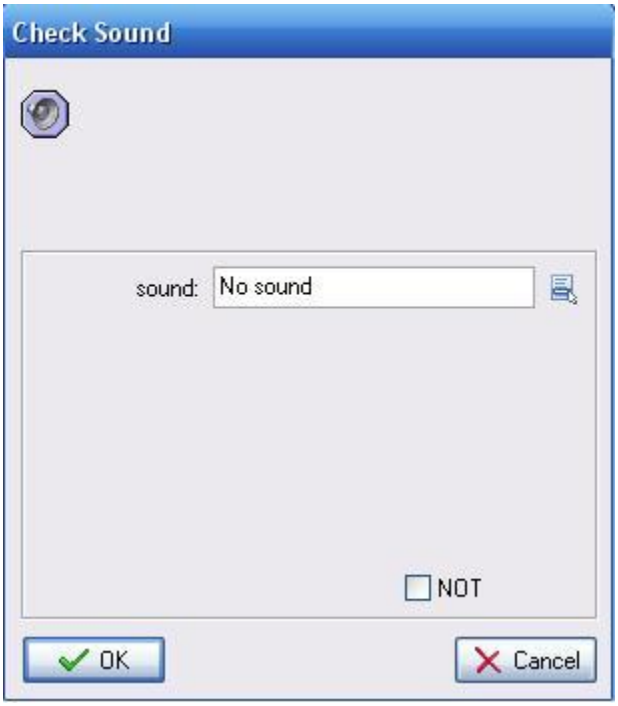
Εικόνα 1.54

Stop Sound : Αντίστοιχα, με αυτό το Action δίνουμε εντολή να σταματήσει η αναπαραγωγή κάποιου ήχου, που έχουμε ήδη ενεργοποιήσει. Αυτό το κάνουμε, για παράδειγμα, εάν θέλουμε να διακόψουμε την αναπαραγωγή κάποιου ήχου, κατά τη διάρκεια του παιχνιδιού, πριν ακόμα αυτός προλάβει να ολοκληρωθεί., ή απλά θέλουμε να σταματήσει το loop, δηλαδή η πολλαπλή και συνεχόμενη αναπαραγωγή του.



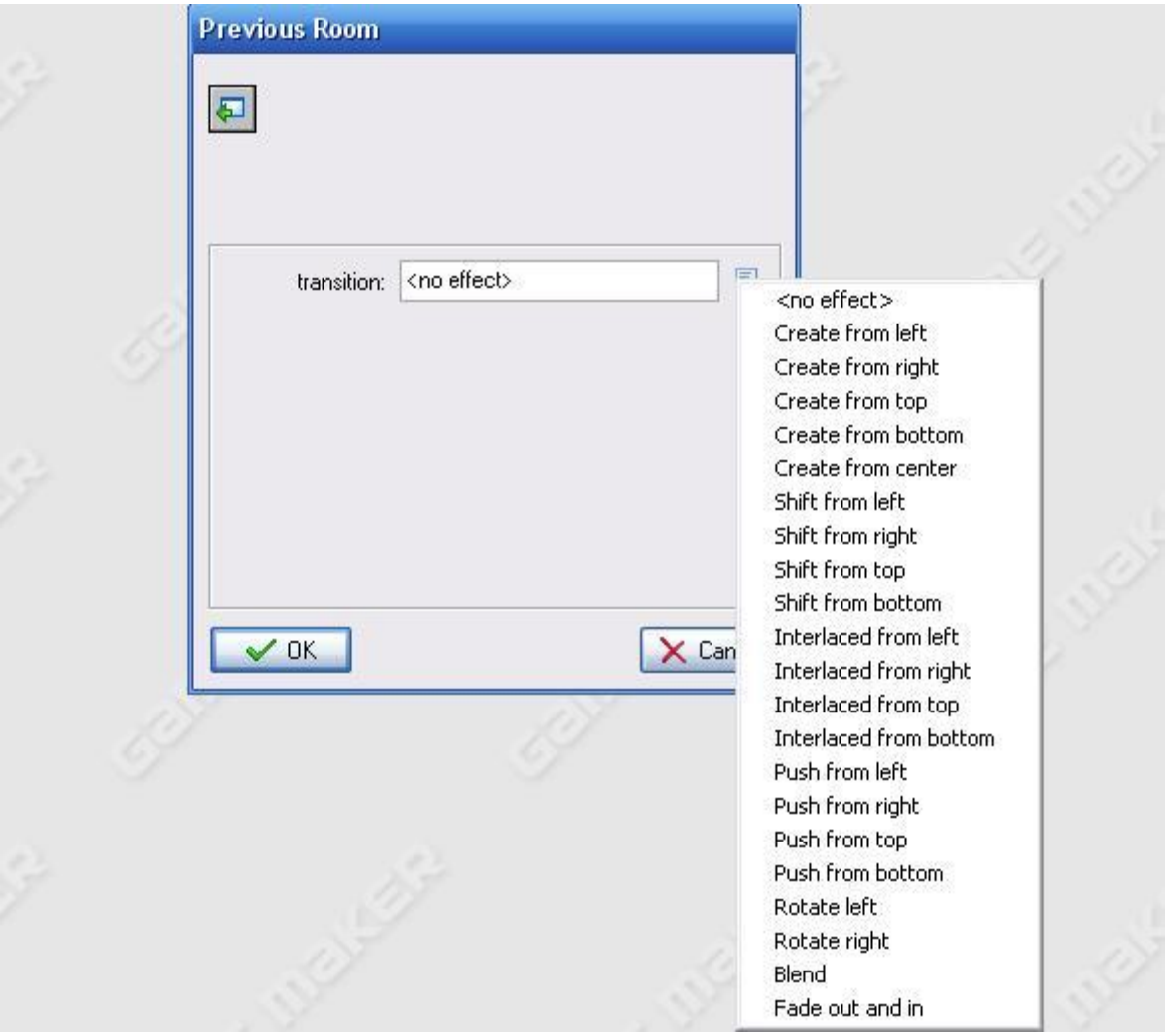
Εικόνα 1.55

Check Sound : Με αυτό το Action, ελέγχουμε την ροή του ήχου που ακούμε κατά την διάρκεια του παιχνιδιού. Ελέγχουμε εάν παίζει κάποιος ήχος, εάν όχι τότε πράττουμε ανάλογα, όπως για παράδειγμα ξεκινάμε την αναπαραγωγή κάποιου άλλου ήχου, με το Play Sound .



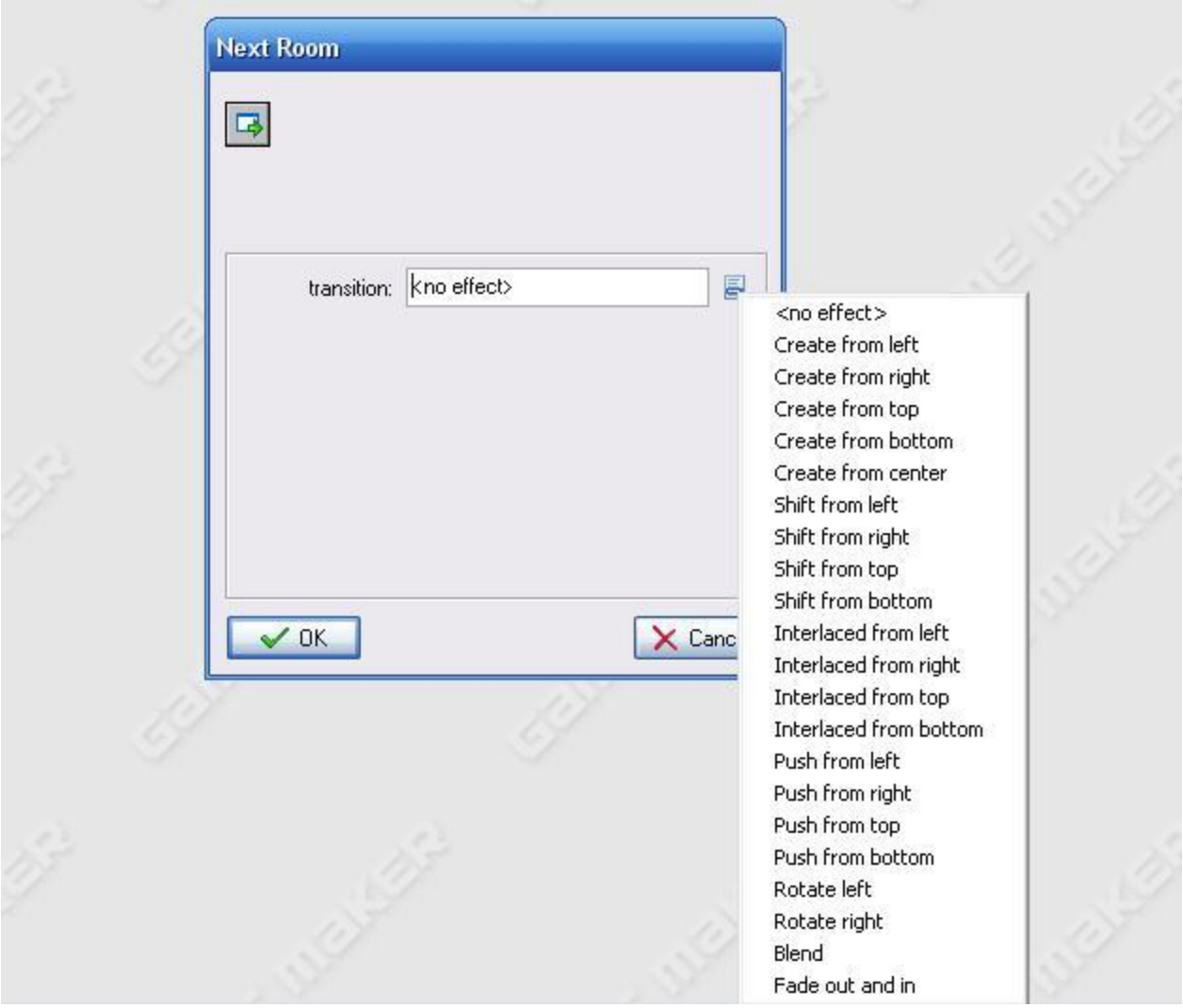
Εικόνα 1.56

 **Previous Room** : Με αυτό το Action, ζητάμε να μεταφερθούμε στο αμέσως προηγούμενο δωμάτιο (room) του παιχνιδιού μας. Επίσης επιλέγουμε τον τρόπο εμφάνισης αυτής της μεταφοράς, δηλαδή με ποιο effect (εφέ) θα γίνει. Εάν είμαστε ήδη στο πρώτο room τότε εμφανίζεται κάποιο σφάλμα (error) .



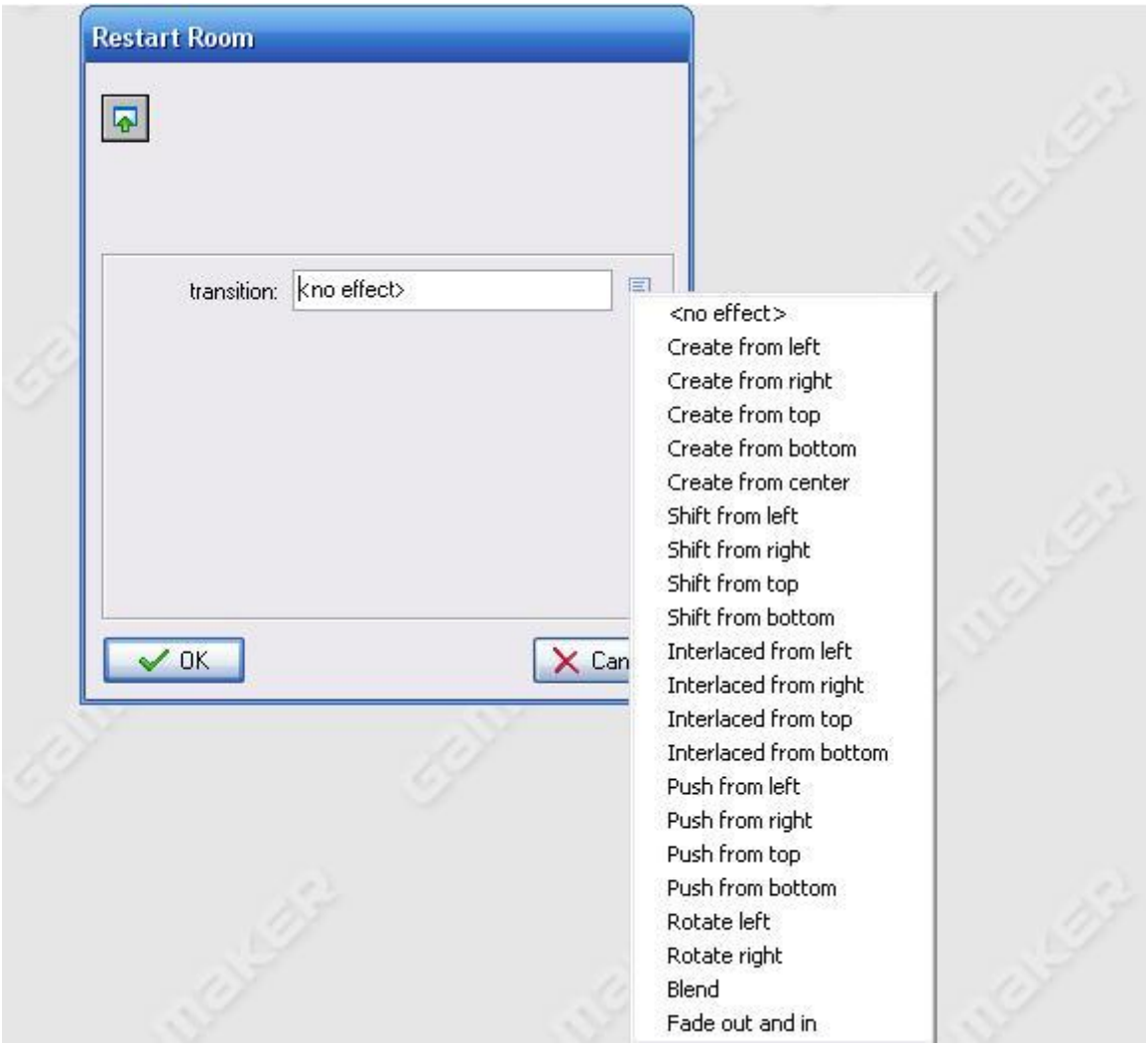
Εικόνα 1.57

Next Room : Αντίστοιχα, με αυτό το Action ζητάμε να μεταφερθούμε στο επόμενο room. Κι εδώ επιλέγουμε τον τρόπο εμφάνισης αυτής της μεταφοράς, δηλαδή με ποιο effect (εφέ) θα γίνει .



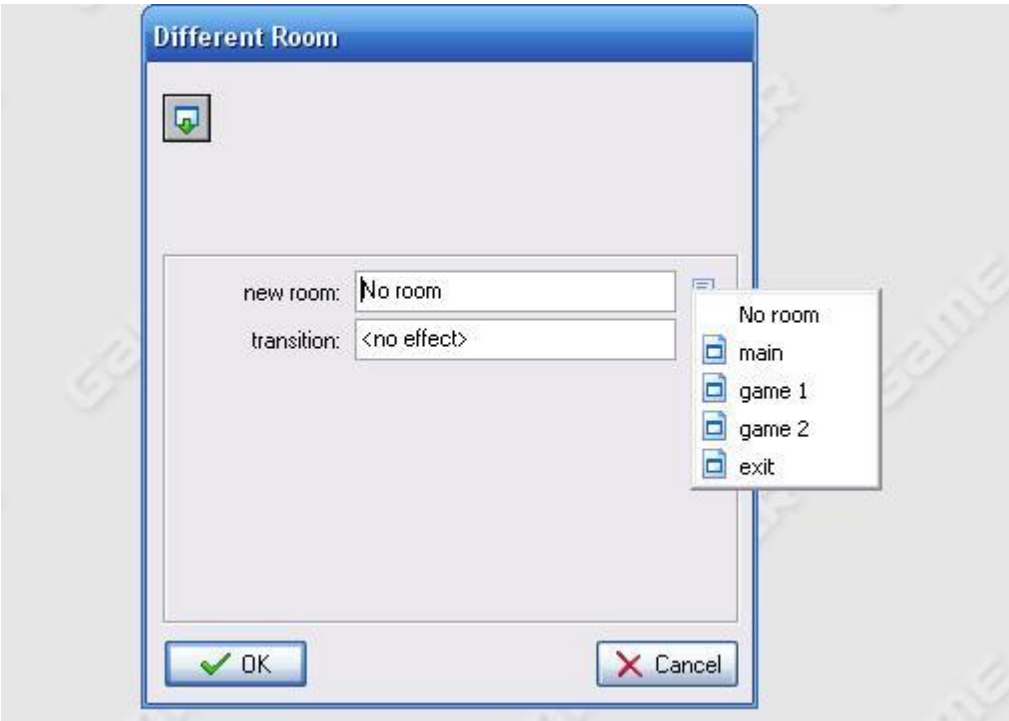
Εικόνα 1.58

Restart Room : Με αυτό το Action, ζητάμε να γίνει επανεκίνηση του δωματίου (room) στο οποίο βρισκόμαστε ήδη . Και πάλι επιλέγουμε τον τρόπο εμφάνισης αυτής της μεταφοράς, δηλαδή με ποιο effect (εφέ) θα γίνει .



Εικόνα 1.59

Different Room : Εδώ ζητάμε να μεταφερθούμε σε κάποιο συγκεκριμένο δωμάτιο, όποιο επιλέξουμε εμείς, άσχετα αν είναι προηγούμενο, επόμενο ή τίποτα από τα δύο. Και εδώ πάλι επιλέγουμε το effect της μεταφοράς και της εναλλαγής.



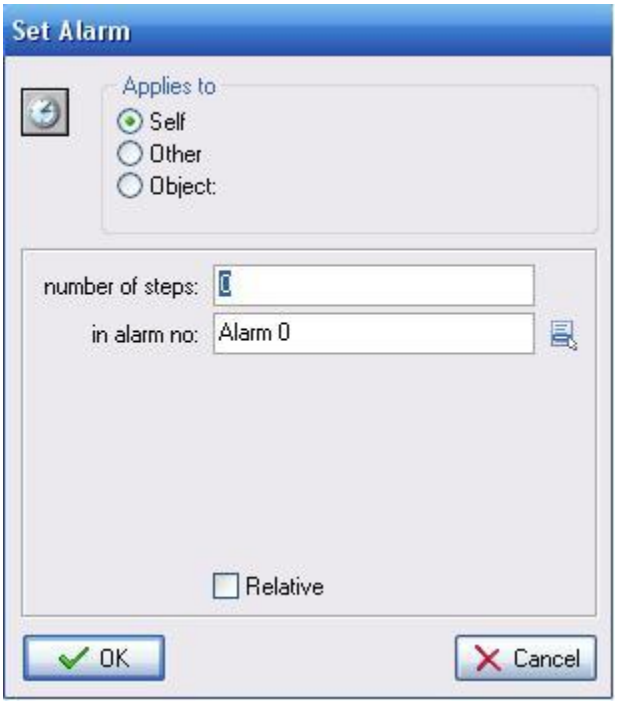
Εικόνα 1.60

Check Previous : Εδώ ελέγχουμε εάν υπάρχει κάποιο δωμάτιο ακριβώς πριν από αυτό στο οποίο βρισκόμαστε ήδη. Αυτό συνήθως το κάνουμε πριν επιλέξουμε το Action Previous Room .

Check Next : Αντίστοιχα εδώ ελέγχουμε εάν υπάρχει κάποιο δωμάτιο ακριβώς μετά από αυτό στο οποίο βρισκόμαστε ήδη, ή αν είμαστε στο τελευταίο δωμάτιο .

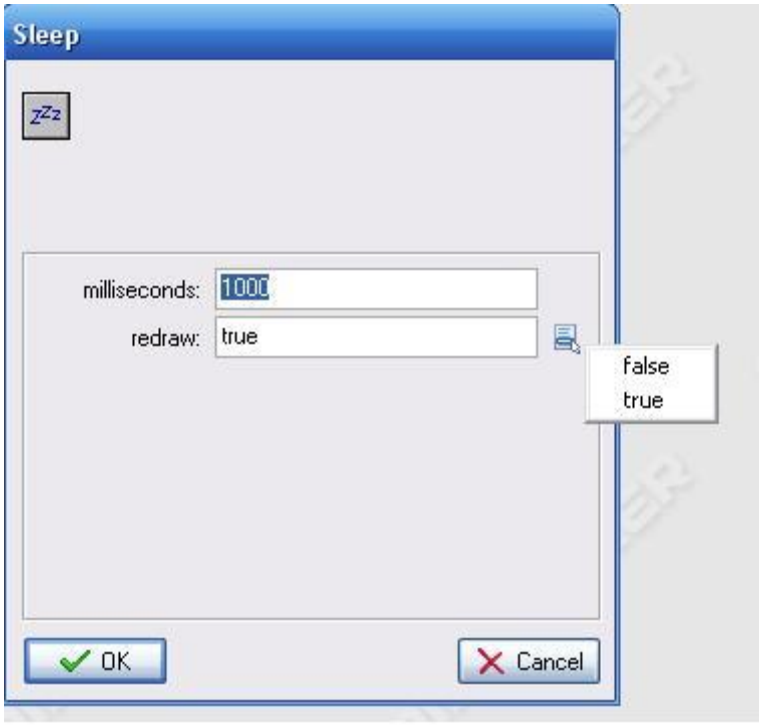
ΥΠΟΚΕΦΑΛΑΙΟ 1.2.2.3
MAIN 2 ACTIONS

Set Alarm : Με αυτό το Action ορίζουμε κάποιο από τα δώδεκα (12) Alarm Clocks, μετά από τον αριθμό των steps (σε milliseconds) που θα ορίσουμε. Με αυτόν τον τρόπο ενεργοποιούμε τα αντίστοιχα Alarm Events. Για ρο Relative έχουμε ξανά πει πιο πάνω τι ρόλο παίζει. Εάν θέσουμε(ή πάρει την τιμή λόγο του Relative) το Alarm Clock μικρότερο ή ίσο του μηδενός (< = 0), τότε απενεργοποιείται και τα αντίστοιχα Events δεν ενεργοποιούνται.



Εικόνα 1.61

Sleep : Με αυτό το Action στην ουσία «κοκαλώνουμε» το παιχνίδι, σταματάμε την σκηνή σε εκείνο το χρονικό σημείο, για όσα δευτερόλεπτα εμείς ορίσουμε. Με αυτόν τον τρόπο μπορούμε για παράδειγμα να εμφανίσουμε κάποιο μήνυμα κατά τη διάρκεια του παιχνιδιού. Επίσης ορίζουμε εάν η σκηνή την οποία «παγώσαμε» θα επανασχεδιαστεί ή όχι.

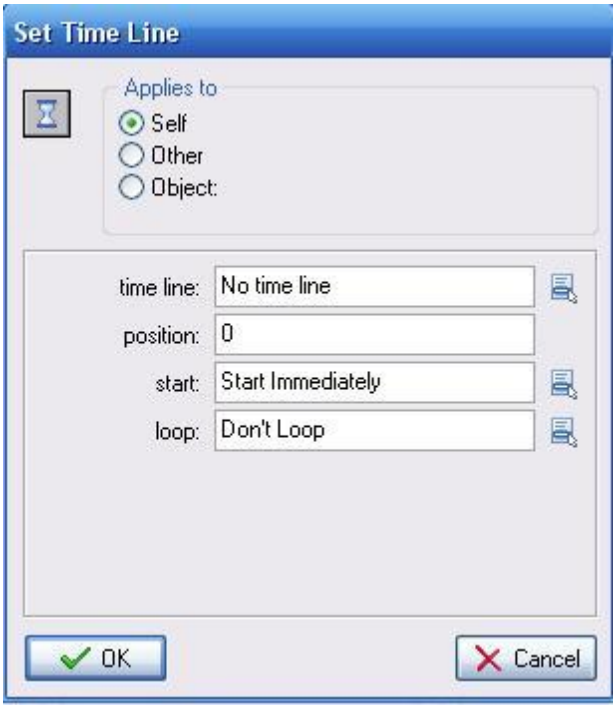


Εικόνα 1.62

Υπάρχουν ακόμα έξι (6) Actions χρόνου, με τα οποία προσδιορίζουμε κάποιο χρονοδιάγραμμα (timeline) . Αυτό μας βοηθάει στο να ελέγχουμε το τον χρόνο και το χρονικό διάστημα στο οποίο θέλουμε να προσδιορίσουμε κάποια Actions και κάποιες ενέργειες. Τα Actions αυτά είναι τα παρακάτω :

- **Set Time Line**
- **Time Line Position**
- **Time Line Speed**
- **Start Time Line**
- **Pause Time Line**
- **Stop Time Line**

Στο πρώτο (το οποίο βλέπουμε και στην εικόνα), ορίζουμε το Time Line, αλλά και άλλες παραμέτρους όπως το σημείο στο οποίο θέλουμε να ξεκινήσει (0 αν θέλουμε να ξεκινήσει από την αρχή) , και εάν θέλουμε μόλις τελειώσει να ξαναρχίσει από την αρχή ή να σταματήσει (Loop). Τα υπόλοιπα νομίζω πως καταλαβαίνουμε τη λειτουργία που έχουν.

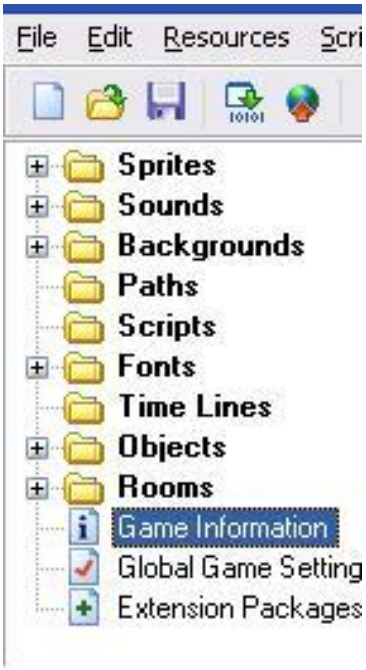


Εικόνα 1.63

Display Message : Αυτό το Action μας δίνει την δυνατότητα να εμφανίσουμε κάποιο μήνυμα. Ίσως αυτό μπορούμε να το κάνουμε σε συνεργασία με το Sleep Action εάν θέλουμε όσο εμφανίζεται το μήνυμα να μην «τρέχει» το παιχνίδι.

Εικόνα 1.64

Show Info : Με αυτό το Action εμφανίζουμε το παράθυρο με τις πληροφορίες για το παιχνίδι μας, τις οποίες έχουμε ήδη συντάξει. Αυτό το κάνουμε συνήθως στην αρχή, μόλις ξεκινάει το παιχνίδι . Μπορούμε να δώσουμε για παράδειγμα πληροφορίες για τον χειρισμό του παιχνιδιού. Την σύνταξη αυτών των πληροφοριών την κάνουμε εδώ, όπως βλέπουμε στην εικόνα .



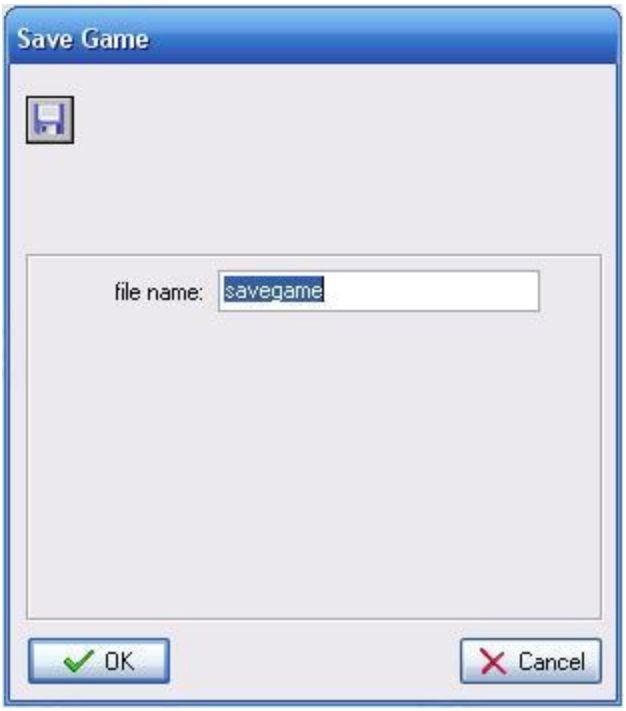
Εικόνα 1.65

Με τα Actions **Splash Video, Splash Image, Splash Webpage, Splash Video, Splash Settings** κάνουμε ότι και με το Show Info**,** ανάλογα με το τι θέλουμε να εμφανίσουμε, βίντεο, εικόνα και τα λοιπά.

Restart Game : Με αυτό το Action, προφανώς δίνουμε την εντολή να ξανά αρχίσει το παιχνίδι από την αρχή.

End Game : Και προφανώς εδώ τερματίζουμε, σταματάμε το παιχνίδι, απλά.

Αντίστοιχα και με τα **Save** και **Load Game**, όπου στο σημείο στο οποίο τα τοποθετούμε, αποθηκεύουμε το παιχνίδι, ή φορτώνουμε κάποιο άλλο.



Εικόνα 1.66

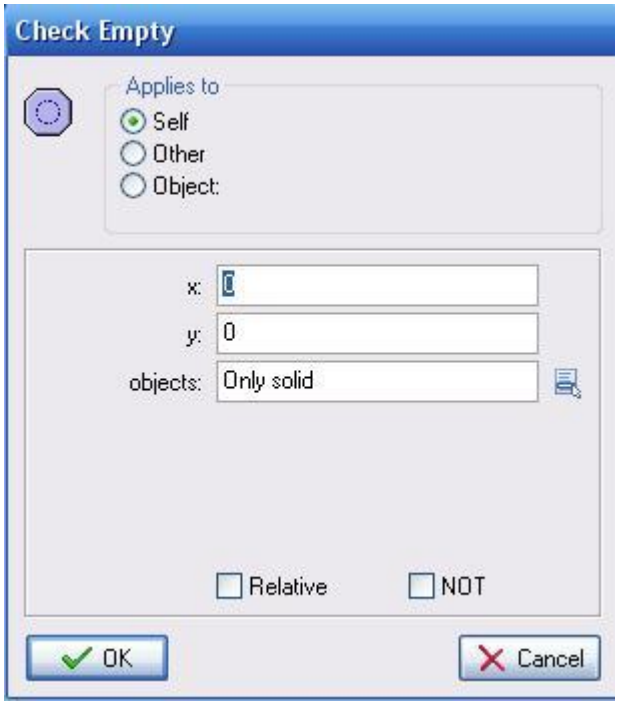
Και τέλος για αυτήν την ομάδα Action, υπάρχουν ακόμη τρείς ενέργειες (3) οι οποίες προφανώς δεν χρειάζονται επεξήγηση για το σε τι χρησιμεύουν :

- **Replace Sprite**
- **Replace Sound**
- **Replace Background**

Εικόνα 1.67

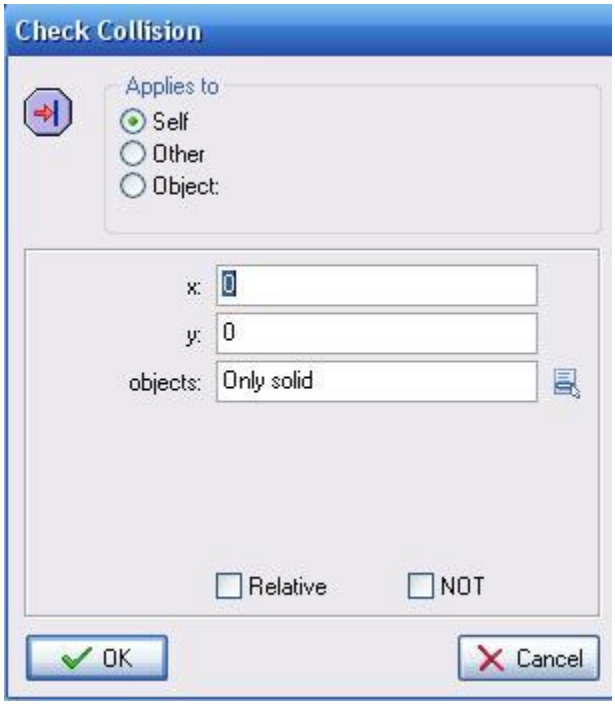
ΥΠΟΚΕΦΑΛΑΙΟ 1.2.2.4
CONTROL ACTIONS

Check Empty : Με αυτό το Action ελέγχουμε εάν σε κάποιο σημείο στο οποίο θέλουμε να κινηθεί το αντικείμενό μας υπάρχει κάποιο άλλο αντικείμενο, ή υπάρχει κενό. Εάν είναι κενό, τότε μπορεί να κατευθυνθεί σε αυτό το σημείο. Θέτουμε επίσης εάν ο έλεγχος αυτός θέλουμε να αφορά μόνο τα Solid αντικείμενα ή όλα.



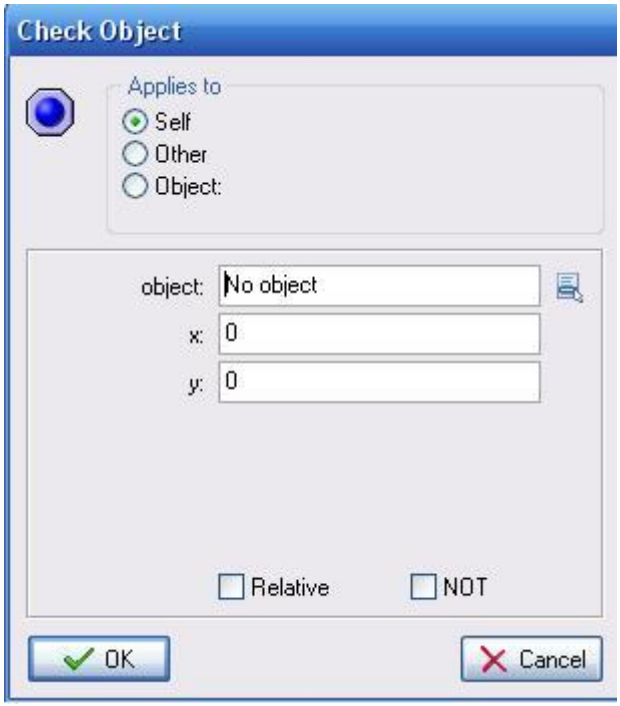
Εικόνα 1.68

Check Collision : Αυτό το Action είναι το αντίστροφο με το Check Empty. Μας επιστρέφει «true» εάν στις συντεταγμένες που δίνουμε δημιουργείται κάποια αλληλεπίδραση (collision) με κάποιο άλλο αντικείμενο , δηλαδή το σημείο αυτό δεν είναι κενό.



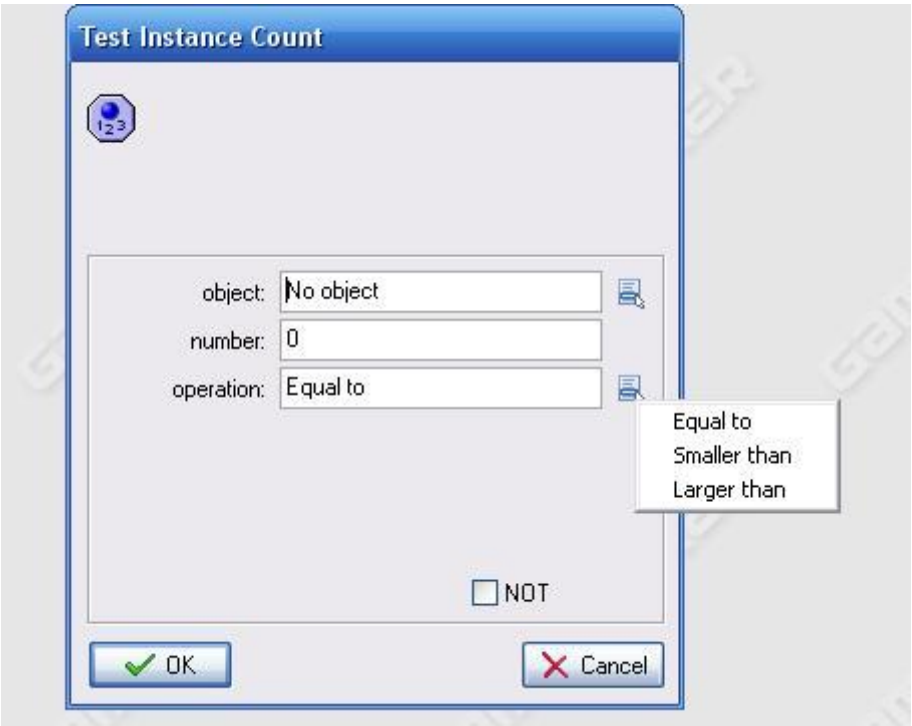
Εικόνα 1.69

Check Object : Εδώ ελέγχουμε εάν σε κάποιο συγκεκριμένο σημείο του δωματίου, υπάρχει κάποιο συγκεκριμένο αντικείμενο, και πράττουμε ανάλογα.



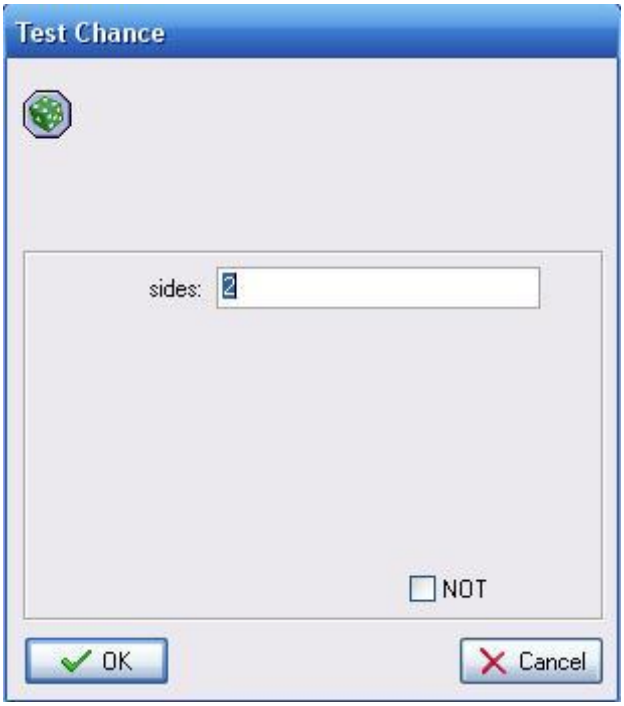
Εικόνα 1.70

Test Instance Count : Εδώ ελέγχουμε εάν κάποιο συγκεκριμένο αντικείμενο εμφανίζεται λιγότερες φορές, ίσες ή περισσότερες από έναν συγκεκριμένο αριθμό που δίνουμε.



Εικόνα 1.71

Test Chance : Με αυτό το Action, είναι σαν να ρίχνουμε ένα ζάρι, το οποίο έχει όσες πλευρές ορίσουμε εμείς, και αν έρθει 1, τότε εκτελούνται οι παρακάτω εντολές που θα δώσουμε. Ένα δεν έρθει κάποιος άλλος αριθμός τότε δεν εκτελούνται (το αντίστροφο συμβαίνει εάν επιλέξουμε το NOT) . Δεν έχει καμία αξία να θέσουμε τον αριθμό των πλευρών μικρότερο του 1. Όσο πιο πολλές είναι οι πλευρές του ζαριού, τόσο λιγότερες είναι και οι πιθανότητες να φέρουμε 1, άρα και να εκτελεστούν οι παρακάτω εντολές.

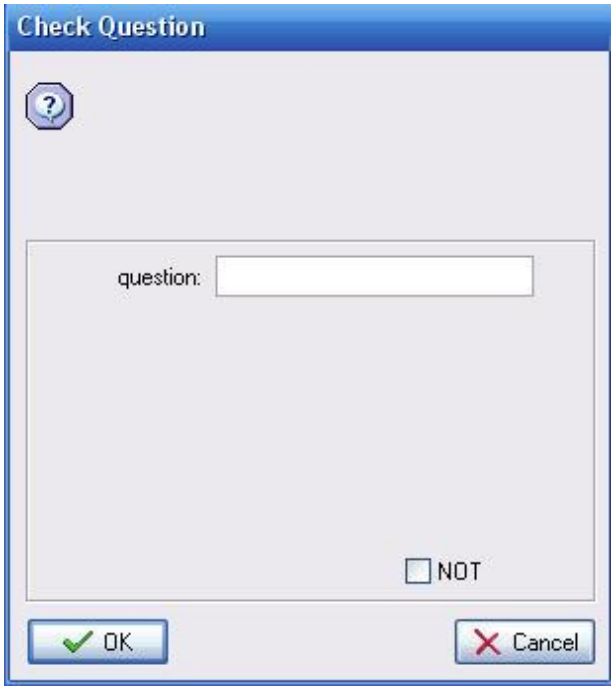


Εικόνα 1.72

Check Question : Εμφανίζουμε μια ερώτηση στον παίκτη του παιχνιδιού, την οποία και τυπώνουμε εμείς, ο οποίος παίκτης πρέπει να απαντήσει πατώντας ένα από τα δύο κουμπιά που του εμφανίζονται « Ναι ή Όχι » . Εάν απαντήσει «Ναι» τότε εκτελούνται και οι παρακάτω εντολές που θα θέσουμε, ενώ εάν απαντήσει «Όχι» τότε δεν εκτελούνται. Το αντίστροφο και πάλι ισχύει εάν επιλέξουμε το Check Box «NOT» κατά την εισαγωγή αυτού του Action.

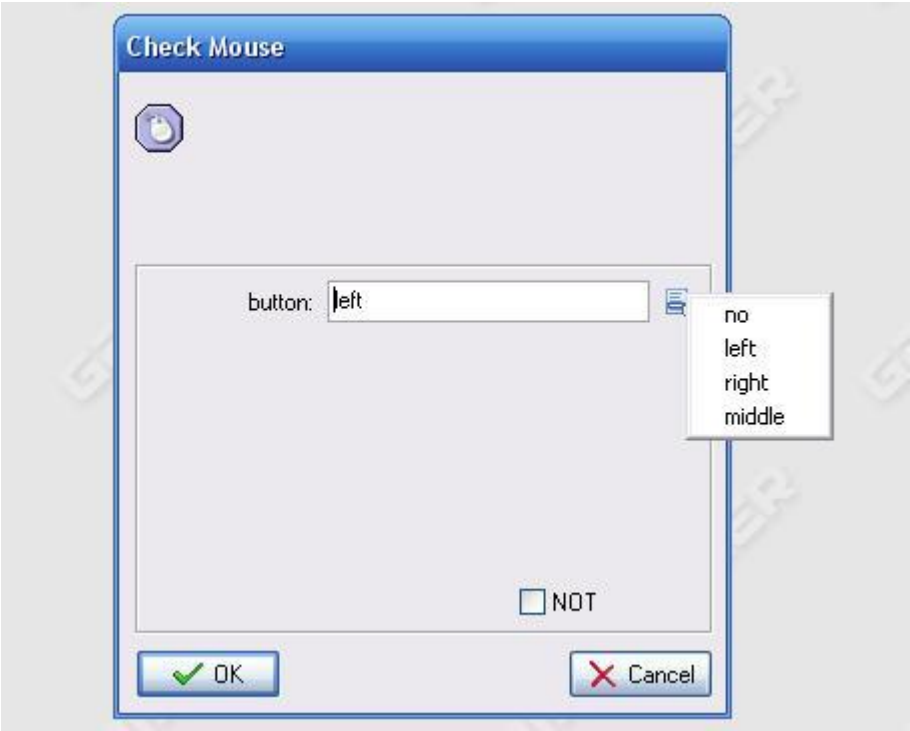
Εικόνα 1.73

Test Expression : Εδώ εισάγουμε μια «έκφραση», όπου ελέγχουμε εάν είναι σωστή, ώστε να εκτελεστούν οι παρακάτω εντολές που θα εισάγουμε, αλλιώς δεν εκτελούνται.



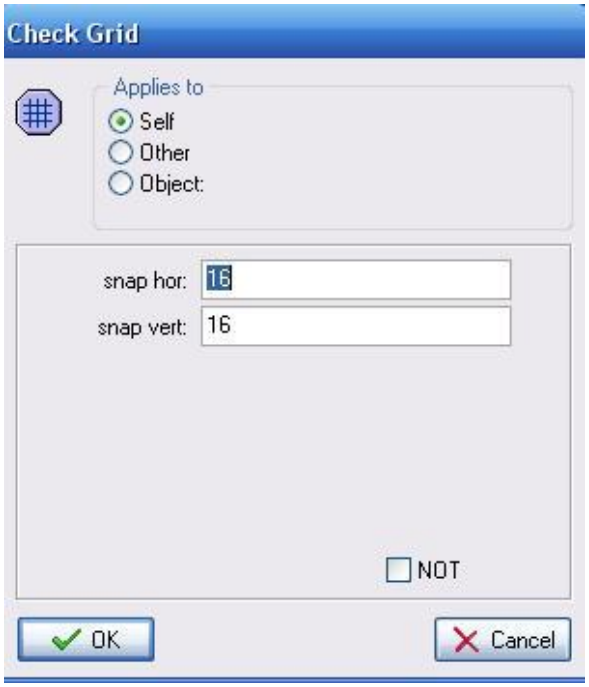
Εικόνα 1.74

Check Mouse : Αυτό το Action επιστρέφει true όταν το πλήκτρο του ποντικιού που έχουμε επιλέξει έχει πατηθεί.



Εικόνα 1.75

Check Grid : Εδώ επιλέγουμε μια περιοχή του δωματίου μας, θέτοντας το ανάλογο ύψος και πλάτος που θέλουμε, και μας επιστρέφεται true, εάν και όταν το αντικείμενό μας βρίσκεται μέσα σε αυτήν.

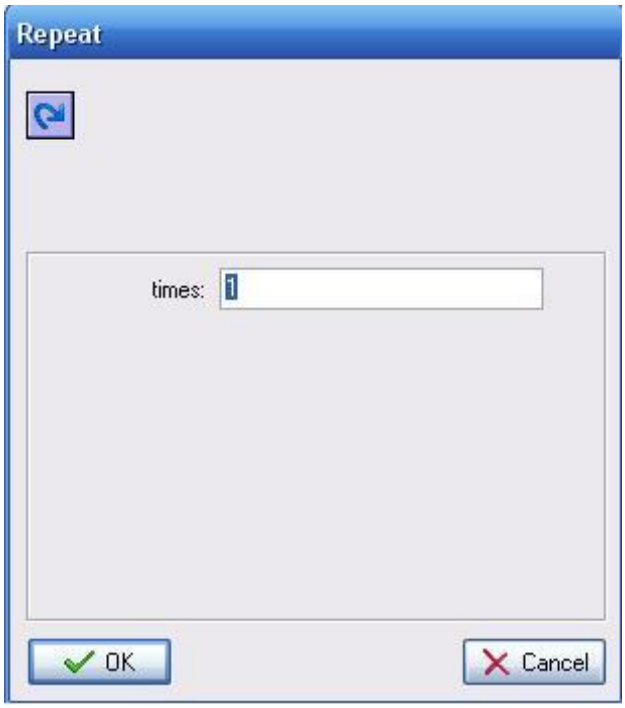


Εικόνα 1.76

Start και Stop Block : Στην ουσία, χρησιμεύουν ως «παρενθέσεις», μέσα στις οποίες τοποθετούμε ένα σύνολο από εντολές οι οποίες θέλουμε να εκτελεστούν ως «πακέτο» , όλες !

Else : Εάν κάνουμε κάποια ερώτηση μέσω κάποιου action και μας δώσει ως απάντηση false, δεν θα εκτελεστούν οι εντολές που είχαμε ορίσει, αλλά οι εντολές που θα ορίσουμε κάτω από το Else.

Repeat : Με το Repeat, επαναλαμβάνεται το Action που ακολουθεί (ή το πακέτο εντολών με τη χρήση των Blocks) όσες φορές το ορίσουμε εμείς .

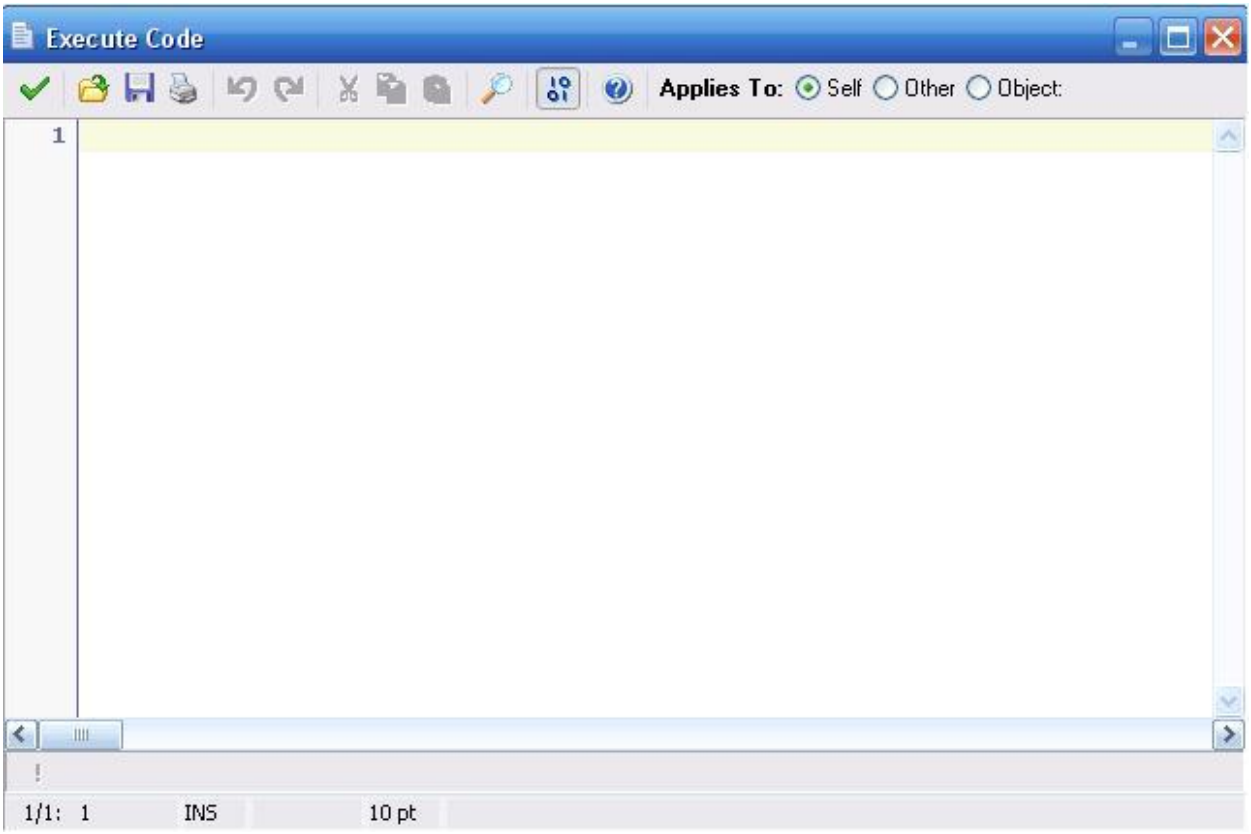


Εικόνα 1.77

Exit Event : Με αυτό το Action, δίνουμε εντολή να βγούμε από το τρέχων Event και να μην συνεχιστούν τα παρακάτω Actions. Για παράδειγμα αυτό μπορούμε να το χρησιμοποιήσουμε σε κάποια ερώτηση, όπου εάν δεν επαληθεύεται να μην συνεχίσουμε παρακάτω.

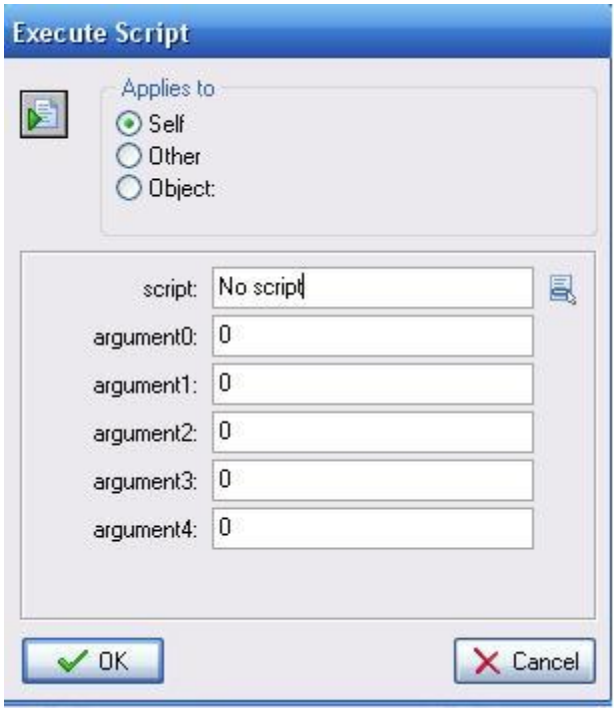
Call Event : Με αυτό το Action, του λέμε να «καλέσει» από την αρχή το Event

Execute Code : Εάν θέλουμε να γράψουμε και να εκτελέσουμε ένα μικρό κομμάτι κώδικα, με τον οποίο θα εκτελεστούν εντολές οι οποίες δεν αντιστοιχούν σε κάποιο από τα υπάρχον Actions, τότε αυτό γίνεται με αυτό το Action.



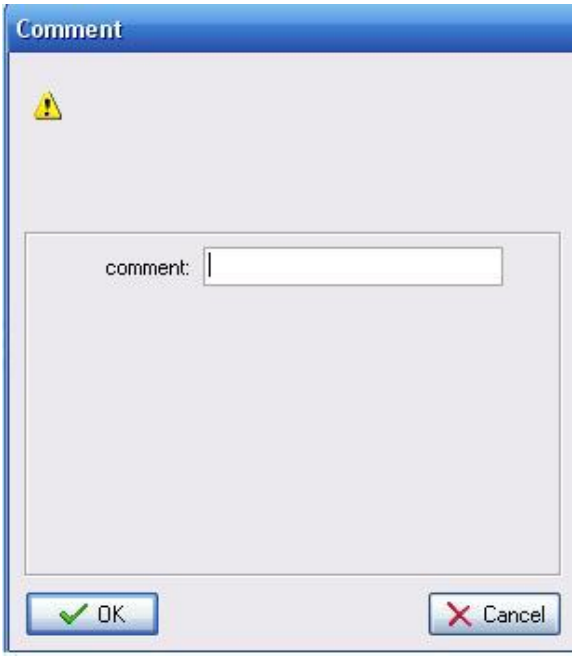
Εικόνα 1.78

Εάν το κομμάτι του κώδικα που θέλουμε να εκτελεστεί είναι μεγάλο, τότε δημιουργούμε και αποθηκεύουμε ένα Script, το οποίο το καλούμε με το Action «Execute Script» .



Εικόνα 1.79

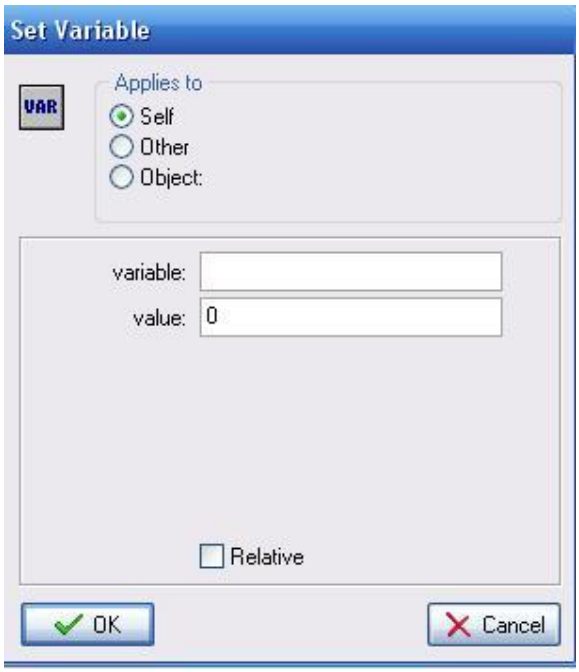
Comment : Όσοι έχουν ασχοληθεί με τον προγραμματισμό, σε οποιαδήποτε γλώσσα προγραμματισμού, θα ξέρουν πως ένα πάρα πολύ χρήσιμο για τον προγραμματιστή κομμάτι, είναι τα σχόλια. Οι εκφράσεις δηλαδή, οι οποίες δεν εκτελούνται, ούτε λαμβάνονται υπόψη από τον compiler, αλλά είναι πάρα πολύ χρήσιμες για αυτόν που θα ανατρέξει στον κώδικα. Βοηθάνε ώστε ο κώδικας να γίνεται πιο κατανοητός αλλά και εύκολα επαναπροσβάσιμος και επανασυντάξιμος .



Εικόνα 1.80

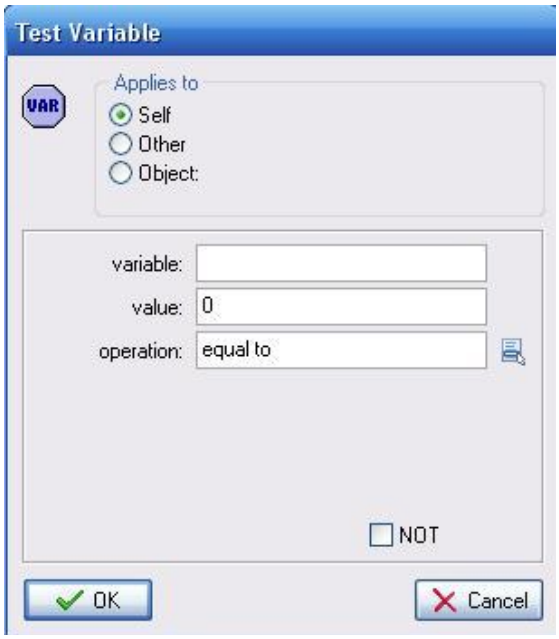
Ακολουθούν τρία(3) Actions που έχουν να κάνουν με τις μεταβλητές, για τις οποίες θα μιλήσουμε και θα εξηγήσουμε στην πορεία.

Set Variable : Με αυτό το Action ορίζουμε μια μεταβλητή, και θέτουμε την αρχική της τιμή .



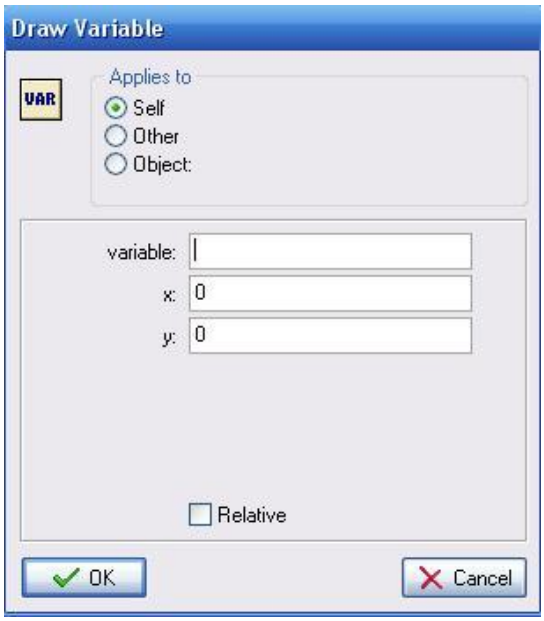
Εικόνα 1.81

Test Variable : Εδώ ελέγχουμε την τιμή μιας μεταβλητής , εάν ισούται με κάποιον βαθμό ή εάν είναι μικρότερη ή μεγαλύτερη από κάποιον βαθμό, ώστε να πράξουμε ανάλογα στην πορεία.



Εικόνα 1.82

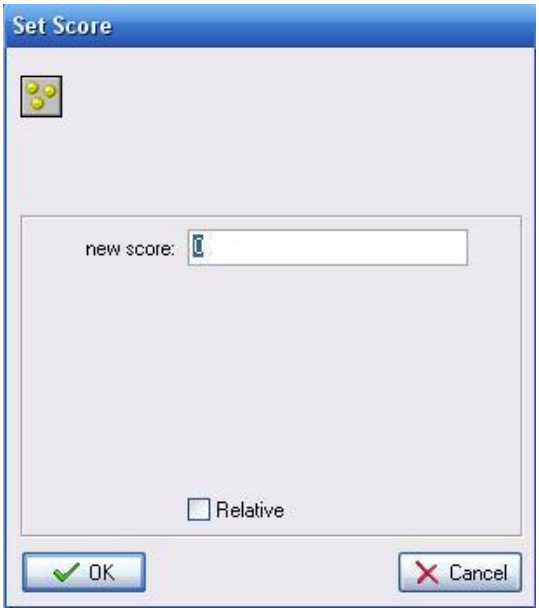
Draw Variable : Και εδώ δίνουμε την εντολή να εμφανιστεί μέσα στο παιχνίδι μας, σε κάποιο σημείο του δωματίου που θα ορίσουμε, η τιμή κάποιας μεταβλητής .



Εικόνα 1.83

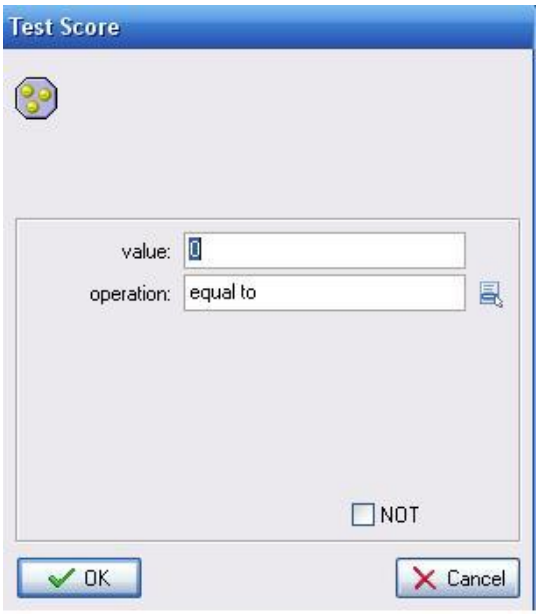
ΥΠΟΚΕΦΑΛΑΙΟ 1.2.2.5
SCORE ACTIONS

Set Score : Στο Game Maker υπάρχει προεγκατεστημένη μια μεταβλητή που έχει να κάνει με το Score. Με αυτό το Action μπορούμε να μεταβάλλουμε την τιμή της. Εάν επιλέξουμε το relative, μπορούμε να προσθέσουμε ή να αφαιρέσουμε την τιμή που δίνουμε, από την ήδη αποθηκευμένη τιμή που έχει το Score .



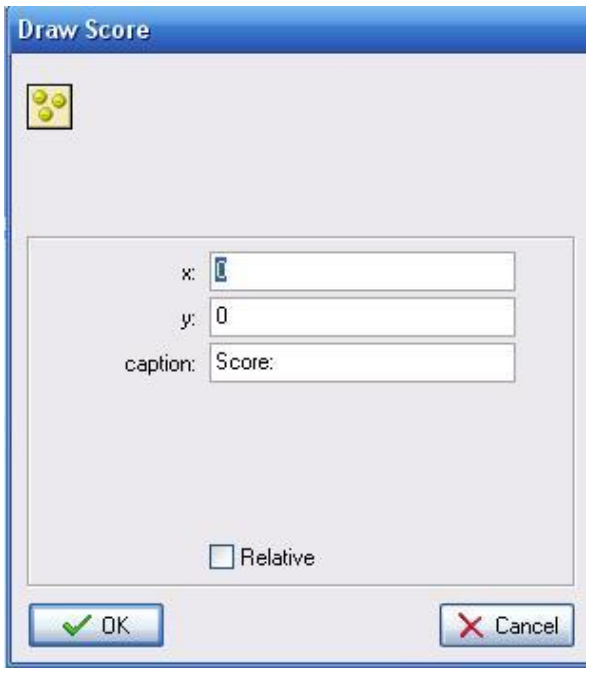
Εικόνα 1.84

Test Score : Όπως και με παρόμοια Actions, έτσι κι εδώ ελέγχουμε την τιμή του Score, εάν είναι ίση με την τιμή που δίνουμε, εάν είναι μεγαλύτερη ή μικρότερη.



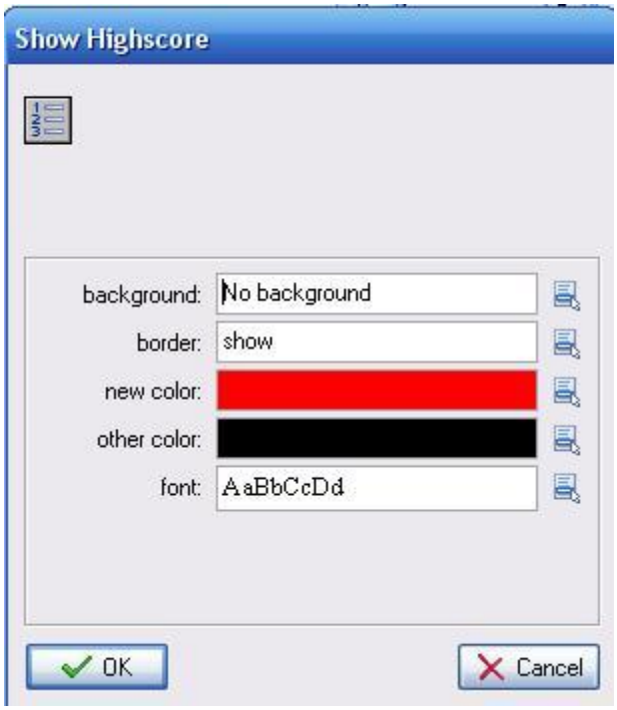
Εικόνα 1.85

Draw Score : Κι εδώ, μέσα από το Drawing Event πάντα, μπορούμε να απεικονίσουμε την τιμή του Score, μέσα στο δωμάτιο, στο σημείο που επιθυμούμε .



Εικόνα 1.86

Show High score : Σε κάθε παιχνίδι που δημιουργούμε και παίζουμε με το Game Maker, οι δέκα(10) καλύτερες επιδόσεις (τα δέκα υψηλότερα Scores) διατηρούνται. Έτσι με αυτό το Action μπορούμε ανά πάσα στιγμή να τα εμφανίσουμε, με τις ανάλογες επιλογές (χρώμα φόντου, περίγραμμα κ.τ.λ.) . Εάν ο παίχτης βρίσκεται στο «Top-10», τότε μπορεί να εισάγει και το όνομά του και με τις ανάλογες εντολές να το αποθηκεύσει.



Εικόνα 1.87

Clear High Score : Αυτό το Action απλά διαγράφει το ήδη υπάρχον High Score.

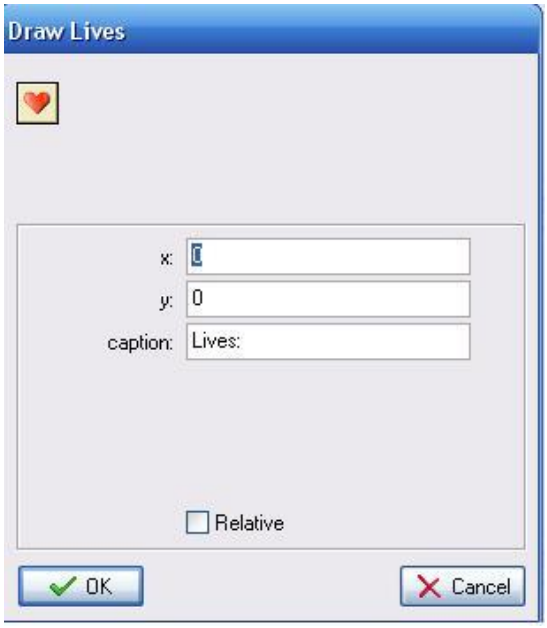
Set Lives : Εδώ, επεμβαίνουμε στο σύστημα των ζωών που υπάρχει στο Game Maker. Θέτουμε τον αριθμό αυτών στην αρχή του παιχνιδιού, και τον αυξομειώνουμε κατά την διάρκειά του (προσοχή , πρέπει να επιλέξουμε το *relative*) . Εάν οι ζωές μηδενιστούν, τότε ενεργοποιείται και το ανάλογο Event, που συνήθως μπορεί να σταματάει επιτόπου το παιχνίδι και να εμφανίζει για παράδειγμα ένα **Game Over** .

Εικόνα 1.88

Test Lives : Όπως και με τα υπόλοιπα «Test» actions , έτσι και με αυτό , ελέγχουμε την τιμή της μεταβλητής που αντιστοιχεί στις ζωές, εάν είναι ίση, μικρότερη ή μεγαλύτερη από έναν αριθμό που θα δώσουμε εμείς. Έτσι για παράδειγμα ελέγχουμε και αν έχουν μηδενιστεί οι ζωές, για να πράξουμε ανάλογα.

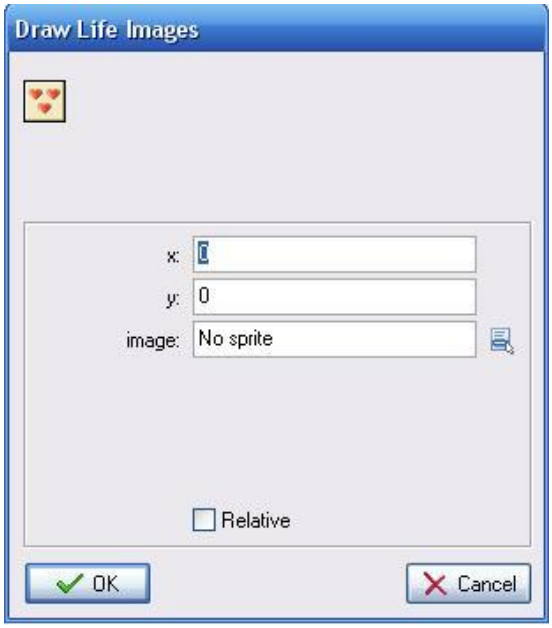
Εικόνα 1.89

Draw Lives : Και εδώ, με αυτό το Action, μπορούμε να απεικονίσουμε μέσα στο παιχνίδι τον αριθμό των ζωών που μας μένουν, με τη βοήθεια ενός *Draw Event*. Και πάλει με τη χρήση του άξονα x-y θέτουμε σε ποιο σημείο θέλουμε να εμφανιστεί.



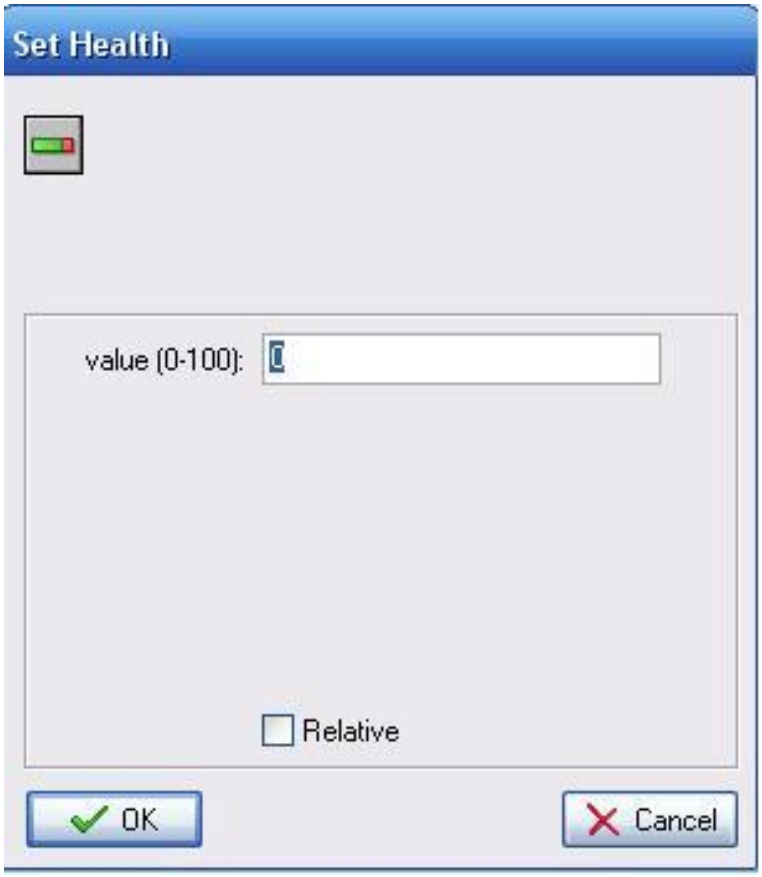
Εικόνα 1.90

Draw life images : Με αυτό το Action, αντί να εμφανίζουμε στο παιχνίδι τον αριθμό των ζωών, με έναν απλό αριθμό, μπορούμε να το δείξουμε εικονικά, εμφανίζοντας κάποια εικόνα (*Sprite*) τόσες φορές, όσες είναι και οι ζωές, για παράδειγμα τρεις καρδούλες, ή τρία αστεράκια (εάν οι ζωές που μας απομένουν είναι τρεις) .



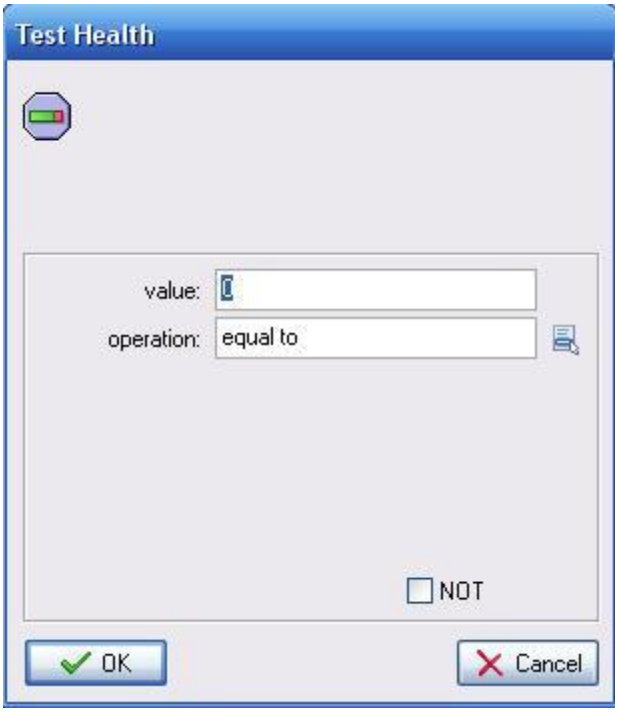
Εικόνα 1.91

Set health : Όπως προείπαμε, το Game Maker, έχει κάποιες μεταβλητές εξ αρχής αποθηκευμένες στις βιβλιοθήκες του. Μία από αυτές είναι και η *Health* . Ή όπως την συναντάμε συχνά σε πολλά παιχνίδια, η *Ενέργεια*. Με αυτό το Action, θέτουμε την τιμή αυτής της μεταβλητής, όπου 100 είναι το μέγιστο και 0 το ελάχιστο. Όταν πάρει την τιμή 0, τότε η λογική λέει πως θα ενεργοποιηθεί και το ανάλογο Event, ανάλογα με το τι θέλουμε να λάβει χώρα στην πορεία στο παιχνίδι μας, ή πιθανόν να μειώνεται ο αριθμός των *ζωών* κατά 1 . Προσέχουμε όμως, εάν με αυτό το Action θέλουμε να αυξομειώσουμε την τιμή αυτής της μεταβλητής, και όχι να θέσουμε την αρχική της τιμή, να «τσεκάρουμε» την επιλογή *Relative* .



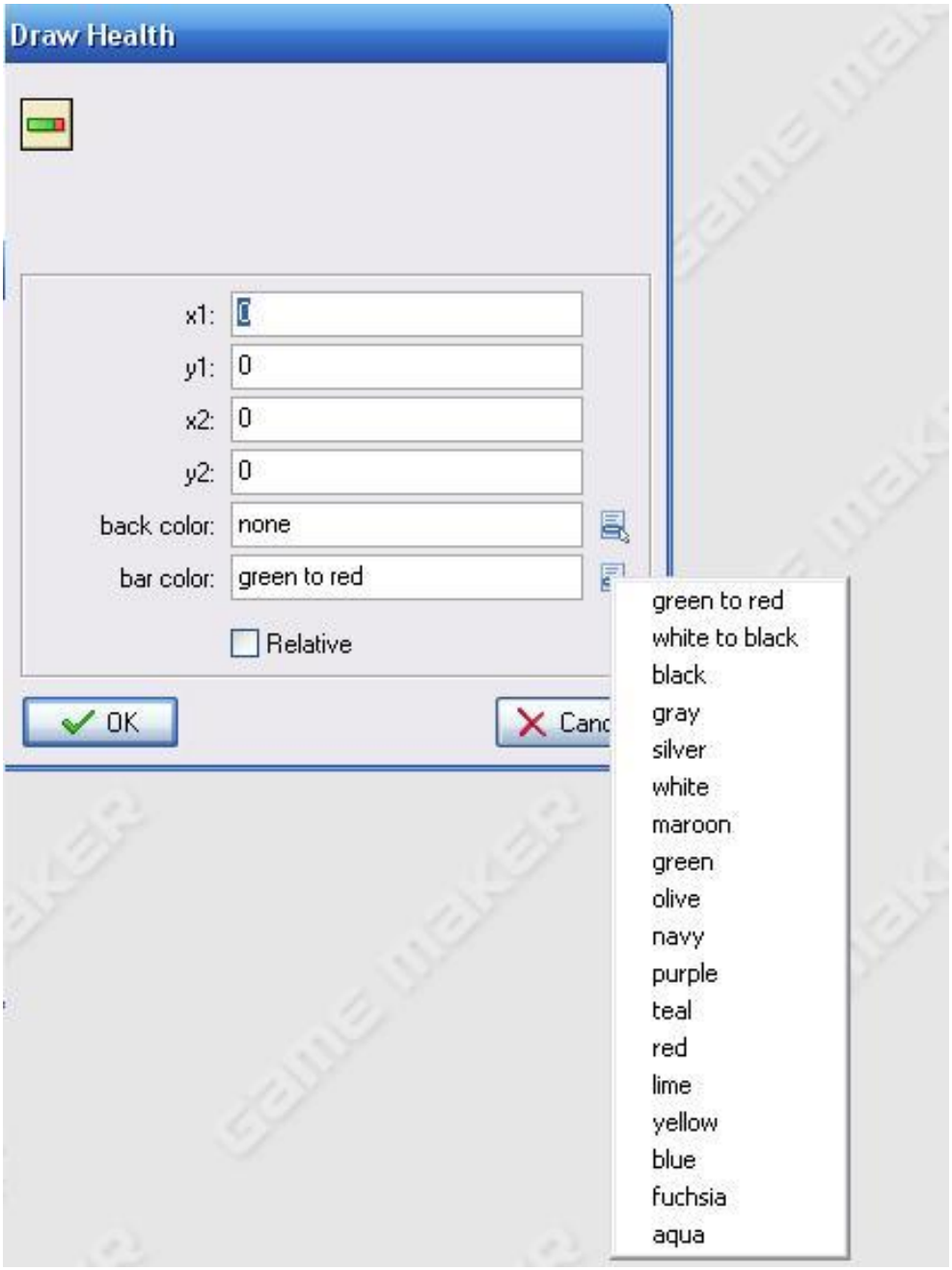
Εικόνα 1.92

Test health : Όπως και με παρόμοια Actions, έτσι κι εδώ ελέγχουμε την τιμή του health, εάν είναι ίση με την τιμή που δίνουμε, εάν είναι μεγαλύτερη ή μικρότερη, και πράττουμε ανάλογα.



Εικόνα 1.93

Draw health : Ακόμη ένα Action που έχουμε συναντήσει πολλές φορές, που μας δίνει την δυνατότητα να απεικονίσουμε σε κάποιο σημείο του δωματίου, το health. Με την μόνη διαφορά ότι το health απεικονίζεται ως μία μπάρα, επομένως και πρέπει να ορίσουμε και τα χρώματα αυτής.


Draw Health
x1:
y1:
0
x2:
0
y2:
0
back color:
none
bar color:
green to red
Relative
OK
green to red
white to black
black
gray
silver
white
maroon
green
olive
navy
purple
teal
red
lime
yellow
blue
fuchsia
aqua

Εικόνα 1.94

Score caption : Εκτός από τα σημεία που επιλέγουμε να εμφανίζονται το Score,οι ζωές, η ενέργεια και τα λοιπά, στο πάνω μέρος του παραθύρου του παιχνιδιού, αυτόματα εμφανίζονται και τα εξής: Το όνομα του δωματίου (room), το Score Με αυτό το Action μπορούμε να επιλέξουμε εμείς τι θέλουμε να εμφανίζεται σε αυτό το σημείο .

Εικόνα 1.95

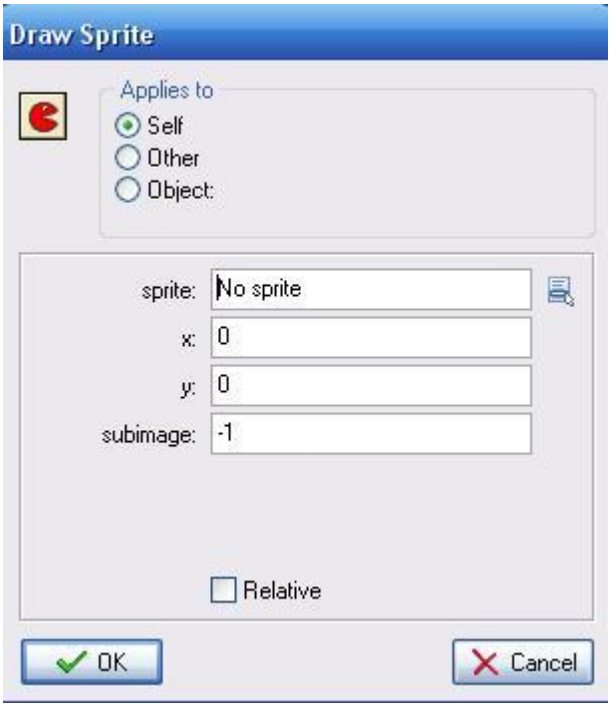
ΥΠΟΚΕΦΑΛΑΙΟ 1.2.2.6
EXTRA ACTIONS

Από αυτήν την ομάδα με Actions, θα αναφέρουμε επιγραμματικά τα σημαντικότερα και αυτά που ίσως ποτέ χρησιμοποιήσουμε. Πιθανόν, κατά τη διάρκεια του παιχνιδιού μας να θέλουμε να αναπαράγουμε κάποιο μουσικό CD, το οποίο υπάρχει ήδη στο CD Driver μας. Με τα ανάλογα Actions, ξεκινάμε την αναπαραγωγή αυτού, την σταματάμε, ή πριν κάνουμε κάτι από αυτά, ελέγχουμε εάν υπάρχει κάποιο CD στο CD Driver μας ή ελέγχουμε εάν βρίσκεται ήδη σε αναπαραγωγή. Επίσης, ένα αρκετά χρήσιμο Action αυτής της ομάδας, είναι αυτό που μας επιτρέπει να αλλάξουμε το εικονίδιο του κέρσορα μας, σε περίπτωση που ο κέρσορας έχει ενεργή δράση στο παιχνίδι.

Εικόνα 1.96

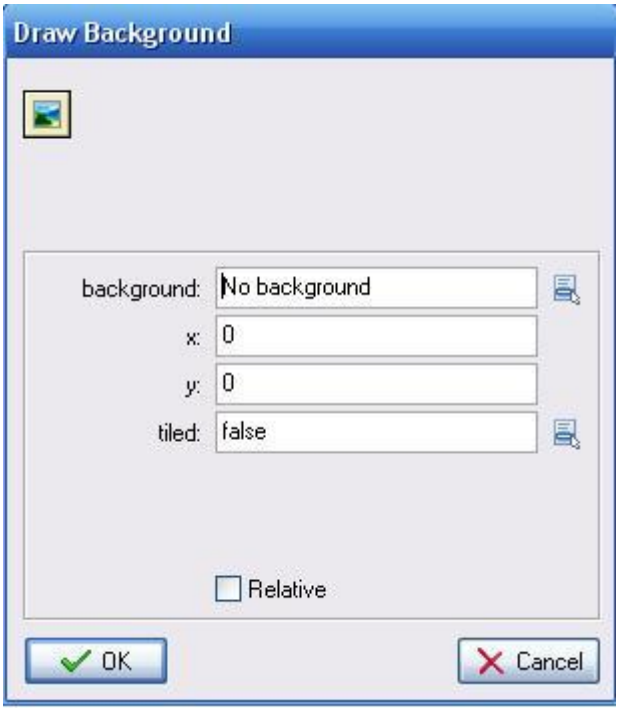
ΥΠΟΚΕΦΑΛΑΙΟ 1.2.2.7
DRAW ACTIONS

Draw Sprite : Απλά για να απεικονίσουμε κάποιο Sprite, μια εικόνα δηλαδή, στο παιχνίδι μας. Εάν το Sprite αποτελείται από περισσότερες από μία εικόνες, δηλαδή είναι μια κινούμενη εικόνα, για να τις εμφανίσει όλες σειριακά, βάζουμε την τιμή *-1* στο *subimage* .



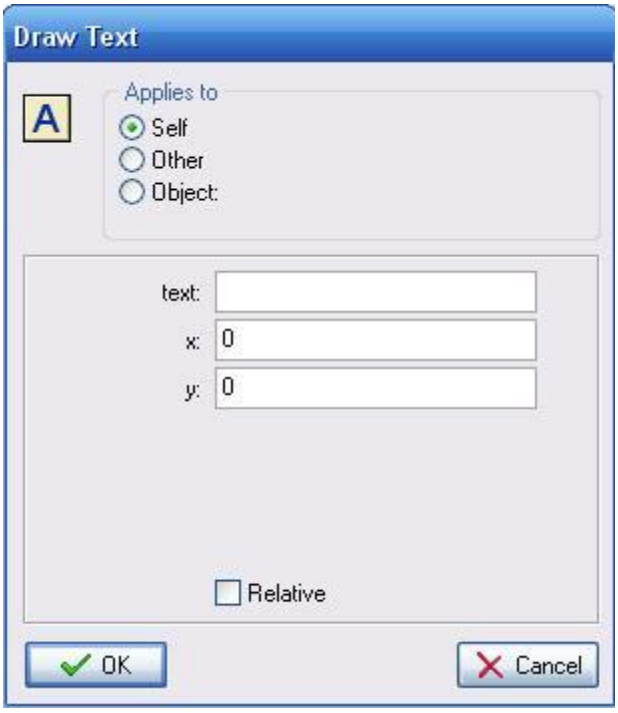
Εικόνα 1.97

Draw Background : Υποδεικνύουμε την εικόνα που θα έχει ως background το δωμάτιο του παιχνιδιού, καθώς και την μορφή που θα έχει η εικόνα αυτή (αν θα καλύπτει όλο το δωμάτιο υπό μορφή πλακιδίων) .



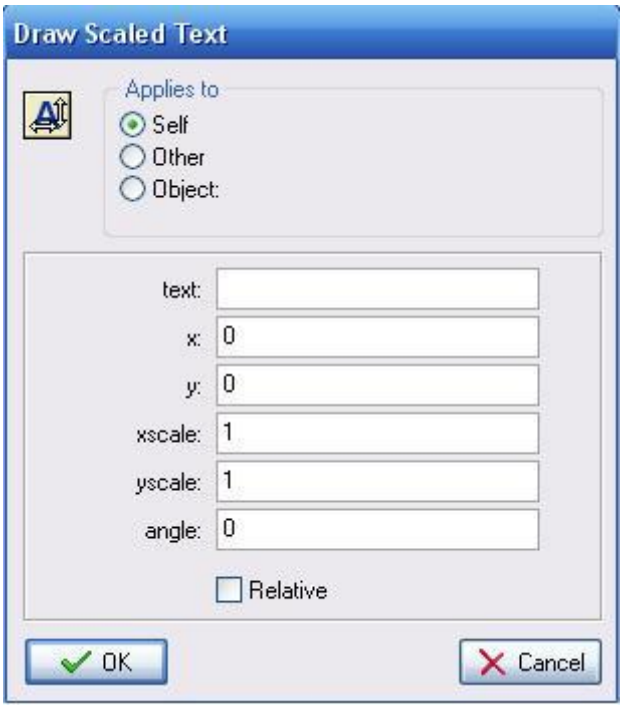
Εικόνα 1.98

Draw Text : Απλά για να εμφανίσουμε κάποιο κείμενο σε κάποιο σημείο του δωματίου. Μπορεί να είναι κάποιο ενημερωτικό κείμενο, ή για παράδειγμα το score ή κάποια άλλη μεταβλητή και τα λοιπά. Χρησιμοποιώντας το σύμβολο # αλλάζουμε γραμμή στο κείμενο που θέλουμε να απεικονίσουμε, επομένως με ένα μόνο Action, μπορούμε να εμφανίσουμε κείμενο πολλών γραμμών. Εάν το κείμενο που εισάγουμε το περικλύσουμε μέσα σε ‘ ‘ ή “ “ τότε το game maker δεν το εμφανίζει σαν απλό κείμενο αλλά το βλέπει ως έκφραση και εμφανίζει τα ανάλογα αποτελέσματα, για παράδειγμα όταν θέλουμε να εμφανίσουμε την τιμή κάποιας μεταβλητής.



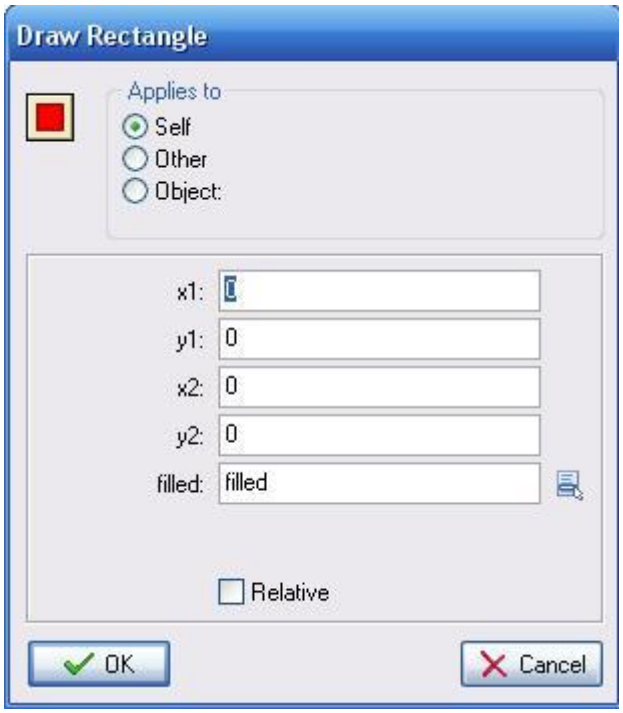
Εικόνα 1.99

Draw Scaled Text : Αυτό το Action είναι ακριβώς το ίδιο με το *Draw Text* Action, με τη διαφορά ότι μας δίνεται η δυνατότητα να τροποποιήσουμε και να επιλέξουμε το μέγεθος και τη μορφή του κειμένου που θέλουμε να εμφανίσουμε (scale και angle).



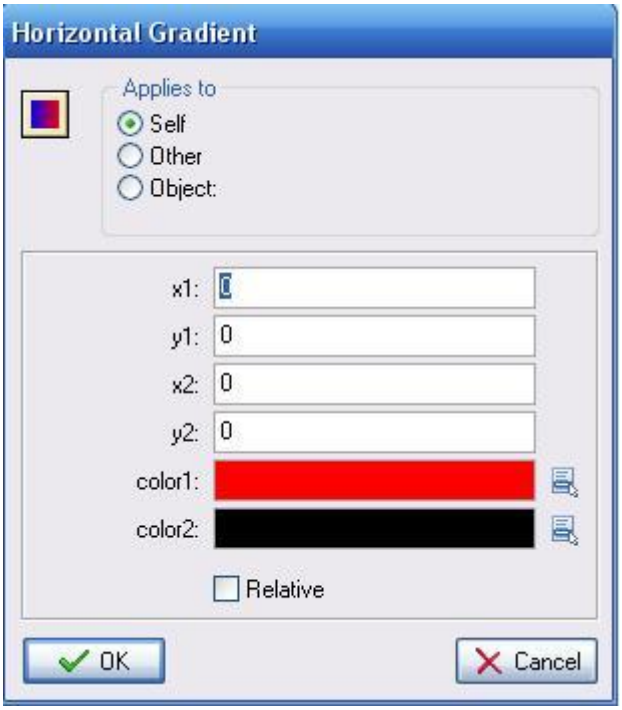
Εικόνα 1.100

Draw Rectangle : Πολύ απλά απεικόνιση ενός τετραγώνου, στο σημείο του δωματίου που θα δώσουμε. Εάν επιλέξουμε το *relative* , τότε οι συντεταγμένες μετράνε από το σημείο που βρίσκεται το αντικείμενό μας και όχι από την «αρχή των αξόνων» .



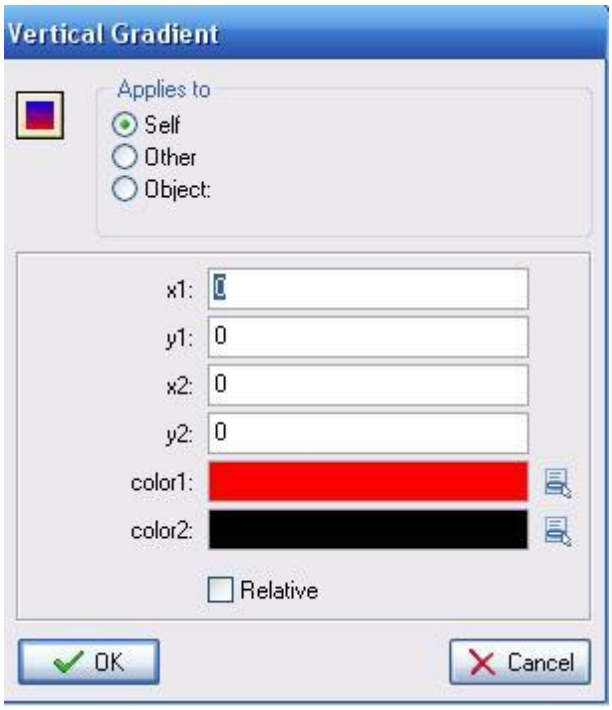
Εικόνα 1.101

Horizontal Gradient : Είναι παρόμοιο με το *Draw Rectangle* Action, με την διαφορά ότι το τετράγωνο που θα «ζωγραφίσουμε», θα έχει δύο(2) διαφορετικά χρώματα, τα οποία θα επιλέξουμε εμείς, και η εναλλαγή των οποίων γίνεται οριζόντια, από τα αριστερά προς τα δεξιά (οριζόντια).



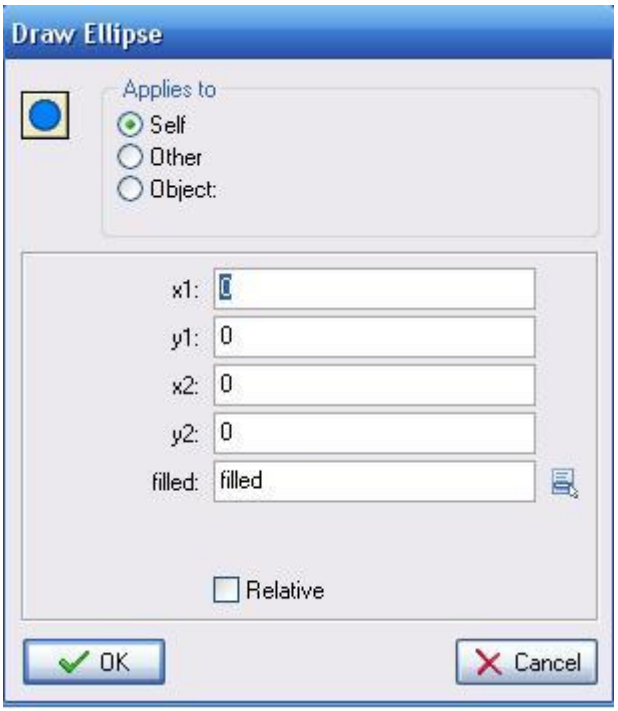
Εικόνα 1.102

Vertical Gradient : Ακριβώς το ίδιο Action με το *Horizontal Gradient* , μόνο που εδώ η εναλλαγή των χρωμάτων γίνεται από πάνω προς τα κάτω (κάθετα) .



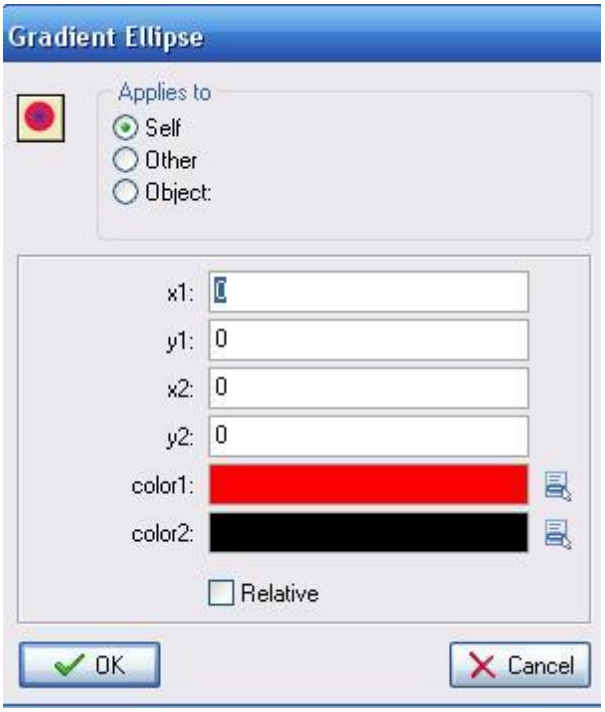
Εικόνα 1.103

Draw Ellipse : Με αυτό το Action, μπορούμε να απεικονίσουμε στο παιχνίδι μας μια έλλειψη, όπως δηλαδή για παράδειγμα έναν κύκλο, στο σημείο που εμείς θα προσδιορίσουμε.



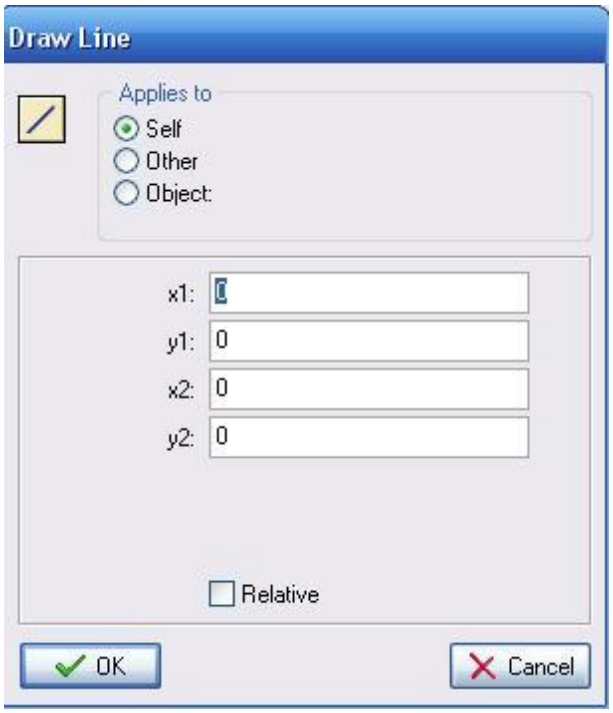
Εικόνα 1.104

Gradient Ellipse : Όπως και με τα Action για την απεικόνιση τετραγώνων με δύο διαφορετικά χρώματα, έτσι και με αυτό, απεικονίζουμε μία έλλειψη η οποία θα έχει άλλο χρώμα στο κέντρο του, στο εσωτερικό του, και διαφορετικό χρώμα εξωτερικά, στο περίγραμμά του .



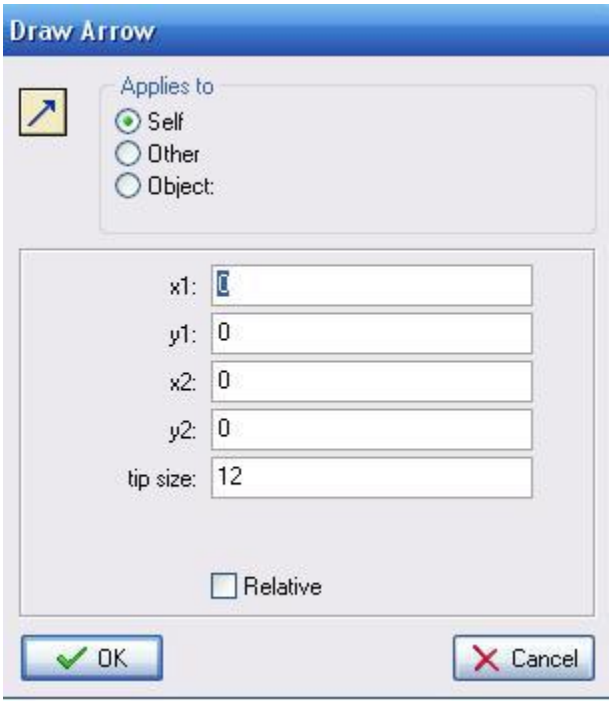
Εικόνα 1.105

Draw Line : Εμείς απλά δηλώνουμε τις συντεταγμένες των δύο άκρων που θέλουμε να έχει μια ευθεία γραμμή, την οποία και θα απεικονίσουμε στο δωμάτιο του παιχνιδιού μας.



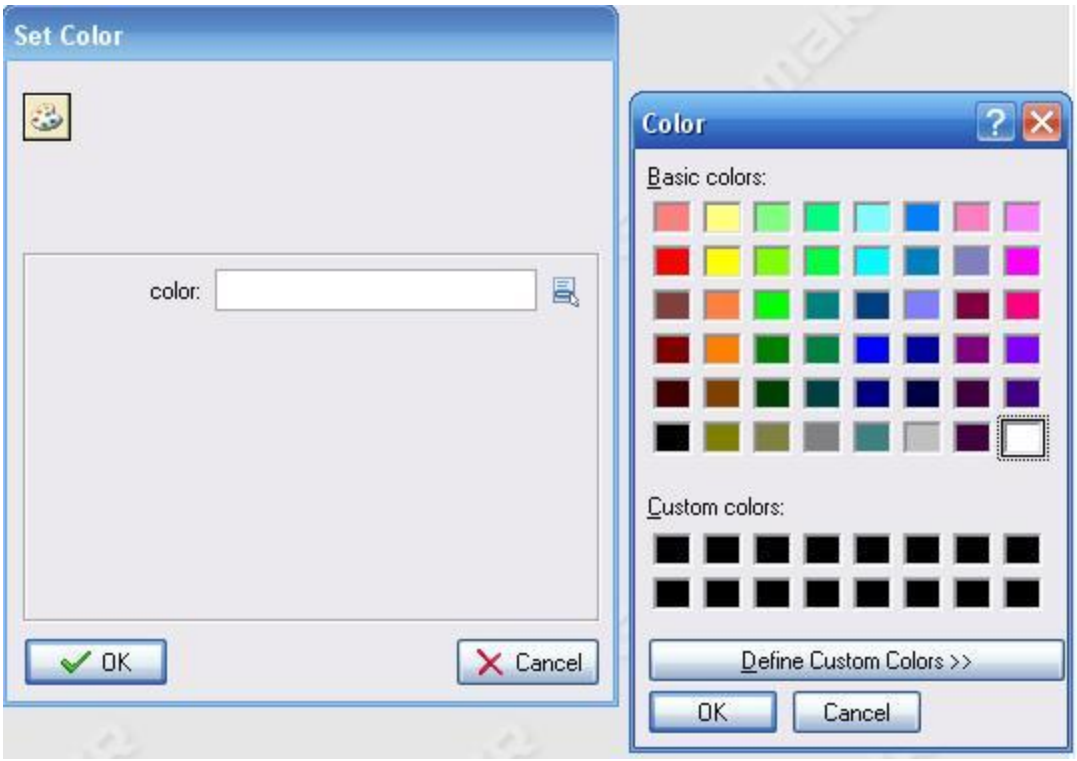
Εικόνα 1.106

 Draw Arrow : Το ίδιο ακριβώς με το *Draw Line* Action, μόνο που με αυτό αντί να απεικονίσουμε μια απλή ευθεία γραμμή, ζητάμε να απεικονίσει μια ευθεία γραμμή με ένα βέλος στην μία της άκρη, του οποίου προκαθορίζουμε το μέγεθος.



Εικόνα 1.107

 **Set Color** : Με αυτό το Action προσδιορίζουμε το χρώμα που θα έχουν να γεωμετρικά σχήματα που θα εισάγουμε στο παιχνίδι μας, αλλά δεν αφορά το χρώμα των Sprites ή των Background.



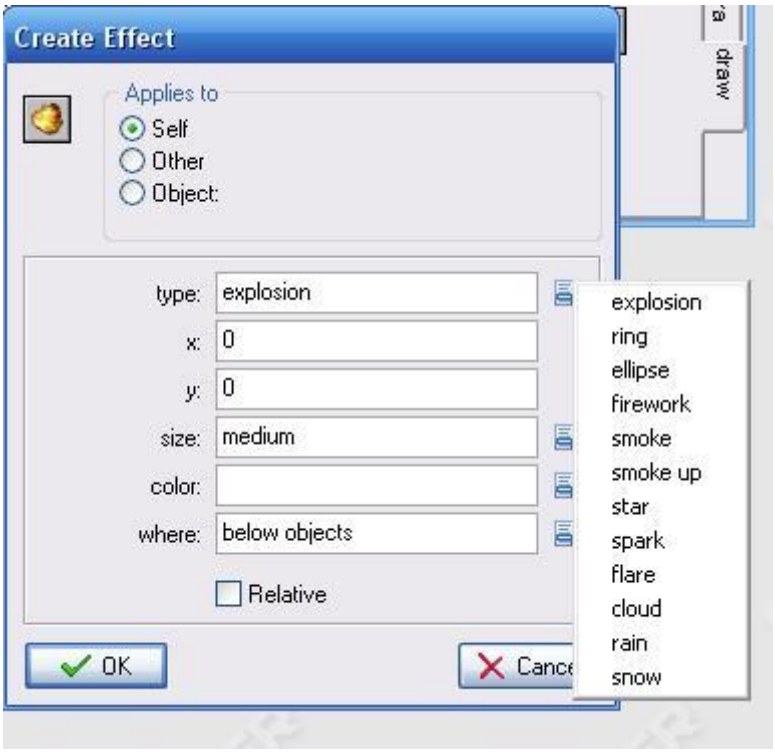
Εικόνα 1.108

Set Full Screen : Αλλάζει τον τρόπο εμφάνισης του παιχνιδιού (mode), αν θα καλύπτει δηλαδή ολόκληρη την οθόνη (full screen) ή θα εμφανίζεται σε μορφή παραθύρου (windowed) .

Εικόνα 1.109

Take Snapshot : Με αυτό το Action ζητάμε να «τραβηχτεί μια φωτογραφία» της οθόνης , δηλαδή ένα screenshot όπως το γνωρίζουμε όλοι. Το αρχείο αυτό είναι της μορφής *.bmp* , του δίνουμε εμείς το όνομα που θέλουμε να έχει και τον φάκελο που θέλουμε να αποθηκευτεί.

Create Effect : Με αυτό το Action δημιουργούμε πολύ απλά και εύκολα κάποιο ή κάποια εφέ κατά τη διάρκεια του παιχνιδιού μας, όπως για παράδειγμα κάποια μορφή έκρηξης, βροχή και τα λοιπά. Εμείς απλά επιλέγουμε τον τύπο του εφέ που θέλουμε να δημιουργήσουμε, το σημείο που θέλουμε να λάβει χώρα το εφέ, το μέγεθός του, το χρώμα που θέλουμε να έχει και εάν θέλουμε να εμφανιστεί πίσω ή μπροστά από το αντικείμενό στο οποίο ενεργεί αυτό το Action. Όπως και σε προηγούμενα Actions, έτσι και σε αυτό, το *relative* το επιλέγουμε, εάν θέλουμε οι τιμές του x και του y που θα δώσουμε να αρχίσουν να μετράνε από το σημείο που βρίσκεται το αντικείμενό μας και όχι από την αρχή του δωματίου του παιχνιδιού.



Εικόνα 1.110

ΥΠΟΚΕΦΑΛΑΙΟ 1.3

GML

Όπως έχουμε ήδη αναφέρει, μπορούμε να δώσουμε τα ανάλογα «Actions» και «Events», εκτός από τον *Drag And Drop* τρόπο, με κώδικα. Ο γλώσσα που δέχεται και μπορεί να τρέξει το Game Maker κατεξοχήν είναι η *Game Maker Language* ή *GML* όπως θα την ονομάζουμε στην πορεία. Το συντακτικό μέρος της GML είναι πολύ απλό και εύκολο, το οποίο έχει αρκετά κοινά στοιχεία με τις ευρέως γνωστότερες γλώσσες προγραμματισμού, όπως είναι η C++ και τα λοιπά. Σε αυτό το κεφάλαιο θα αναφέρουμε τα πιο σημαντικά στοιχεία της GML, αυτά τα οποία θα μας είναι βασικά για μια «αρχάρια» σύνταξή της.

Ας ξεκινήσουμε από τα βασικά. Πρώτα από όλα, Ο κώδικας που θα γράψουμε πρέπει να βρίσκεται ανάμεσα σε **{ }** και δεύτερον πολύ σημαντικό, κάθε γραμμή κώδικα που γράφουμε πρέπει να έχει στο τέλος ένα ελληνικό ερωτηματικό **;** ή όπως είναι γνωστό, ένα **semicolon** . Παραδείγματος χάριν :

```
}
  <statement>;
  <statement>;
  ...
}
```

Το Game Maker έχει προ- εγκατεστημένες στις βιβλιοθήκες του κάποιες βασικές **Functions**. Χωρίς αυτό όμως να σημαίνει πως δεν μπορούμε να δημιουργήσουμε και κάποιες δικές μας, ανάλογα με τις ανάγκες του παιχνιδιού που θέλουμε να δημιουργήσουμε. Στις ήδη «**build in** » functions μπορούμε να βρούμε μια πληθώρα από συναρτήσεις, άλλες βασικές και πολύ χρήσιμες, άλλες που επεμβαίνουν πολύ στην λεπτομέρεια και τα λοιπά. Θα βρούμε συναρτήσεις οι οποίες υπάρχουν ήδη σε κάποια Events ή Actions και μπορούμε να τις χρησιμοποιήσουμε με τον Drag And Drop τρόπο, όπως για παράδειγμα η **room_next(numb)** η οποία δίνει εντολή να μεταβούμε στο επόμενο δωμάτιο του παιχνιδιού. Θα βρούμε όμως και συναρτήσεις, από τις οποίες μπορούμε να έχουμε το επιθυμητό αποτέλεσμα μόνο με τη χρήση Script και την συγγραφή αυτών με χρήση κώδικα GML , όπως για παράδειγμα η **move_towards_point(x,y,spd)** η οποία δίνει την εντολή να μεταφερθεί το αντικείμενό μας στο σημείο (x,y) με ταχύτητα spd .

Αλφαβητικά και ονομαστικά, οι προ-εγκατεστημένες συναρτήσεις του Game Maker είναι οι παρακάτω :

ΠΙΝΑΚΑΣ 1.1

A

- Abs
- Arccos
- Arcsin
- Arctan
- Arctan2
- Audio emitter falloff
- Audio falloff set model
- Audio play sound at

B

- Background create gradient
- Background get texture

C

- Ceil
- Choose
- Chr
- Clipboard has text
- Collision point
- Collision rectangle
- Cos

D

- D3d draw block
- D3d draw cone
- D3d draw cylinder
- D3d draw ellipsoid
- D3d draw floor
- D3d draw wall
- D3d end
- D3d light

D cont.

- D3d transform set identity
- D3d transform set rotation x
- D3d transform set rotation y
- D3d transform set rotation z
- D3d transform set scaling
- D3d transform set translation
- D3d vertex
- D3d vertex color
- D3d vertex normal
- D3d vertex normal color
- D3d vertex normal texture
- D3d vertex normal texture color
- D3d vertex texture
- D3d vertex texture color
- Degtorad
- Display mouse get x
- Display mouse get y
- Display mouse set
- Distance to object
- Distance to point

D cont.

- Draw text transformed color
- Draw triangle
- Draw triangle color
- Draw vertex
- Draw vertex color
- Ds list create
- Ds list destroy

E

- Effect clear
- Effect create above
- Effect create below
- Event perform
- Event perform object
- Event user
- Execute program
- Execute shell
- Exp

F

- File attributes
- File bin close
- File bin open
- File bin position
- File bin read

- define direction
- D3d light define point
- D3d light enable
- D3d model block
- D3d model clear
- D3d model cone
- D3d model create
- D3d model cylinder
- D3d model destroy
- D3d model draw
- D3d model ellipsoid
- D3d model floor
- D3d model load
- D3d model primitive begin
- D3d model primitive end
- D3d model save
- D3d model vertex
- D3d model vertex color
- D3d model vertex normal
- D3d model vertex normal color
- D3d model vertex normal texture
- D3d model vertex normal texture color
- D3d model vertex texture
- Draw background
- Draw circle
- Draw circle color
- Draw clear
- Draw ellipse
- Draw ellipse color
- Draw healthbar
- Draw line
- Draw line color
- Draw line width
- Draw line width color
- Draw point
- Draw point color
- Draw primitive begin
- Draw primitive end
- Draw rectangle
- Draw rectangle color
- Draw roundrect
- Draw roundrect color
- Draw self
- Draw set alpha
- Draw set blend mode
- Draw set color
- Draw set font
- Draw set halign
- Draw set
- byte
- File bin rewrite
- File bin seek
- File bin size
- File bin write byte
- File exists
- File find close
- File find first
- File find next
- File text close
- File text eof
- File text eoln
- File text open append
- File text open read
- File text open write
- File text read real
- File text read string
- File text readln
- File text write real
- File text write string
- File text writeln
- Filename change ext
- Filename dir
- Filename drive
- Filename ext
- Filename name
- Filename

- D3d model vertex texture color
- D3d model wall
- D3d primitive begin
- D3d primitive begin texture
- D3d primitive end
- D3d set culling
- D3d set depth
- D3d set fog
- D3d set hidden
- D3d set lighting
- D3d set perspective
- D3d set projection ext
- D3d set projection ortho
- D3d set projection perspective
- D3d set shading
- D3d start
- D3d transform add rotation x
- D3d transform add scaling
- D3d transform add translation

valign

- Draw sprite
- Draw sprite ext
- Draw sprite part
- Draw sprite stretched
- Draw sprite stretched ext
- Draw sprite tiled ext
- Draw surface
- Draw surface ext
- Draw surface general
- Draw surface part
- Draw surface part ext
- Draw surface stretched
- Draw surface stretched ext
- Draw surface tiled
- Draw surface tiled ext
- Draw text color
- Draw text ext
- Draw text ext color
- Draw text ext transformed
- Draw text ext transformed color
- Draw text transformed

path

- Floor
- Frac

G

- GML Functions: Particle Destroyers
- Game end
- Game load
- Game restart
- Game save
- Get integer
- Get string

I

- Ini close
- Ini key delete
- Ini key exists
- Ini open
- Ini read real
- Ini read string
- Ini section delete
- Ini section exists
- Ini write real
- Ini write string

I

- Instance copy
- Instance create
- Instance deactivate all
- Instance destroy
- Instance exists
- Instance number
- Instance place
- Instance position
- Io clear
- Irandom
- Irandom range
- Is real
- Is string

K

- Keyboard check
- Keyboard check direct
- Keyboard check pressed
- Keyboard check released

L

- Lengthdir x
- Lengthdir y
- Ln
- Log10
- Log2
- Logn

M cont.

- Mplay init ipx
- Mplay init modem
- Mplay init serial
- Mplay init tcpip
- Mplay ipaddress
- Mplay message clear
- Mplay message count
- Mplay message id
- Mplay message name
- Mplay message player
- Mplay message receive
- Mplay message send
- Mplay message send guaranteed
- Mplay message value
- Mplay player find
- Mplay player id
- Mplay player name
- Mplay session create
- Mplay session end
- Mplay session find
- Mplay session join
- Mplay

S cont.

- Show message
- Show message ext
- Show question
- Sign
- Sin
- Splash show image
- Splash show text
- Splash show video
- Splash show web
- Sprite get texture
- Sqr
- Sqrt
- String (function)
- String char at
- String copy
- String delete
- String format
- String height
- String height ext
- String length
- String lower
- String pos
- String repeat
- String upper
- String width
- String width ext
- Surface copy
- Surface copy part
- Surface create
- Surface exists
- Surface free
- Surface get height
- Surface get texture

M

- Make color hsv
- Make color rgb
- Max
- Mean
- Median
- Merge color
- Message background
- Message button
- Min
- Motion add
- Motion set
- Mouse check button
- Mouse check button pressed
- Move bounce all
- Move bounce solid
- Move snap
- Move towards point
- Move wrap
- Mp linear step
- Mp linear step object
- Mp potential step
- Mp potential step object
- Mplay connect status
- Mplay data mode
- Mplay data read
- Mplay session mode
- Mplay session name
- Mplay session status

O

- Object set parent
- Ord

P

- Parameter count
- Parameter string
- Place empty
- Place free
- Place meeting
- Point direction
- Point distance
- Position empty
- Position meeting
- Power

R

- Radtodeg
- Random
- Random range
- Room goto
- Room goto next
- Room goto previous
- Room next
- Room previous
- Room restart
- Surface get width
- Surface getpixel
- Surface reset target
- Surface save
- Surface save part
- Surface set target

T

- Tan
- Template:GMfunc
- Texture preload
- Texture set interpolation
- Texture set priority
- Texture set repeat

W

- Window get fullscreen
- Window mouse get x
- Window mouse get y
- Window mouse set
- Window set cursor
- Window set fullscreen

- Mplay data write
- Mplay end
- Round

S

- Screen redraw
- Set application title
- Show debug message
- Show error
- Show menu

Κάνοντας «*ctrl+δεξί κλικ του ποντικιού*» επάνω στις συναρτήσεις, θα μεταφερθείτε στο ανάλογο Link της ιστοσελίδας, με τη χρήση του browser, ώστε να πάρετε τις ανάλογες πληροφορίες για την σύνταξή τους, τις μεταβλητές που χρειαζόμαστε, για τη χρήση τους, για το αποτέλεσμα που έχουν και τα λοιπά.

Υπάρχουν ακόμη εξήντα τρεις (63) συναρτήσεις οι οποίες αφορούν την σχεδίαση των τρισδιάστατων παιχνιδιών (3D) , οι οποίες είναι οι παρακάτω, και πάλι αλφαβητικά :

ΠΙΝΑΚΑΣ 1.2

D

- D3d draw block
- D3d draw cylinder
- D3d draw floor
- D3d draw wall
- D3d set projection ext
- D3d set projection ortho
- D3d set projection perspective
- D3d set shading
- D3d draw cone

M cont.

- D3d model draw
- D3d model floor
- D3d model load
- D3d model primitive begin
- D3d model primitive end
- D3d model save
- D3d model vertex
- D3d model vertex color
- D3d model vertex normal

S cont.

- D3d set lighting
- D3d set perspective
- D3d start

T

- D3d transform add rotation x
- D3d transform add scaling
- D3d transform add translation
- D3d transform set identity

- D3d draw ellipsoid
- D3d model ellipsoid

E

- D3d end

L

- D3d light define direction
- D3d light define point
- D3d light enable

M

- D3d model block
- D3d model clear
- D3d model cone
- D3d model create
- D3d model cylinder
- D3d model destroy
- D3d model vertex normal color
- D3d model vertex normal texture
- D3d model vertex normal texture color
- D3d model vertex texture
- D3d model vertex texture color
- D3d model wall

P

- D3d primitive begin
- D3d primitive begin texture
- D3d primitive end

S

- D3d set culling
- D3d set depth
- D3d set fog
- D3d set hidden
- D3d transform set rotation x
- D3d transform set rotation y
- D3d transform set rotation z
- D3d transform set scaling
- D3d transform set translation

V

- D3d vertex
- D3d vertex color
- D3d vertex normal
- D3d vertex normal color
- D3d vertex normal texture
- D3d vertex normal texture color
- D3d vertex texture
- D3d vertex texture color

Και εδώ, κάνοντας «*ctrl+δεξί κλικ του ποντικιού*» επάνω στις συναρτήσεις, θα μεταφερθείτε στο ανάλογο Link της ιστοσελίδας, με τη χρήση του browser, ώστε να πάρετε τις ανάλογες πληροφορίες για την σύνταξή τους, τις μεταβλητές που χρειαζόμαστε, για τη χρήση τους, για το αποτέλεσμα που έχουν και τα λοιπά.

Όλοι όσοι ασχολούνται ή έχουν ασχοληθεί με τον προγραμματισμό, σίγουρα θα γνωρίζουν πως σε ένα κομμάτι κώδικα, οποιουδήποτε μεγέθους, θα πρέπει πάντα να υπάρχουν και σχόλια, τα οποία θα υπενθυμίζουν στον συντάκτη ή θα εξηγούν στον αναγνώστη του κώδικα ή στον νέο συντάκτη τη σημασία και τη λειτουργία του. Τα σχόλια είναι γραμμές τις οποίες αγνοεί ο compiler του προγράμματος, τα οποία φυσικά δεν έχουν καμία απολύτως επιρροή στον κώδικά μας. Τα σχόλια μίας γραμμής ξεκινούν με ***//*** , για παράδειγμα :

//Comments

ενώ τα σχόλια τα οποία χρειάζονται περισσότερες από μία γραμμές μπορούμε να τα εσωκλείσουμε ανάμεσα σε ***/**** και ****/*** , ώστε να μην βάζουμε σε κάθε γραμμή τα ***//*** , για παράδειγμα :

/ Comments……*

Comments……

*Comments….. */*

Για τις μεταβλητές μιλάμε εκτενέστερα παρακάτω, απλά να αναφέρουμε πως μια μεταβλητή δημιουργείται τη στιγμή που θα της δώσουμε για πρώτη φορά μια τιμή . Για παράδειγμα :

food = "fruits" ;

Να τονίσουμε πως όπως και σε πολλές άλλες γλώσσες προγραμματισμού, έτσι και στην GML, μπορούμε να εκφράσουμε κάτι που θέλουμε με περισσότερους από έναν τρόπους, όπως για παράδειγμα η αύξηση της τιμής μιας μεταβλητής κατά ένα (1) :

g = g + 1;

είναι το ίδιο ακριβώς και με το :

g += 1;

Μια μεταβλητή εκτός από τις κλασσικές τιμές που μπορεί να έχει, όπως για παράδειγμα έναν αριθμό, ένα κείμενο και τα λοιπά, μπορεί να πάρει και τιμές του 16αδικου συστήματος, αρκεί να ξεκινάει με το σύμβολο **$** , ή να αποτελείται από μια ολόκληρη έκφραση . Για παράδειγμα :

```
{
  x = 23;
  color = $FFAA00;
  str = 'hello world';
  y += 5;
  x *= y;
  x = y << 2;
  x = 23*((2+4) / sin(y));
  str = 'hello' + " world";
  b = (x < 5) && !(x==2 || x==4);
}
```

Στην GML, μπορούμε να δημιουργήσουμε μοναδικούς ή δυαδικούς πίνακες, όπως και στις περισσότερες γλώσσες προγραμματισμού. Η αρίθμηση των κελιών του πίνακα ξεκινάει από το **0** . Ποτέ δεν χρησιμοποιούμε αρνητικές τιμές στην αρίθμηση των κελιών. Υπάρχει ένα όριο 32000 κελιών ανά στήλη και 1000000 κελιών γενικά για κάθε πίνακα. Ας δώσουμε όμως ένα παράδειγμα για την σύνταξη του πίνακα. Έστω ότι θέλουμε να δώσουμε την αριθμητική τιμή 29 στο κελί του πίνακα *a[]* που βρίσκεται στην τέταρτη γραμμή της δεύτερης στήλης :

A[3,1] = 29 ;

ΥΠΟΚΕΦΑΛΑΙΟ 1.3.1
LOOPS

IF STATEMENT

Η σύνταξη του **IF** είναι παρόμοια με τις υπόλοιπες γλώσσες προγραμματισμού, και θα την δείξουμε με παραδείγματα :

```
if (<expression>) <statement> else <statement>
```

```
if (<expression>)
{
  <statement>
}
else
{
  <statement>
}
```

```
{
  if (x<200) {x += 4} else {x -= 4}
}
```

REPEAT STATEMENT

Με το *repeat* εκτελείται η έκφραση που περικλείεται σε αυτό, όσες φορές το ορίσουμε εμείς, συνεχόμενα. Για παράδειγμα :

```
repeat (<expression>) <statement>
```

```
{
  repeat (5) instance_create(random(400),random(400),ball);
}
```

WHILE STATEMENT

Το γνωστό σε όλους μας *while* , ή αλλιώς «Όσο» . Όσο συμβαίνει και ισχύει κάτι, εκτελείται το *Loop* που περιλαμβάνει την έκφραση που θέλουμε. Για παράδειγμα :

```
{
  while (!place_free(x,y))
  {
    x = random(room_width);
    y = random(room_height);
  }
}
```

DO STATEMENT

Ή αλλιώς *Do…until*, δηλαδή το γνωστό σε όλους μας «do…while». Είναι το *loop* το οποίο πρώτα εκτελεί την έκφραση την οποία περικλείει και μετά ελέγχει τον περιορισμό, ο οποίος εάν είναι ακόμα σε ισχύει τότε η έκφραση εκτελείται ξανά και ούτω καθεξής . Για παράδειγμα :

```
{
  do
  {
    x = random(room_width);
    y = random(room_height);
  }
  until (place_free(x,y))
}
```

FOR STATEMENT

Το γνωστό σε όλους μας *for* ή *για* . Είναι το κλασσικό *loop* στο οποίο δίνουμε μια μεταβλητή και του λέμε να εκτελεί συνεχόμενα την έκφραση, όσο η τιμή της μεταβλητής αυτής δεν έχει φτάσει στην «τάδε», αυξάνοντας σε κάθε *loop* την τιμή της, ανάλογα με την εντολή που θα δώσουμε. Για παράδειγμα, έστω ότι θέλουμε να γεμίσουμε τα 20 κελιά του μοναδικού πίνακα *a[]* με τιμές από το 1 μέχρι το 20 :

```
{
  for (i=0; i<=19; i+=1) a[i] = i+1;
}
```

SWITCH STATEMENT

Εάν οι ενέργειες που θέλουμε να ακολουθήσουν είναι εξαρτημένες από διάφορες περιπτώσεις, τότε χρησιμοποιούμε το γνωστό μας και από τις άλλες γλώσσες προγραμματισμού *switch* . Ας δούμε παρακάτω την σύνταξη αυτού του loop και ένα παράδειγμα :

```
switch (<expression>)
{
  case <expression1>: <statement1>; ... ; break;
  case <expression2>: <statement2>; ... ; break;
  ...
  default: <statement>; ...
}
```

```
switch (keyboard_key)
{
  case vk_right:
  case vk_numpad4:
    x -= 10; break;
  case vk_left:
  case vk_numpad6:
    x += 10; break;
}
```

Με την λέξη κλειδί *break* όπως βλέπουμε «σταματάμε» την κάθε ομάδα περιπτώσεων (*case*) .

Εάν θέλουμε από μία έκφραση να επιστρέφεται στην εφαρμογή μας κάτι, όπως μια άλλη έκφραση, μια αριθμητική τιμή, μια μεταβλητή και τα λοιπά, τότε χρησιμοποιούμε την λέξη κλειδί ***return***.

```
return <expression> ;
```

ΥΠΟΚΕΦΑΛΑΙΟ 1.3.2
ΜΕΤΑΒΛΗΤΕΣ

Οι μεταβλητές είναι αυτό ακριβώς που λέει και το όνομά τους. Είναι συνιστώσες οι οποίες δεν έχουν σταθερή τιμή, δηλαδή η τιμή τους μπορεί να αλλάξει ανά πάσα στιγμή. Μπορούμε μέσα σε μια μεταβλητή να ορίσουμε μια ολόκληρη αριθμητική παράσταση την οποία να χρησιμοποιούμε όποτε τη χρειαζόμαστε. Ή απλά να αποτελεί έναν απλό αριθμό, ο οποίος θα παίρνει τις ανάλογες τιμές που θα του ορίζουμε, ή ακόμα και μια τιμή όπως *true* ή *false* κ.τ.λ. .

Το όνομα των μεταβλητών που ορίζουμε, μπορεί να περιέχει γράμματα, αριθμούς και την κάτω παύλα (_) , αρκεί να μην ξεκινάει με αριθμό . Δηλαδή, ένα αποδεκτό όνομα μεταβλητής είναι το **_derp5** , ενώ ένα μη αποδεκτό είναι το **5red** . Επίσης ένα ακόμη παράδειγμα μη αποδεκτού ονόματος μεταβλητής είναι και το **red$7** , γιατί όπως είπαμε μόνο η κάτω παύλα είναι αποδεκτή από τα σύμβολα, ενώ κανένα από τα παρακάτω δεν επιτρέπεται :

```
. !     @     #     $
%       ^     &     *     (
) -     =     +     \
< >     /     ?     <space>
{ }     [     ]     ,
| `     ~     "     '
```

Για να ορίσουμε μια μεταβλητή, υπάρχουν δύο τρόποι. Ο ένας είναι με την drag-and-drop μέθοδο, με το *Set Variable* Action, και ο άλλος είναι με την χρήση κώδικα, με μια πολύ απλή εντολή :

Variable = 1 ;

ή

Variable = random(10) ; *//ένα παράδειγμα εκφράσεων που προείπαμε*

Οι μεταβλητές χωρίζονται σε δύο(2) κατηγορίες, την τοπικές (**Local**) και τις **Global**. Κάθε φορά που δημιουργούμε μια μεταβλητή με έναν από τους τρόπους που είπαμε παραπάνω, τότε αυτή έχει ισχύ μόνο για το αντικείμενο στο οποίο την δημιουργήσαμε, είναι δηλαδή τοπική. Εάν βρισκόμαστε σε κάποιο αντικείμενο και

θέλουμε να αλλάξουμε την τιμή μιας τοπικής μεταβλητής ενός άλλου αντικειμένου, αυτό γίνεται με τον εξής τρόπο :

Car.speed = 5 ;

όπου Car είναι το αντικείμενο και speed η μεταβλητή που θέλουμε. Αυτό όμως προϋποθέτει φυσικά να έχουμε ήδη δημιουργήσει αυτήν την μεταβλητή στο αντικείμενο car.

Εάν θέλουμε, μια μεταβλητή που δημιουργούμε, να μην είναι τοπική, αλλά γενική, δηλαδή να ισχύει παντού, τότε την δηλώνουμε ως εξής :

Global.speed=1;

Επομένως, όπου κι αν βρισκόμαστε, θα μπορούμε να καλούμε την μεταβλητή speed. Συνήθως, στην αρχή του παιχνιδιού, δηλώνουμε όλες μαζί τις global μεταβλητές που θα χρειαστούμε στην πορεία, ώστε να είναι και όλες μαζί μαζεμένες και όχι διάσπαρτες.

Στο Game Maker υπάρχουν ήδη κάποιες global μεταβλητές, τις οποίες μπορούμε να χρησιμοποιήσουμε όποτε χρειαστεί. Μερικές από αυτές είναι οι εξής :

x
y
hspeed
vspeed direction
speed
visible
image_index
image_speed
score
lives
health
mouse_x
mouse_y

ΚΕΦΑΛΑΙΟ 2
ΔΗΜΙΟΥΡΓΙΑ ΕΝΟΣ ΠΑΙΧΝΙΔΙΟΥ – OIONOS GAME

ΕΙΣΑΓΩΓΗ

Το παιχνίδι *Oionos* αποτελεί ένα αντιπροσωπευτικό δείγμα των ικανοτήτων του *Game Maker* . Υπάρχουν πολλές λεπτομέρειες οι οποίες μπορεί να μην είναι εμφανείς με την πρώτη ματιά αλλά δεν παύουν να είναι σημαντικές για την εκπόνηση αυτού του project. Σε αυτό το παιχνίδι θα συναντήσουμε στοιχεία όπως αλλαγή Sprite, δημιουργία αντικειμένων στην πορεία του παιχνιδιού, αλλαγή δωματίων, κίνηση με το ποντίκι, παύση, αποθήκευση-φόρτωση, τυχαία κίνηση αντικειμένων, προσθαφαίρεση ζωών και πόντων, αντιστροφή της κατεύθυνσης των αντικειμένων καθώς και πολλά άλλα ακόμη Κάποιες από αυτές τις λεπτομέρειες λοιπόν, ίσως τις πιο σημαντικές, θα εξηγήσουμε παρακάτω.

Ας πάρουμε τα πράγματα από την αρχή. Τι είναι το *Oionos Game* ; Είναι ένα παιχνίδι δύο επιπέδων. Στο πρώτο επίπεδο υπάρχει ένας βασικός χαρακτήρας ο οποίος κινείται με τα βελάκια του πληκτρολογίου μας (arrow keys) . Ενώ στο δεύτερο επίπεδο ο χαρακτήρας αυτός κινείται με το ποντίκι του υπολογιστή μας (η με το touchpad και γενικά απλά ακολουθεί τον κέρσορα/δείκτη) . Ο σκοπός του παιχνιδιού είναι η συλλογή όσων περισσότερων πόντων μπορούμε, πριν τον μηδενισμό του αριθμού των ζωών μας. Αυτά είναι τα γενικά χαρακτηριστικά του παιχνιδιού, τα οποία θα εξηγήσουμε παρακάτω.

ΥΠΟΚΕΦΑΛΑΙΟ 2.1
OIONOS GAME – Η ΥΛΟΠΟΙΗΣΗ

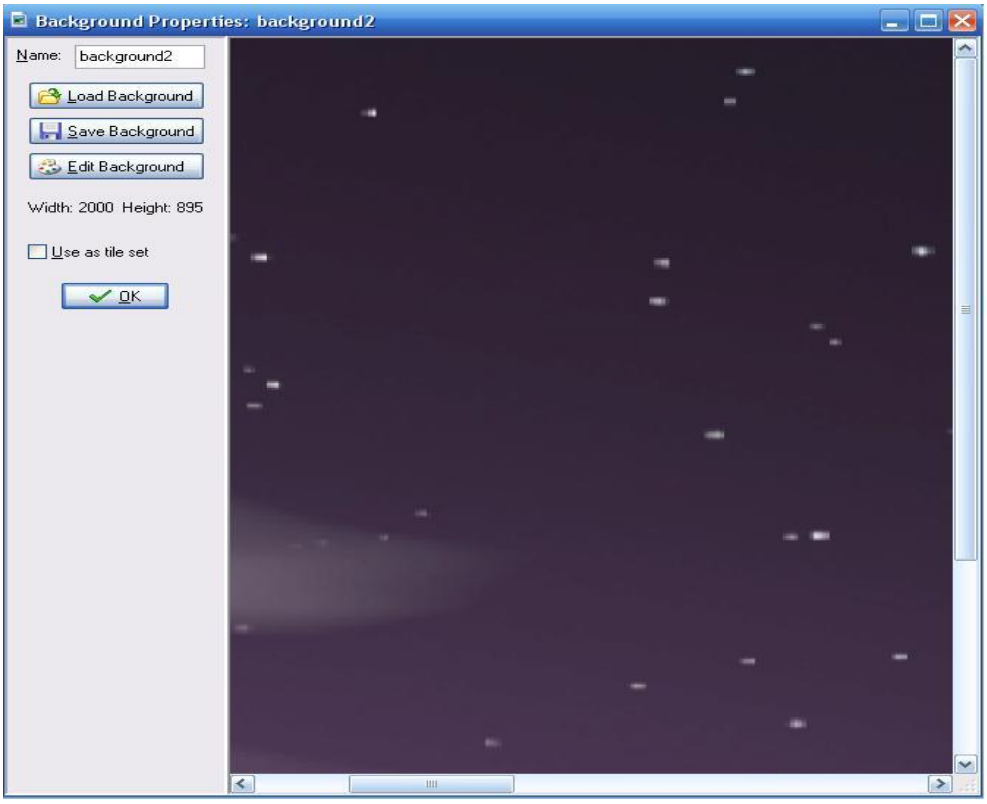
Ας ξεκινήσουμε από τα βασικά μέρη του παιχνιδιού. Για τις ανάγκες του παιχνιδιού αυτού έχουν δημιουργηθεί πέντε (5) διαφορετικά δωμάτια. Το βασικό δωμάτιο του παιχνιδιού (main), το δωμάτιο του πρώτου σκέλους του παιχνιδιού (game 1), η συνέχεια του πρώτου παιχνιδιού (game 1.2), το δωμάτιο του δεύτερου σκέλους του παιχνιδιού(game 2) και το δωμάτιο τερματισμού-εξόδου του παιχνιδιού (exit).

Εικόνα 2.1

Το κάθε δωμάτιο, όπως έχουμε αναφέρει ξανά, πρέπει να έχει κάποιο background, έτσι ώστε να μην είναι σκέτο και μονότονο. Το κάθε background συνήθως αντιπροσωπεύει το είδος και την κατηγορία του κάθε δωματίου ή ακόμη και ολόκληρου του παιχνιδιού. Σε κάποια παιχνίδια συνίσταται η χρήση του ίδιου background σε όλα τα δωμάτια, ενώ σε κάποια άλλα παιχνίδια συνίσταται διαφορετικό, ανάλογα όπως είπαμε στην περίσταση. Το background μπορεί να είναι μία εικόνα μόνη της, η οποία θα έχει την ίδια ακριβώς ανάλυση και μέγεθος με το δωμάτιο, αλλά μπορεί να είναι και μια μικρότερη εικόνα η οποία θα εμφανίζεται πολλές φορές, τόσες όσες χρειάζεται ώστε να καλύψει την επιφάνεια του δωματίου. Έτσι λοιπόν το *main* δωμάτιο του *Oionos Game* έχει την εξής μορφή :

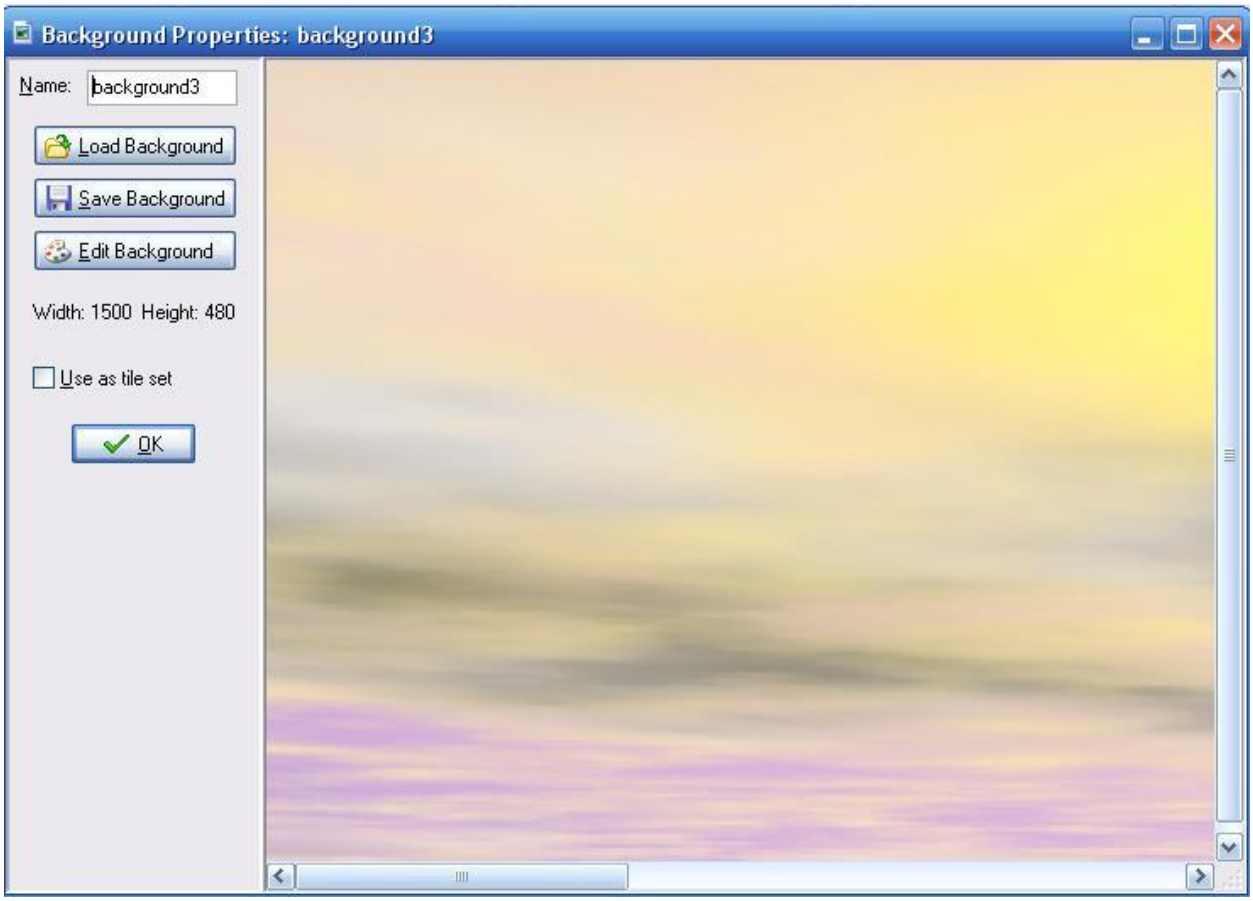
Εικόνα 2.2

Τα *game 1 και game 1.2* δωμάτια έχουν ως background το παρακάτω :



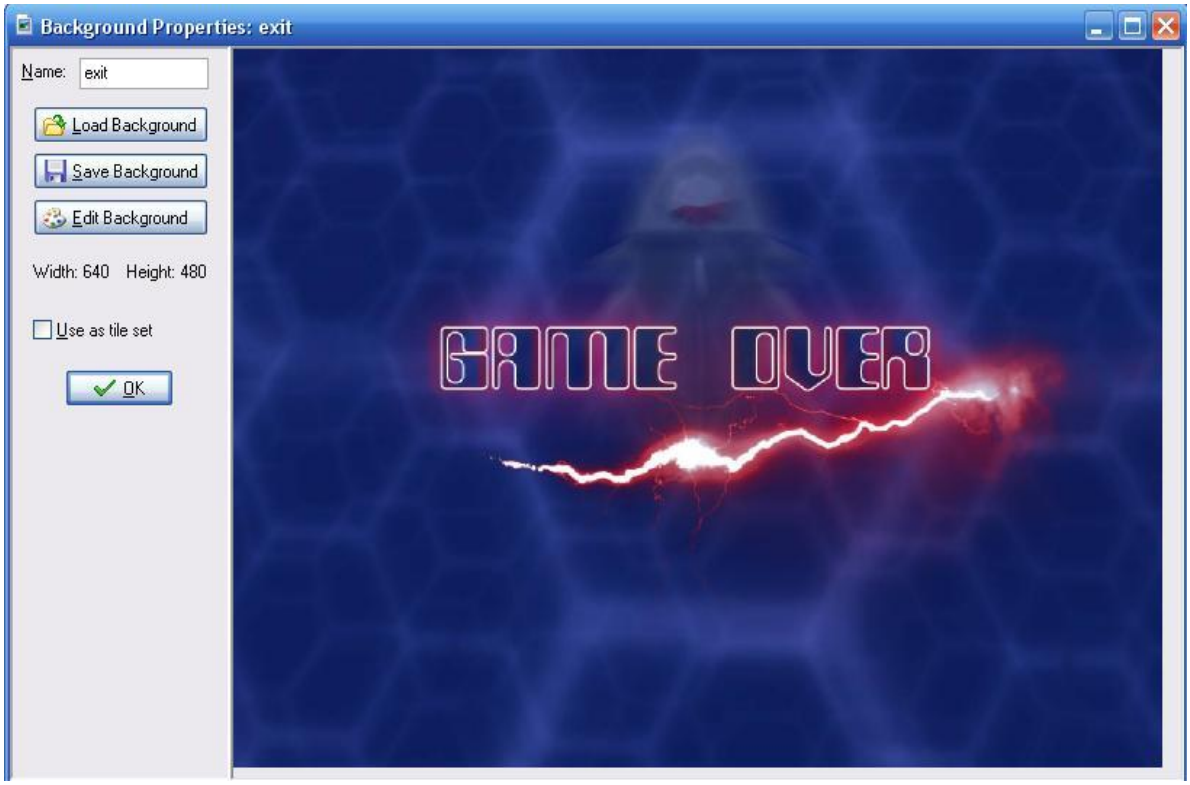
Εικόνα 2.3

Το *game 2* έχει το εξής φόντο :



Εικόνα 2.4

Ενώ, τέλος, το δωμάτιο *exit* έχει ως υπόβαθρο το παρακάτω :



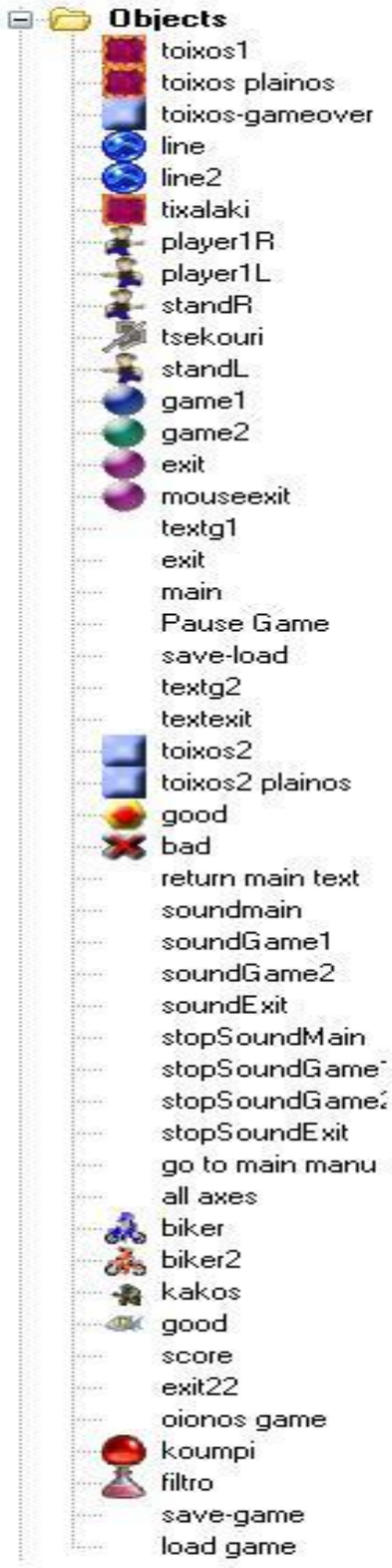
Εικόνα 2.5

Για να περιγράψουμε τα *sprites* και τα *objects* του παιχνιδιού, ας μιλήσουμε περιληπτικά για τα σκέλη αυτού. Εκτενέστερα θα μιλήσουμε βήμα βήμα στην πορεία. Στο game 1, ο χαρακτήρας μας κινείται σε ένα δωμάτιο περισυλλέγοντας κάποια πετράδια τα οποία και του δίνουν πόντους, ενώ πρέπει να αποφύγει τα «κακά» πετράδια και τα αυτοκινούμενα τσεκούρια τα οποία του αφαιρούν ζωές. Πατάει ένα κουμπί για να ανοίξει μια πύλη όπου οδηγεί στο δεύτερο κομμάτι του δωματίου, όπου συνεχίζει να κάνει το ίδιο, μέχρι να φτάσει σε ένα φίλτρο το οποίο δίνει πολλούς πόντους στο σκορ του παίκτη και τον μεταφέρει στο main room . Στο δεύτερο σκέλος του παιχνιδιού, ο χαρακτήρας μας κινείται με το ποντίκι, μαζεύοντας και πάλι καλά και κακά αντικείμενα .

Τα sprites λοιπόν που χρειαστήκαμε για την δημιουργία αυτού του παιχνιδιού είναι τα παρακάτω :

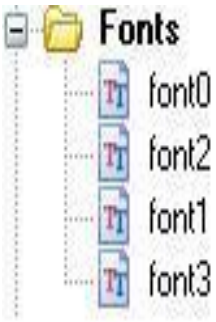
Εικόνα 2.6

Και τα ανάλογα objects είναι τα παρακάτω. Εδώ να τονίσουμε πως υπάρχουν διαφορετικά αντικείμενα (objects) με διαφορετικές ιδιότητες, τα οποία όμως έχουν το ίδιο sprite. Αυτό φυσικά γίνεται διότι κάποιες φορές θέλουμε να υπάρχουν στο παιχνίδι ίδια στην εμφάνιση αντικείμενα, όπως για παράδειγμα τοίχοι, αλλά με διαφορετικά αποτελέσματα κατά την επαφή του παίκτη με αυτούς. Είναι επίσης σημαντικό να αναφέρουμε πως υπάρχουν αντικείμενα στα οποία δεν αντιστοιχεί καμία εικόνα, για τον απλούστατο λόγο ότι δεν θέλουμε να έχουν κάποια απεικόνιση στο παιχνίδι, αλλά να υπάρχουν «διάφανα» στο υπόβαθρο. Ένα παράδειγμα είναι τα αντικείμενα τα οποία αντιπροσωπεύουν ήχους και τα λοιπά .


Objects
toixos1
toixos plainos
toixos-gameover
line
line2
tixalaki
player1R
player1L
standR
tsekouri
standL
game1
game2
exit
mouseexit
textg1
exit
main
Pause Game
save-load
textg2
textexit
toixos2
toixos2 plainos
good
bad
return main text
soundmain
soundGame1
soundGame2
soundExit
stopSoundMain
stopSoundGame
stopSoundGame2
stopSoundExit
go to main manu
all axes
biker
biker2
kakos
good
score
exit22
oionos game
koumpi
filtro
save-game
load game

Εικόνα 2.7

Σε κάποια δωμάτια του Oionos Game θα παρατηρήσετε πως υπάρχουν κάποια κομμάτια κειμένου. Αυτά συνήθως υπάρχουν για να δίνουν κάποιες οδηγίες στον παίκτη του παιχνιδιού, κάποιες πληροφορίες, κάποιες υποδείξεις και τα λοιπά. Η εισαγωγή κειμένου γίνεται με την βοήθεια του ανάλογου action όπως έχουμε ξαναπεί. Κατά την δημιουργία αυτού του action, δίνουμε ως παράμετρο και το ανάλογο *font* το οποίο έχουμε ήδη δημιουργήσει ανάλογα με τις ανάγκες του παιχνιδιού και αφορά την γραμματοσειρά που θα έχει το κείμενό μας, το μέγεθος των γραμμάτων, το χρώμα και τα λοιπά. Τα fonts λοιπόν που έχουμε χρειαστεί και έχουμε δημιουργήσει είναι τέσσερα και τα βλέπετε στην διπλανή φωτογραφία .

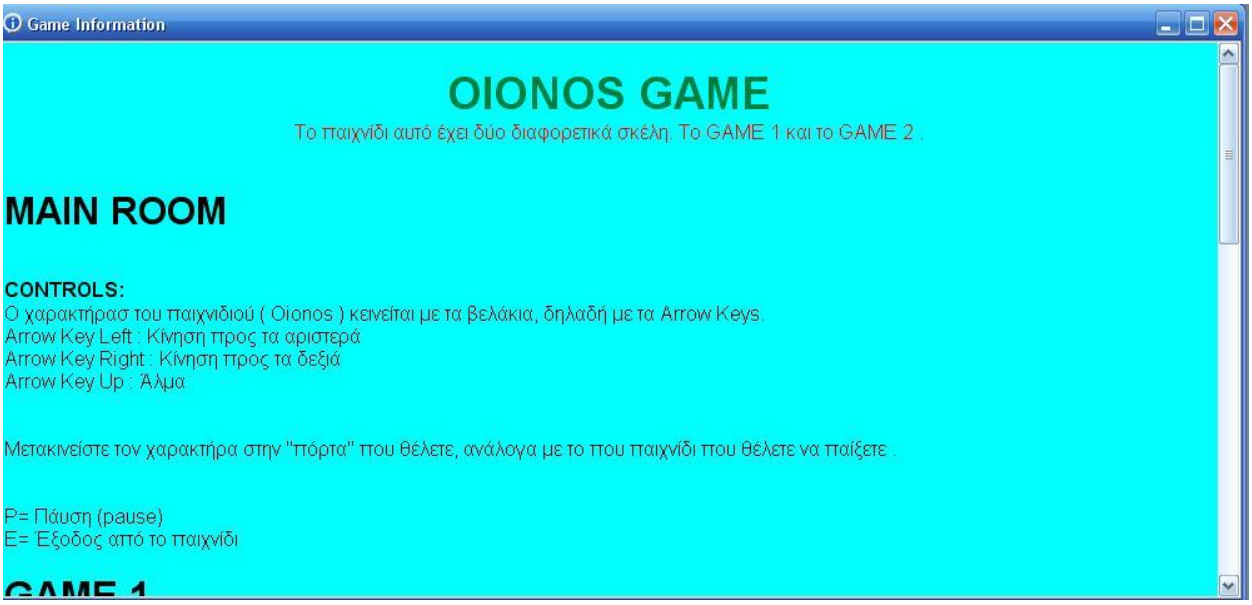


Εικόνα 2.8

Τεράστιο ρόλο σε ένα παιχνίδι είναι οι ήχοι (sounds) που ακούγονται. Χωρίς ήχους, μουσικό υπόβαθρο και τα λοιπά ένα παιχνίδι μπορεί να χάσει μέχρι και το 50%-70% του ενδιαφέροντός του, διότι θα είναι «στεγνό» (νομίζω πως όλοι μπορούμε να το φανταστούμε και να το κατανοήσουμε αυτό). Στο Oionos Game λοιπόν έχουν εισαχθεί μουσικά υπόβαθρα, δηλαδή μουσικά κομμάτια τα οποία παίζουν κατά την διάρκεια του παιχνιδιού (διαφορετικά σε κάθε δωμάτιο) και οι ανάλογοι ήχοι οι οποίοι αναπαράγονται ανάλογα με τα διάφορα δρώμενα που λαμβάνουν χώρα σε αυτό (χάσιμο ζωής και τα λοιπά). Έτσι, οι ήχοι που έχουμε εισάγει με τα (χρήσιμα) «ψευδώνυμά» τους είναι οι παρακάτω :

Εικόνα 2.9

Ξεκινώντας λοιπόν κάποιος να παίξει το παιχνίδι, μόλις ολοκληρωθεί το loading (φόρτωση), το πρώτο πράγμα που του εμφανίζεται είναι το *game info* . Το game info είναι ένα παράθυρο που δίνει στην χειριστή βασικότατες πληροφορίες για το παιχνίδι και για τον χειρισμό του. Το παράθυρο αυτό λοιπόν είναι κάπως έτσι :



Εικόνα 2.10

Ο παίκτης, με scroll down διαβάζει όλες τις πληροφορίες που υπάρχουν σε αυτό το παράθυρο. Νομίζω πως σε αυτό το σημείο πρέπει να δούμε την πλήρες μορφή αυτού του εγχειριδίου χρήσης, όπως ακριβώς εμφανίζεται στον χρήστη :

OIONOS GAME
Το παιχνίδι αυτό έχει δύο διαφορετικά σκέλη. Το GAME 1 και το GAME 2 .

MAIN ROOM

CONTROLS:
Ο χαρακτήρας του παιχνιδιού (Oionos) κινείται με τα βελάκια, δηλαδή με τα Arrow Keys.
Arrow Key Left : Κίνηση προς τα αριστερά
Arrow Key Right : Κίνηση προς τα δεξιά
Arrow Key Up : Άλμα

Μετακινείστε τον χαρακτήρα στην "πόρτα" που θέλετε, ανάλογα με το που παιχνίδι που θέλετε να παίξετε .

P= Παύση (pause)
E= Έξοδος από το παιχνίδι

GAME 1

CONTROLS:
Arrow Key Left : Κίνηση προς τα αριστερά
Arrow Key Right : Κίνηση προς τα δεξιά
Arrow Key Up : Άλμα

P= Παύση (pause)
E= Έξοδος από το παιχνίδι
M= Μεταφορά στο κεντρικό Room

Ακουμπώντας τα κίτρινο-κόκκινα αντικείμενα παίρνετε 1 πόντο στο score σας.
Ακουμπώντας τα Χ και τα κινούμενα τσεκούρια αντικείμενα χάνεται 1 ζωή.
Αν αγγίξεις το κόκκινο κουμπί τότε ανοίγει η κόκκινη είσοδος.
ΠΡΟΣΟΧΗ!!! Εάν ακουμπήσετε όλα τα τσεκούρια το κουμπί καταστρέφεται, επομένως η πόρτα δεν ανοίγει ποτέ!!!!!!
Ακουμπώντας το φίλτρο παίρνετε 30 πόντους στο score σας και πηγαίνετε στην δεύτερη πίστα αυτού του παιχνιδιού!!

Στην δεύτερη πίστα ο χαρακτήρας κάνει ελεύθερη πτώση, κι εσείς προσπαθείτε να μαζέψετε όσους περισσότερους πόντους μπορείτε μέχρι να φτάσει κάτω. Προσοχή και πάλι στα αντικείμενα Χ.

GAME 2

CONTROLS:
Arrow Key Left : Κίνηση προς τα αριστερά
Arrow Key Right : Κίνηση προς τα δεξιά
Arrow Key Up : Άλμα

P= Παύση (pause)
E= Έξοδος από το παιχνίδι
M= Μεταφορά στο κεντρικό Room

Ακουμπώντας τους γαλάζιους μηχανόβιους παίρνεται 1 πόντο στο score σας.
Ακουμπώντας τους κόκκινους μηχανόβιους χάνεται 1 ζωή
Ακουμπώντας τα ψαράκια παίρνεται 2 πόντους στο score σας.
Ακουμπώντας τα Χ αντικείμενα χάνεται 3 ζωές

ΠΡΟΣΟΧΗ

Το παιχνίδι τελειώνει όταν πατήσετε έξοδο ή όταν μηδενιστούν οι ζωές σας.

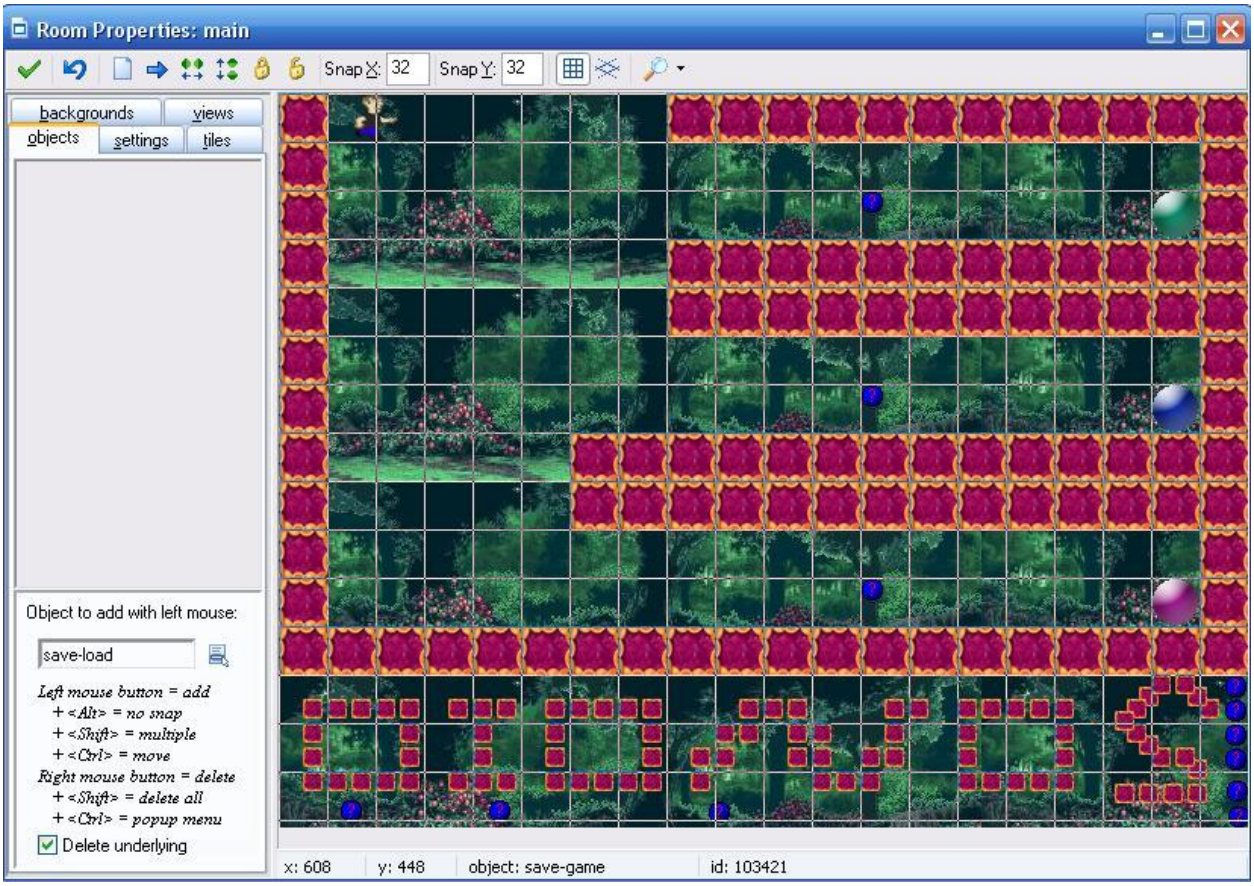
Μέχρι να τελειώσουν οι ζωές σας μπορείτε να ξαναπαίξετε όποιο παιχνίδι θέλετε όσες φορές θέλετε.

Στο Main Room, μπορείτε να πατήσετε το πλήκτρο s για να κάνετε Save το παιχνίδι και το πλήκτρο L για να κάνετε Load . Αλλά MONO στο Main Room .

Έτσι λοιπόν, αφού ο παίκτης διαβάσει τις οδηγίες αυτές, κλείνει αυτό το παράθυρο και ξεκινάει το παιχνίδι. Και ξεκινάει από το *main room* :

Εικόνα 2.11

Θα δούμε παρακάτω ακόμη μία φωτογραφία του main room, κα την δημιουργία της. Στην δεύτερη αυτή φωτογραφία θα παρατηρήσουμε και τα αντικείμενα τα οποία δεν βλέπει ο παίκτης του παιχνιδιού, όπως τα αντικείμενα για την έναρξη και την παύση του ήχου και τα λοιπά. Αυτά βρίσκονται κάτω δεξιά, και επειδή δεν αντιστοιχούν σε κάποιο sprite, έχουν την μορφή ενός ερωτηματικού.



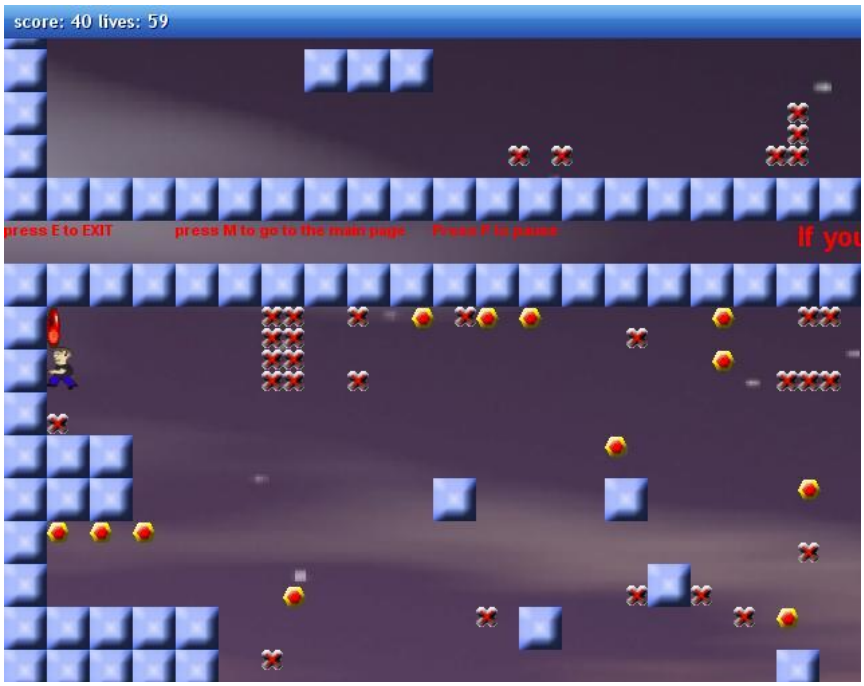
Εικόνα 2.12

Στη συνέχεια θα δούμε κάποια ακόμη screenshots κατά την χρήση του παιχνιδιού και μετά θα περάσουμε στα events και στα actions που έχουν δημιουργηθεί. Επομένως ο παίκτης μπαίνοντας στο *game 1* συναντάει τα παρακάτω :


score: 6 lives: 92
If you touch ALL the axes, the door will NEVER open!

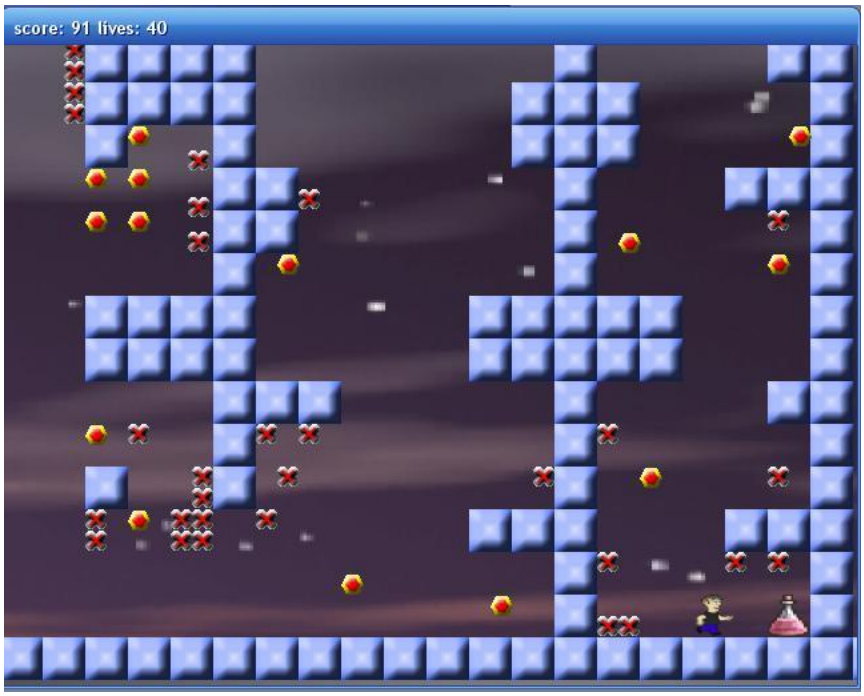
Εικόνα 2.13

Εικόνα 2.14



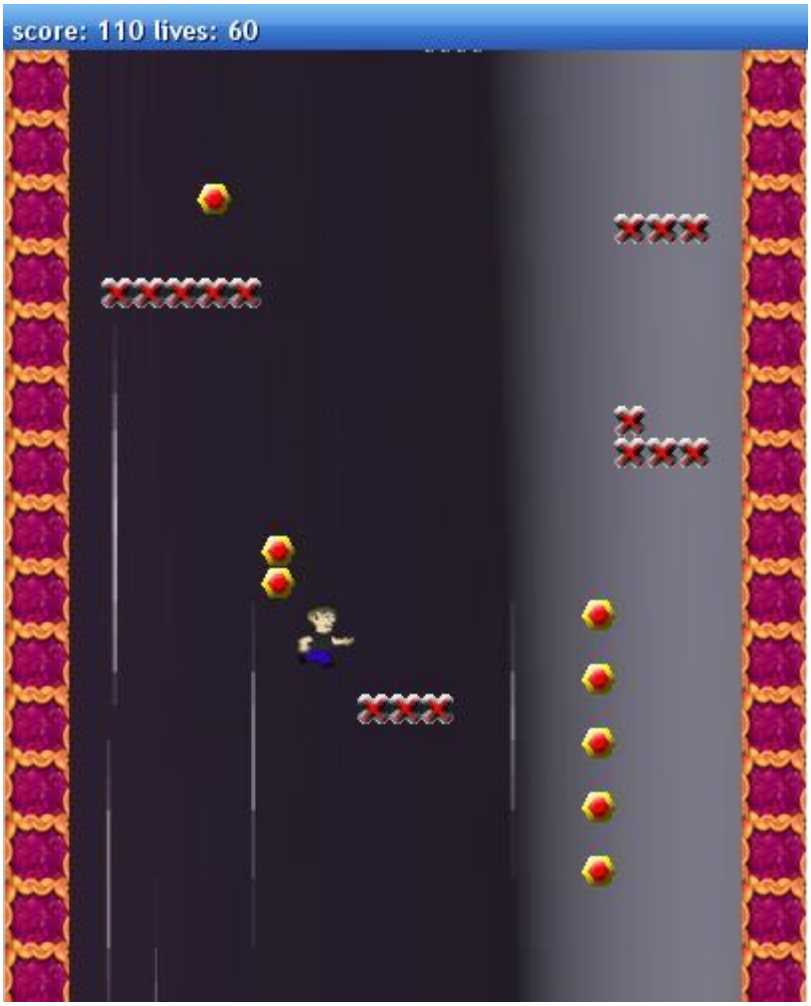
Εικόνα 2.15

Εικόνα 2.16



Εικόνα 2.17

Μόλις πάρουμε το «φίλτρο», αφού πάρουμε τους πόντους και προστεθούν στο score μας, μεταφερόμαστε αυτόματα στην δεύτερη πίστα αυτού του παιχνιδιού, δηλαδή στο room *game 1.2* .



Εικόνα 2.18

Όπως είπαμε, σε αυτήν την πίστα ο χαρακτήρας μας κάνει ελεύθερη πτώση προσπαθώντας να μαζέψει όσους περισσότερους πόντους μπορεί χωρίς να χάνει ζωές. Μόλις φτάσει κάτω κάτω, τελειώνει αυτόματα αυτό το game και μεταφέρεται στο main room. Από εκεί μπορεί να ξαναμπεί εάν θέλει στο πρώτο παιχνίδι, ειδάλλως πηγαίνει στο δεύτερο.

Στο *game 2* συναντάμε αυτά που θα δούμε στις εικόνες παρακάτω :

Εικόνα 2.19

Εικόνα 2.20

Επίσης ας δούμε και ένα screenshot την στιγμή που έχουμε πατήσει το πλήκτρο P (pause) , δηλαδή έχουμε κάνει παύση του παιχνιδιού.

Εικόνα 2.21

Τέλος, θα δούμε και ένα screenshot από το *exit room* του παιχνιδιού, στο δωμάτιο που μεταφέρεται ο παίχτης μας εάν πατήσουμε έξοδο ή εάν τελειώσουν οι ζωές μας :

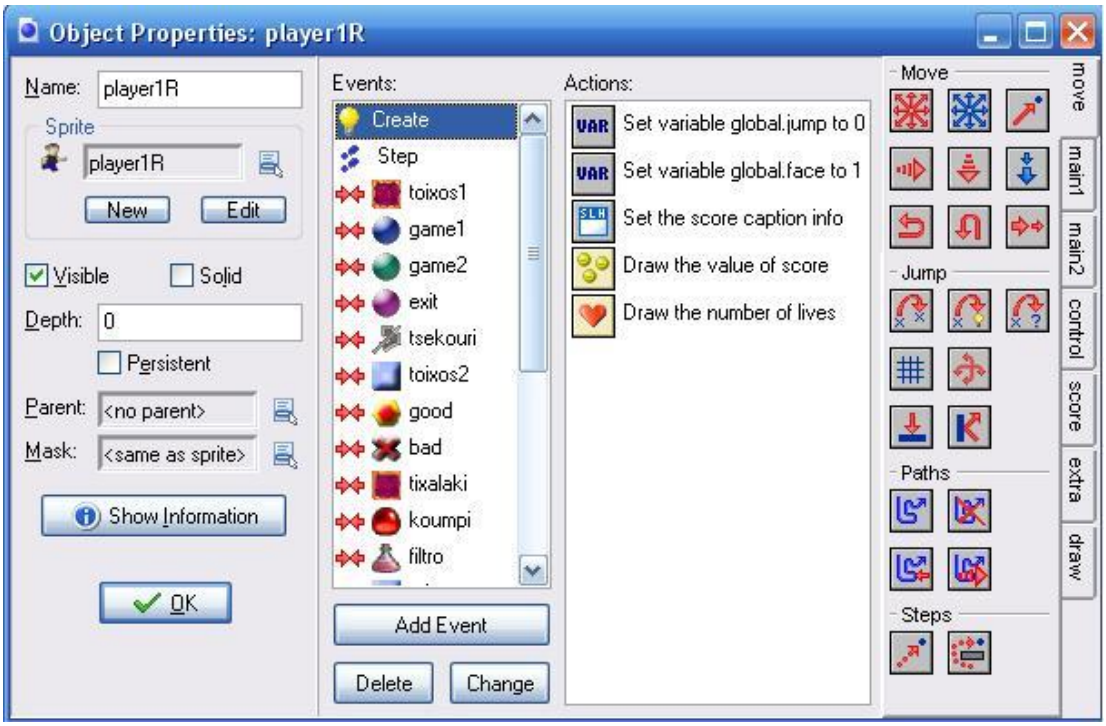

score: 17 lives: 91
Score: 17
GAME OVER
Press Escape to Exit
Dionos Game version 2.0.1 contact: takis4gm@gmail.com

Εικόνα 2.22

ΥΠΟΚΕΦΑΛΑΙΟ 2.2
OIONOS GAME – EVENTS ΚΑΙ ACTIONS

Και ήρθη λοιπόν η ώρα να περάσουμε στα αντικείμενα και στα actions που τους έχουμε καταχωρήσει. Θα αναφέρουμε ένα ένα μόνο τα αντικείμενα τα οποία περιέχουν κάποια events και actions .

Αντικείμενο: **player1R**

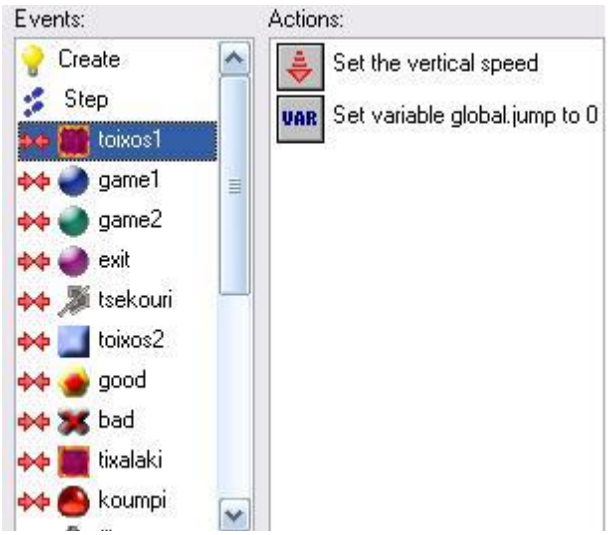


Εικόνα 2.23

Κατά τη δημιουργία του παίρνουν αρχικές τιμές οι μεταβλητές που έχουν να κάνουν με την φορά που θα έχει ο χαρακτήρας ανάλογα με την κατεύθυνσή του αλλά και για το εάν έχει το δικαίωμα να πραγματοποιήσει άλμα. Επίσης ζητάει να εμφανίζεται στο παράθυρο του παιχνιδιού ο αριθμός των ζωών που του απομένουν καθώς και το score που έχει κάνει μέχρι στιγμής.

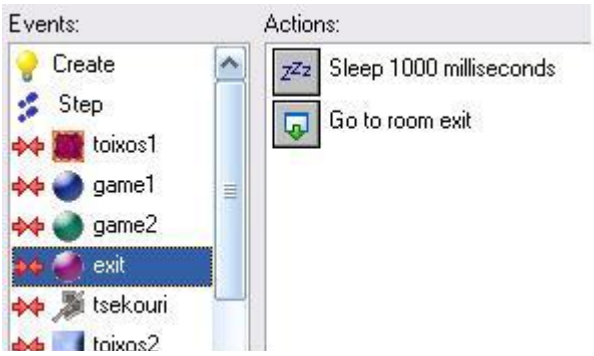
Εικόνα 2.24

Στο event Step, ελέγχει κάθε φορά εάν υπάρχει χώρος για να προχωρήσει καθώς και την ταχύτητά του, έτσι ώστε να σταματάει πριν από κάθε εμπόδιο.



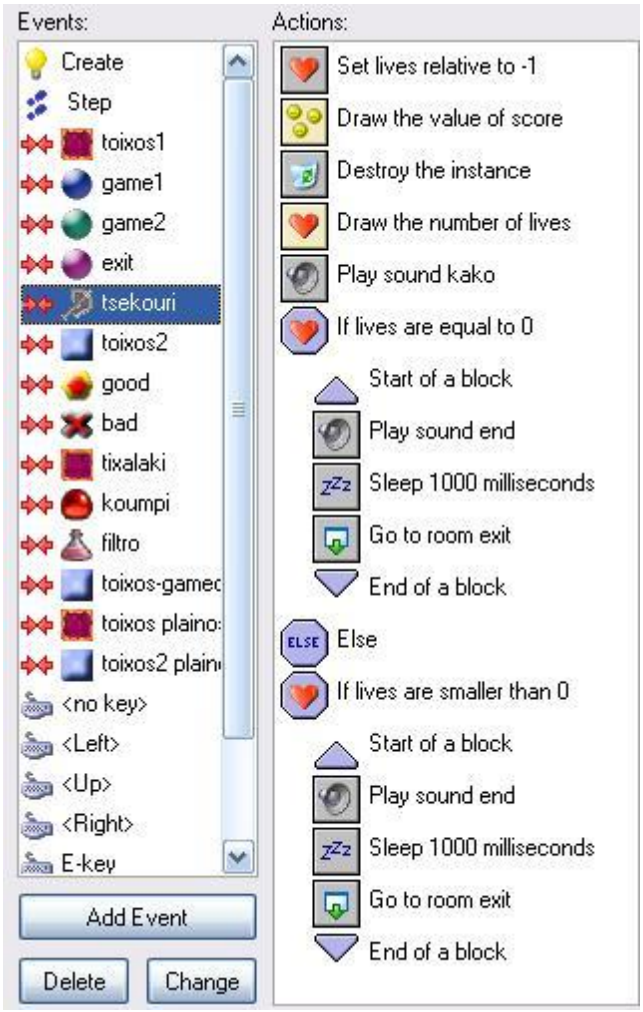
Εικόνα 2.25

Εδώ έχουμε ένα collision event με το αντικείμενο toixos1. Εάν λοιπόν ο χαρακτήρας μας έρθει σε επαφή με αυτό το αντικείμενο τότε σταματάει(η ταχύτητά του παίρνει αντίθετη τιμή) και επίσης ο μεταβλητή του άλματος ξανά μηδενίζεται ώστε να έχει ξανά το δικαίωμα να κάνει άλμα. Τα ίδια ακριβώς actions υπάρχουν και σε ενδεχόμενη επαφή του χαρακτήρα με τα αντικείμενα tixalaki, toixos2, toixos-gameover . Με τα αντικείμενα toixos plainos και toixos2 plainos, απλά δεν ξανά μηδενίζεται η μεταβλητή του άλματος, ώστε ο χαρακτήρας μας να μην έχει το δικαίωμα να πραγματοποιήσει άλμα από εκείνο το σημείο.



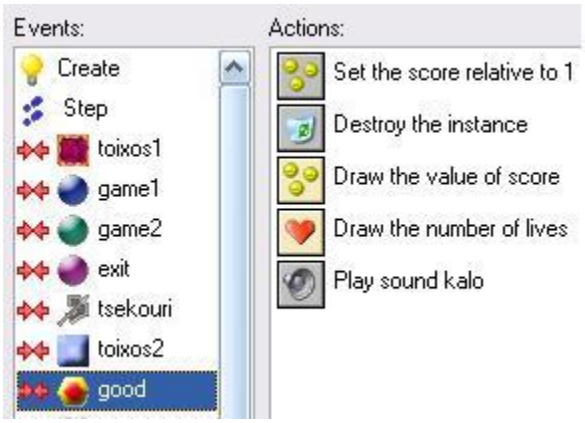
Εικόνα 2.26

Στα collision events με τα game1, game2 και exit αντικείμενα, ο παίκτης απλά μεταφέρεται στο ανάλογο room(δωμάτιο). Απλά στο αντικείμενο exit έχουμε προσθέσει μία ακόμη λεπτομέρεια, η οποία είναι η παύση ενός δευτερολέπτου πριν από την μετάβαση, έτσι ώστε να μην γίνεται απότομα η αλλαγή του δωματίου. Αυτό θα το παρατηρήσει κάποιος καλύτερα μόλις παίξει το παιχνίδι.



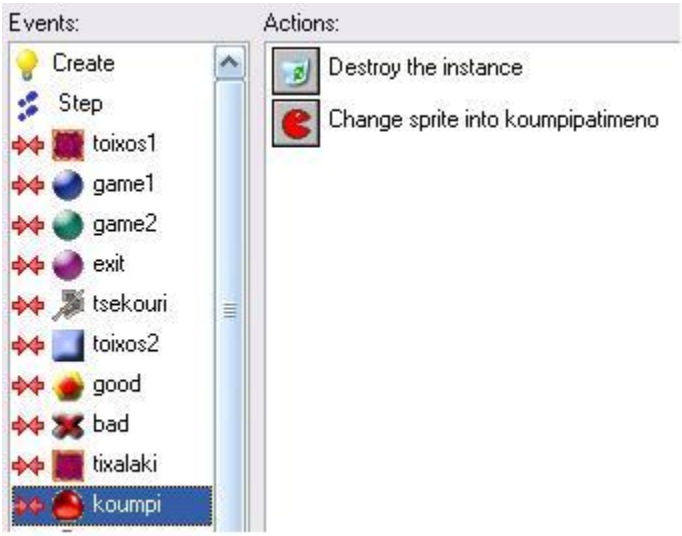
Εικόνα 2.27

Σε ένα ενδεχόμενο collision του χαρακτήρα με το αντικείμενο tsekouri, μειώνονται οι ζωές κατά ένα(1), εμφανίζεται και σε εμάς αυτή η μείωση, καταστρέφεται το αντικείμενο tsekouri(προσοχή όχι όλα τα αντικείμενα tsekouri αλλά μόνο το συγκεκριμένο), ακούγεται ο ανάλογος ήχος και μετά γίνεται ο έλεγχος εάν έχουν μηδενιστεί οι ζωές ώστε να μεταφερθούμε στο room exit. Τα ίδια ακριβώς actions έχουμε και σε ενδεχόμενο collision με το αντικείμενο bad.



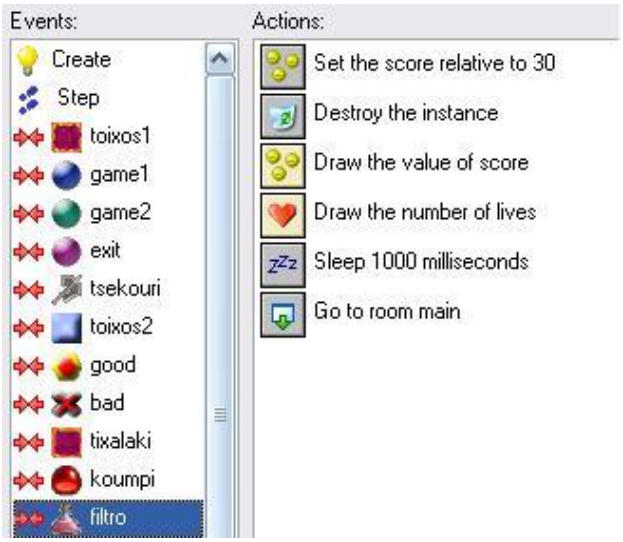
Εικόνα 2.28

Αντίθετα με το collision με τα αντικείμενα bad ή tsekouri, ένα collision με το αντικείμενο good μας δίνει έναν πόντο στο score. Τα υπόλοιπα είναι τα ίδια, όπως η αναπαραγωγή του ανάλογου ήχου και η καταστροφή του αντικειμένου good.



Εικόνα 2.29

Όταν ο χαρακτήρας μας έρθει σε επαφή με το αντικείμενο koumpi, τότε το αντικείμενο αυτό αλλάζει sprite(σε koumpipatimeno), δηλαδή εικόνα, επομένως σε εμάς φαίνεται πως έχουμε πατήσει αυτό το κουμπί. Επίσης ο λόγος ύπαρξής του είναι ώστε μόλις το πατήσουμε να ανοίγει μια πύλη. Η πύλη αυτή είναι φτιαγμένη από μικρά τετράγωνα κυβάκια (tixalaki), τα οποία καταστρέφονται, επομένως η πύλη είναι πλέον ελεύθερη.



Εικόνα 2.30

Με ένα ενδεχόμενο collision του αντικειμένου μας με το αντικείμενο *filtro*, το score μας ανεβαίνει κατά τριάντα(30) πόντους, το filtro καταστρέφεται και ο χαρακτήρας μας μεταφέρεται στο δωμάτιο game 1.2.

Εικόνα 2.31

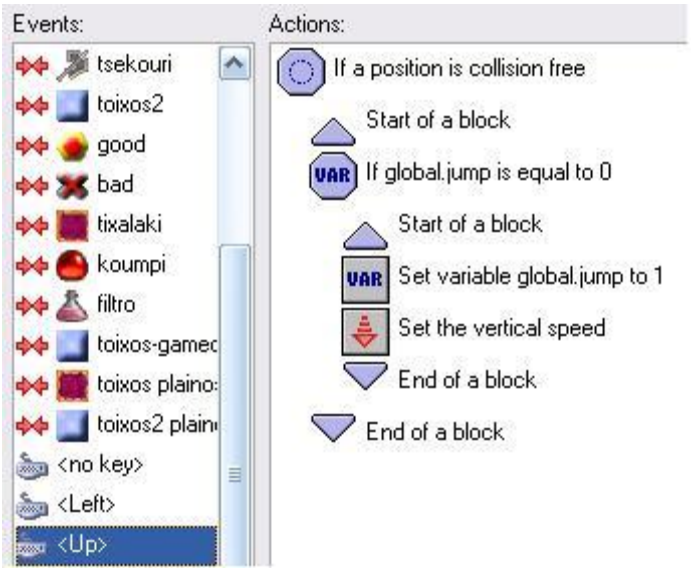
Όταν τώρα δεν πατάμε κάποιο πλήκτρο από το πληκτρολόγιο, ο χαρακτήρας μας πρέπει να μένει ακίνητος, με φορά την φορά της κατεύθυνσης που είχε πριν σταματήσει. Ως default είναι η δεξιά. Εάν για παράδειγμα κινούσαμε τον χαρακτήρα προς τα αριστερά, όταν σταματήσουμε θα μείνει ακίνητος «κοιτώντας» προς τα αριστερά.

Εικόνα 2.32

Με το πάτημα του αριστερού βέλους από το πληκτρολόγιο (αριστερό arrow key), ο χαρακτήρας μας κινείται προς τα αριστερά, με την προϋπόθεση ο χώρος ακριβώς δίπλα και αριστερά του να είναι ελεύθερος.

Εικόνα 2.33

Το ίδιο ακριβώς συμβαίνει και με το δεξί βέλος του πληκτρολογίου, με τις ανάλογες αλλαγές για την κίνηση του χαρακτήρα μας.

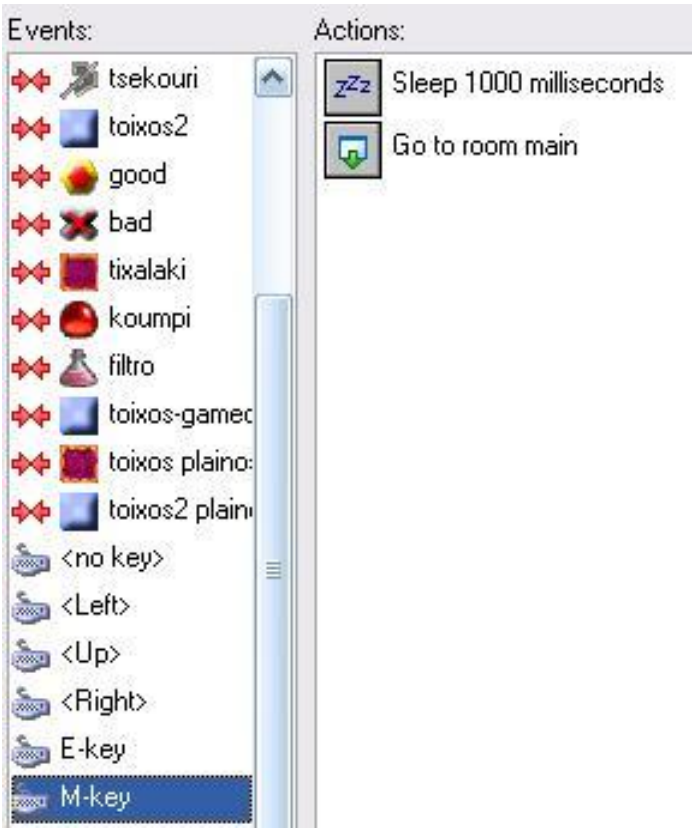


Εικόνα 2.34

Με το up arrow key (πάνω βελάκι) ο χαρακτήρας μας πραγματοποιεί ένα άλμα. Πριν πραγματοποιηθεί το άλμα αυτό γίνονται κάποιοι έλεγχοι. Πρώτον εάν ο χώρος ακριβώς πάνω από τον χαρακτήρα μας είναι ελεύθερος και δεύτερον εάν η μεταβλητή του άλματος έχει τιμή τέτοια που να επιτρέπει το άλμα. Εάν δηλαδή έχει τιμή ίση με το μηδέν(1) σημαίνει πως ο χαρακτήρας μας βρίσκεται ήδη σε άλμα χωρίς να έχει ακόμα προσγειωθεί.

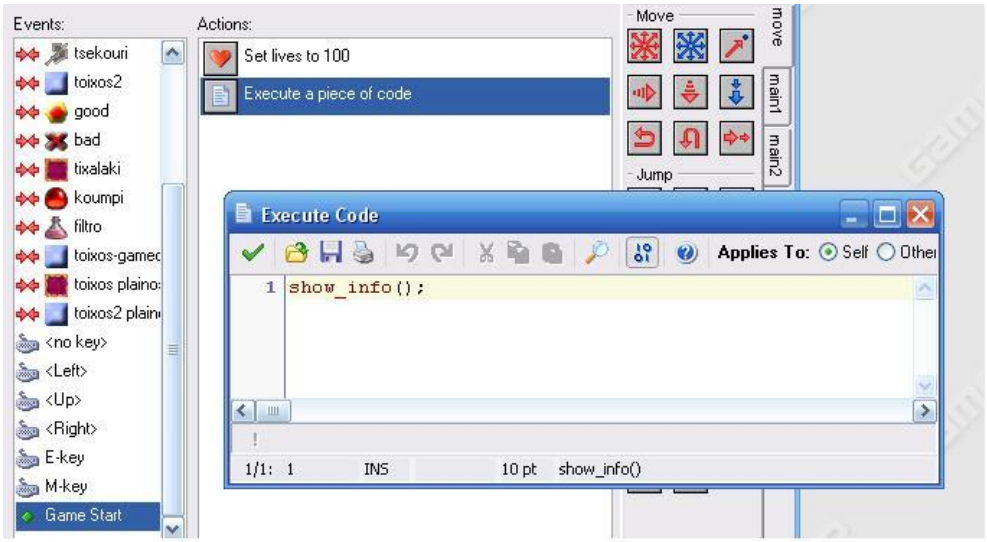
Εικόνα 2.35

Όταν πατήσουμε το πλήκτρο E του πληκτρολογίου μας, τότε αυτομάτως μεταφερόμαστε στο exit room, ακούγοντας τον ανάλογο ήχο εξόδου.



Εικόνα 2.36

Με το πλήκτρο M του πληκτρολογίου απλά μεταφερόμαστε στο main room του παιχνιδιού.



Εικόνα 2.37

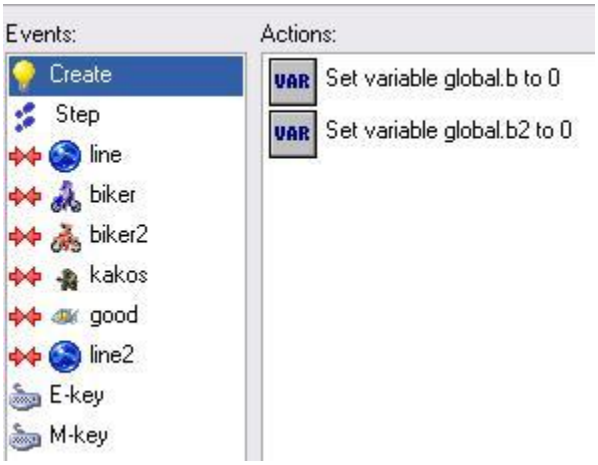
Τέλος, υπάρχει ένα ακόμη event το οποίο ενεργοποιείται μόλις ξεκινήσει το παιχνίδι. Το event αυτό έχει δύο σκοπούς. Ο πρώτος είναι η αρχικοποίηση του αριθμού των ζωών σε εκατό(100) και ο δεύτερος είναι η εμφάνιση του *Game Information* παραθύρου, το οποίο γίνεται με τη χρήση μιας πολύ απλής γραμμής κώδικα GML.

Αντικείμενο: **Player 1.2R**

Σε αυτό το αντικείμενο (για το room game 1.2) τα events και τα actions είναι τα ίδια με αυτά του αντικειμένου player 1R, όσο αφορά τα αντικείμενα good, bad, τοίχος πλαϊνός, και τα arrow keys (εδώ δεν χρησιμοποιείται το up arrow). Οι διαφορές είναι στην ταχύτητα με την οποία κινείται προς τα κάτω ο χαρακτήρας, όπου σε αυτήν την περίπτωση κινείται πιο αργά (vspeed 4 και όχι 12) , και στο collision με τον τοίχο που βρίσκεται στο δάπεδο του δωματίου (κάτω κάτω) το οποίο μας αποφέρει την μεταφορά του χαρακτήρα στο main room.

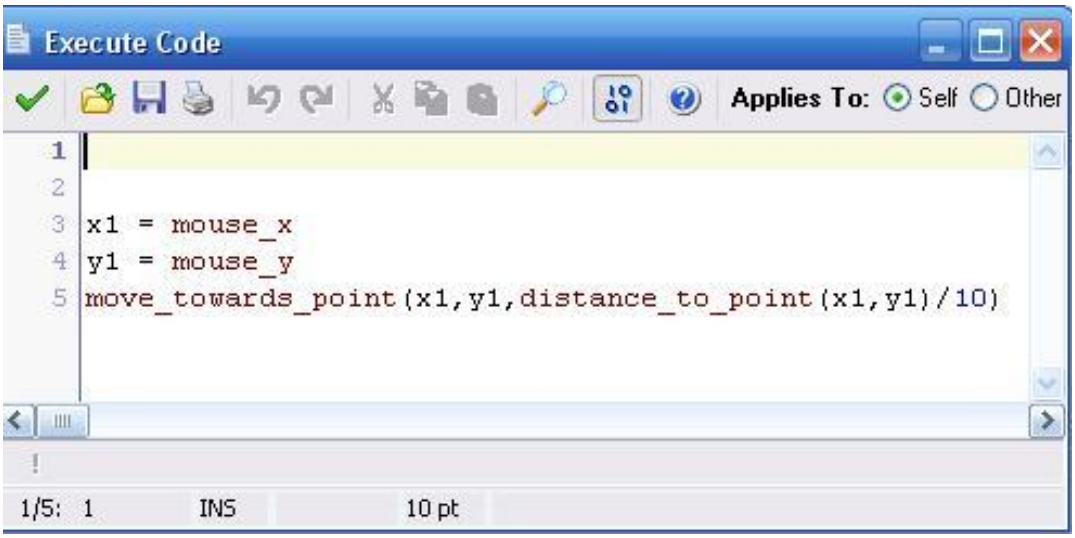
Αντικείμενο: **standR**

Το αντικείμενο αυτό αντιπροσωπεύει τον βασικό μας χαρακτήρα στο *game 2 room*. Τα actions με τα αντικείμενα line, line2, biker, biker2, kakos και good είναι αντίστοιχα με αυτά του player1R με τα αντικείμενα toixos1, tsekouri, good και τα λοιπά. Με τη μόνη διαφορά ότι σε ενδεχόμενη επαφή με το kakos του αφαιρούνται τρεις(3) ζωές και σε ενδεχόμενη επαφή με το good κερδίζει 2 πόντους στο score. Τα πλήκτρα E και M αποφέρουν τα ίδια αποτελέσματα επίσης.



Εικόνα 2.38

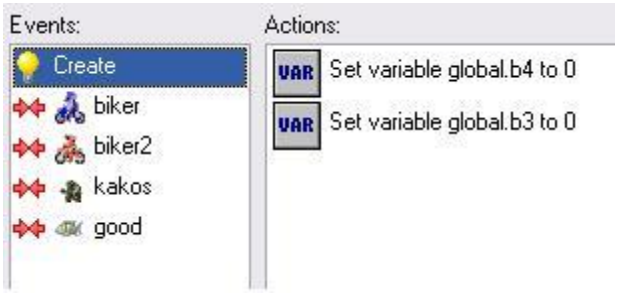
Μεγάλη διαφορά όμως με το αντικείμενο player1R υπάρχει στην κίνηση του χαρακτήρα μας. Το αντικείμενο standR κινείται με την χρήση του ποντικιού. Στην ουσία το object αυτό ακολουθεί τον δείκτη του ποντικιού. Αυτό έγινε εφικτό με την χρήση του παρακάτω κώδικα σε GML στο event Step :



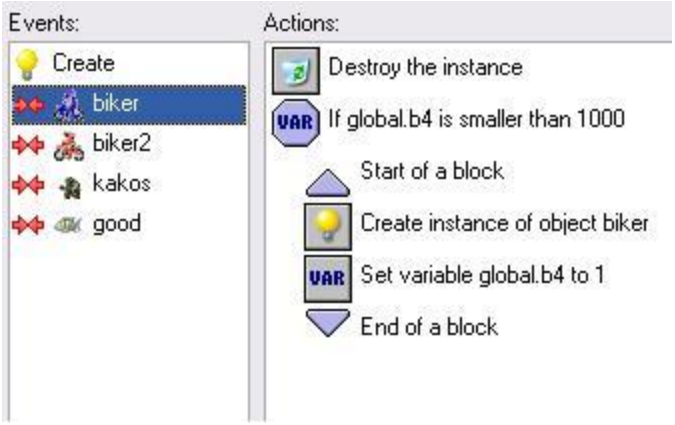
Εικόνα 2.39

Αντικείμενο: **line** και **line2**

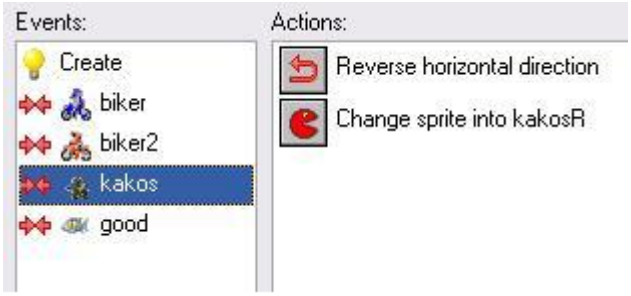
Τα αντικείμενα αυτά περιλαμβάνονται στο game 2 room, και αποτελούν το «περίβλημα» του δωματίου. Το line2 έχει μόνο δύο(2) events. Όταν τα αντικείμενα kakos και good έρθουν σε επαφή με αυτό, τότε εκείνα αλλάζουν πορεία και κατεύθυνση (φυσικά και sprite) . Το line περιλαμβάνει αυτά τα δύο events αλλά και κάποια άλλα. Αρχικοποιεί δύο(2) μεταβλητές που αφορούν τα αντικείμενα bike και bike2. Επίσης μόλις τα κάποιο από τα (αυτοκινούμενα) αντικείμενα bike ή bike2 έρθει σε επαφή με κάποιο από τα αντικείμενα line, τότε το πρώτο καταστρέφεται, ελέγχεται εάν η μεταβλητή που του αντιστοιχεί έχει μικρότερη τιμή από 1000 και αν ισχύει δημιουργείται ένα νέο αντικείμενο(bike ή bike2 ανάλογα) στο τέλος του δωματίου κατά μήκος και σε τυχαίο ύψος. Τέλος αυξάνεται η μεταβλητή κατά 1. Όλο αυτό έχει ως αποτέλεσμα την συνεχή δημιουργία αντικειμένων bike και bike2 για την ομαλή εξέλιξη και πορεία του παιχνιδιού.



Εικόνα 2.40



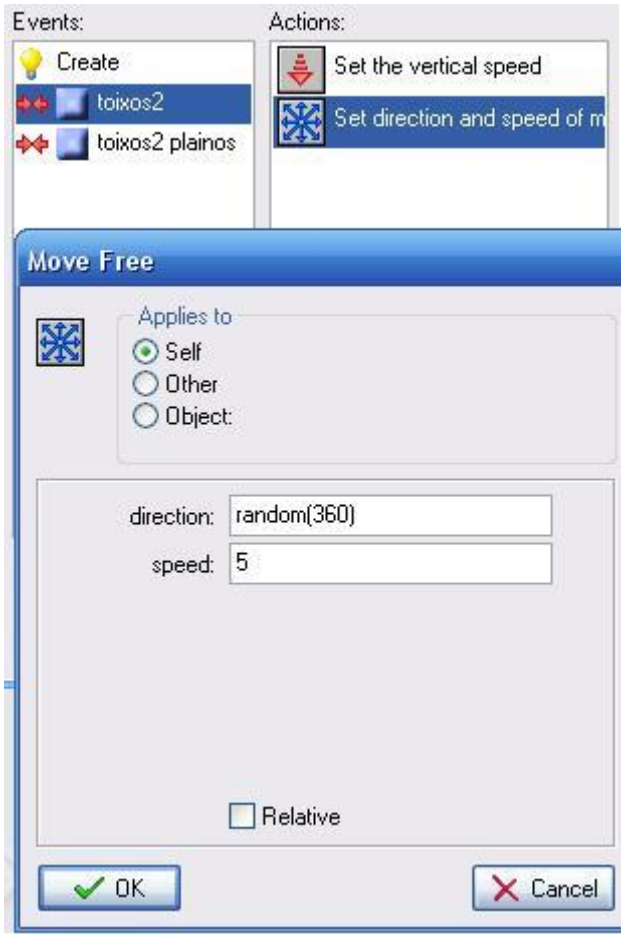
Εικόνα 2.41



Εικόνα 2.42

Αντικείμενο: **tsekouri**

Το αντικείμενο αυτό είναι ένα από τα αντικείμενα του *game 1* τα οποία αφαιρούν ζωές από τον χαρακτήρα μας. Έχει την ιδιαιτερότητα ότι κινείται μόνο του μέσα στο χώρο, σε τυχαία κατεύθυνση. Μόλις συναντήσει κάποιο εμπόδιο(κάποιον τοίχο) σταματάει για κλάσματα του δευτερολέπτου και ξεκινάει πάλι προς μια νέα τυχαία κατεύθυνση.

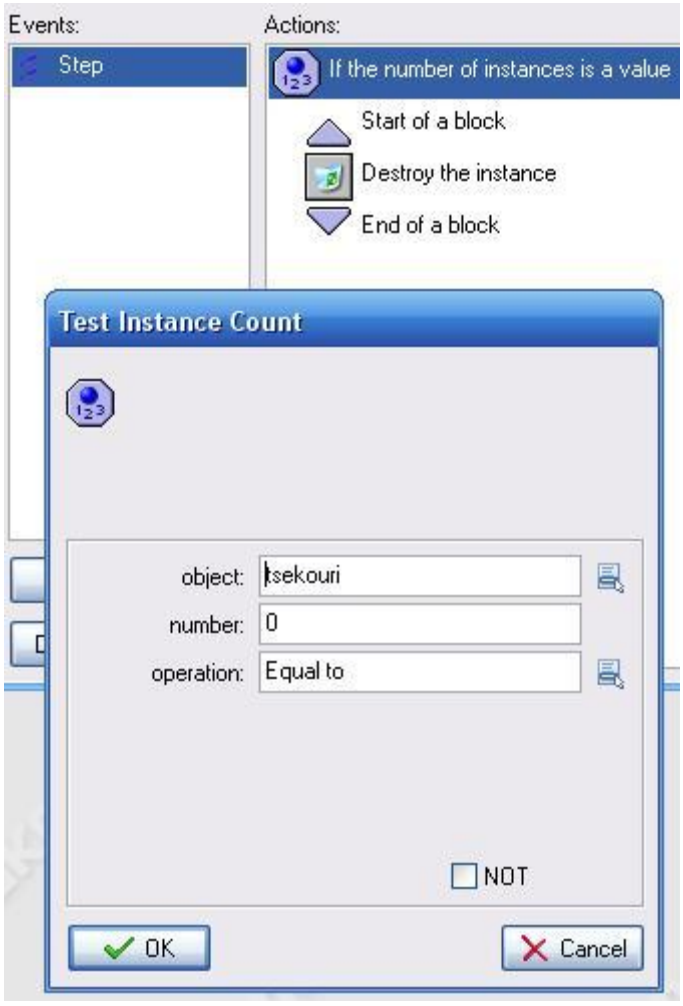


Εικόνα 2.43

Επίσης και τα αντικείμενα **biker**, **biker2**, good και **kakos** του game 2 κινούνται από μόνα τους αλλά προς μία συγκεκριμένη κατεύθυνση. Μάλιστα, τα good και kakos ανάλογα με το εμπόδιο που θα συναντήσουν αλλάζουν πορεία, αλλά πάντα παράλληλη με τον άξονα των x.

Αντικείμενο: **koumpi**

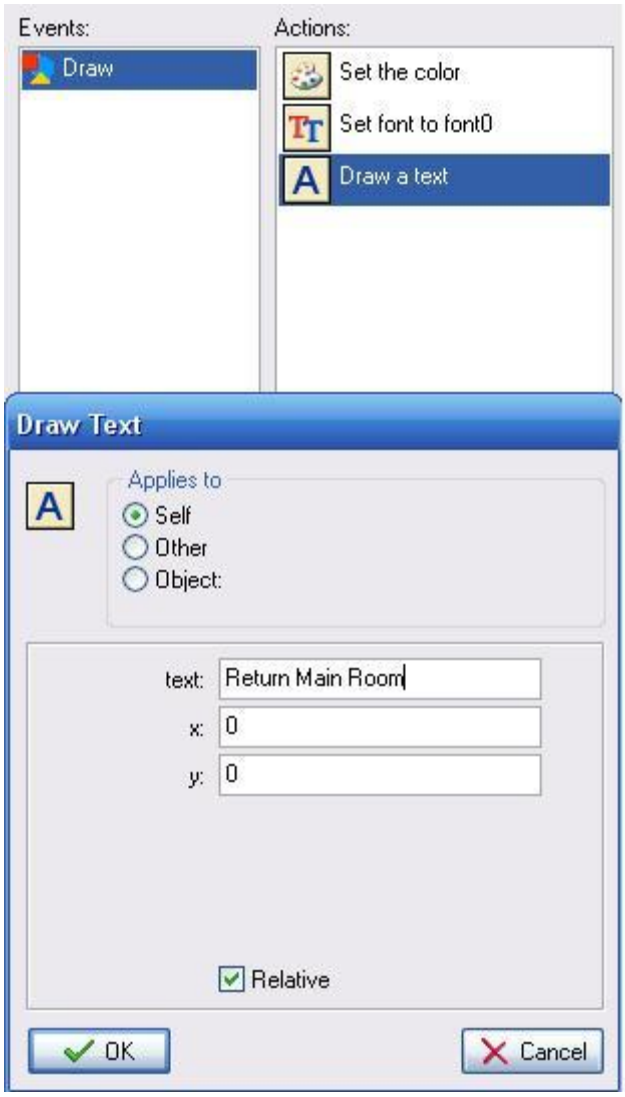
Όπως αναφέρεται και στις οδηγίες χρήσης του παιχνιδιού αλλά και μέσα στο παιχνίδι, εάν στο game 2 έρθεις σε επαφή με όλα τα αντικείμενα tsekouri, τότε το αντικείμενο koumpi εξαφανίζεται, επομένως η πύλη μένει κλειστή και δεν μπορείς τα πάρεις το αντικείμενο filtro. Αυτό γίνεται με τα παρακάτω actions της εικόνας :



Εικόνα 2.44

Τέλος, αναφέραμε και πιο πριν πως υπάρχουν αντικείμενα τα οποία δεν τα έχουμε αντιστοιχήσει με κάποιο sprite. Αυτά τα αντικείμενα είτε είναι τελείως «αόρατα» κατά την εκτέλεση του παιχνιδιού(όπως για παράδειγμα οι ήχοι) αλλά υπάρχουν στο δωμάτιο είτε έχουν την μορφή απλού κειμένου. Στην πρώτη περίπτωση απλά εισάγουμε ένα ήχο ενώ στην δεύτερη περίπτωση εισάγουμε το κείμενο που θέλουμε να απεικονίσουμε, το χρώμα και την γραμματοσειρά που θέλουμε να έχει.

Εικόνα 2.45


Events:
Draw
Actions:
Set the color
Set font to font0
Draw a text
Draw Text
Applies to
Self
Other
Object:
text:
Return Main Room
x:
0
y:
0
Relative
OK
Cancel

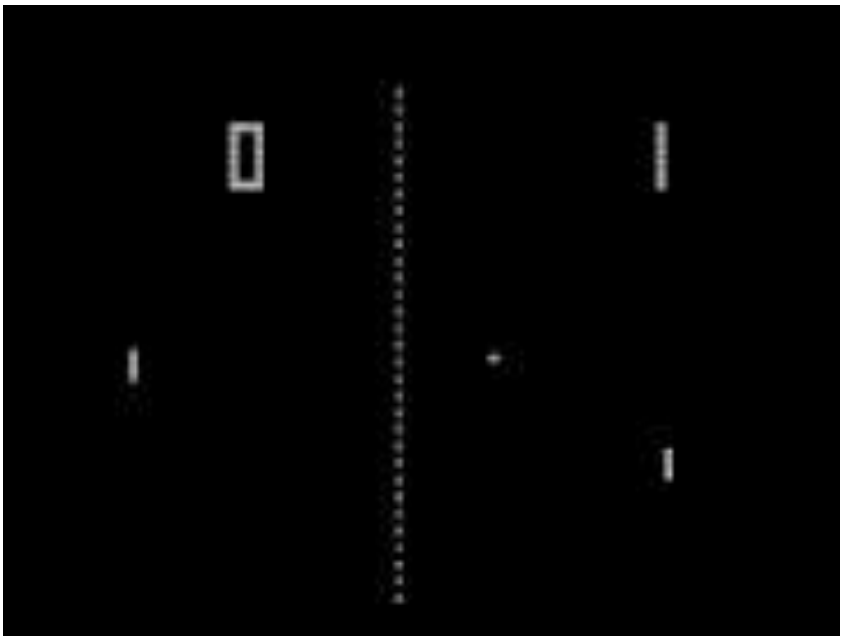
Εικόνα 2.46

ΚΕΦΑΛΑΙΟ 3
ΙΣΤΟΡΙΚΗ ΑΝΑΔΡΟΜΗ

Γνωρίζουμε, πως σε οτιδήποτε κι αν θέλουμε κάνουμε, και κυρίως όταν έχει σχέση με τον προγραμματισμό, την υλοποίηση κάποιου έργου και τα λοιπά, το πρώτο πράγμα θα που πρέπει να υλοποιήσουμε, είναι ένας καλός, σωστός και αντάξιος των απαιτήσεών μας αρχικός σχεδιασμός. Αυτό λοιπόν ισχύει και όταν θέλουμε να δημιουργήσουμε ένα ηλεκτρονικό παιχνίδι, άσχετα με την πλατφόρμα που θα χρησιμοποιήσουμε.

Υπάρχει μια σχετική «διαφωνία» για το ποιο πιστοποιείται ως το πρώτο ηλεκτρονικό (ψηφιακό) παιχνίδι που δημιουργήθηκε για τους υπολογιστές. Αυτό βέβαια μπορεί να απαντηθεί λαμβάνοντας υπόψη τον ορισμό που θα δώσει ο καθένας για τα «βιντεοπαιχνίδια». Αυτό το λέμε διότι παραδείγματος χάρη η λέξη «βίντεο» στη σύνθετη λέξη «βιντεοπαιχνίδι» αναφέρεται στη συσκευή απεικόνισης που χρησιμοποιείται για την χρήση του παιχνιδιού. Πλέον, με τη λέξη «βιντεοπαιχνίδι», λαμβάνουμε υπόψη μας και άλλους όρους και παραμέτρους, όπως τον τύπο της οθόνης που χρησιμοποιούμε, την πλατφόρμα και τα λοιπά.

Ως πρώτο βιντεοπαιχνίδι στην ιστορία των ηλεκτρονικών υπολογιστών πάντως έχει επικρατήσει το γνωστό ως «Pong» το οποίο εμφανίστηκε την δεκαετία του 60’.



Εικόνα 3.1

Το «Pong» αποτελούσε μια πάρα πολύ απλή εφαρμογή για τα σημερινά δεδομένα, αφού αποτελούνταν μόνο από κινούμενα τετραγωνάκια, τα γνωστά σε όλους μας pixels. Είναι ένα παιχνίδι, που πολλοί από εμάς ίσως να έχουν παίξει, στην πιο εξελιγμένη μορφή του φυσικά, όπου ο παίχτης κινεί με το ποντίκι του (όσο αφορά την έκδοσή του για ηλεκτρονικό υπολογιστή) ένα παραλληλόγραμμό αντικείμενο πάνω κάτω, και προσπαθεί έτσι να απωθήσει ένα μπαλάκι ώστε να μην περάσει την νοητή γραμμή πίσω από αυτό το αντικείμενο. Αλλά αντιθέτως προσπαθεί να στείλει αυτό το μπαλάκι ακριβώς απέναντί του, στην νοητή γραμμή του αντιπάλου, που στην προκειμένη περίπτωση αποτελεί μια πρώιμη μορφή Τεχνητής Νοημοσύνης, αφού ο αντίπαλός του είναι ο υπολογιστής.

Εικόνα 3.2

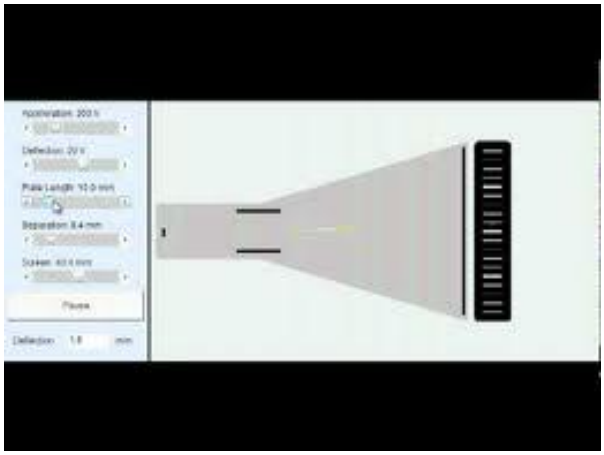
Να επισημάνουμε πως το συγκεκριμένο ηλεκτρονικό παιχνίδι, γνώρισε τεράστια επιτυχία, σε παγκόσμιο επίπεδο, αλλά και στην χώρα μας. Δεν άργησαν να εμφανιστούν στις καφετέριες τα γνωστά κουτιά ηλεκτρονικών παιχνιδιών που λειτουργούν με κέρματα, με το όνομα Atari-Pong, αφού η Atari ήταν και η πρώτη εταιρία που το έβγαλε σε αυτήν την μορφή. Επομένως είναι λάθος να πιστεύουμε πως το Packman, για το οποίο μας μιλάνε οι γονείς μας που έπαιζαν στις καφετέριες, είναι και το πιο παλιό ηλεκτρονικό παιχνίδι που υπήρχε σε αυτή τη μορφή.

Εικόνα 3.3

Για την ιστορία, να σημειώσουμε πως μπορεί το Pong έγινε γνωστό στην Ελλάδα μέσω της Atari, ο δημιουργός του όμως ήταν ο Γερμανός κάτοικος της Αμερικής Bayer (ο οποίος ήταν και ο δημιουργός της πρώτης κονσόλας ηλεκτρονικων παιχνιδιών, γνωστή ως Magnavox Odyssey) , στον οποίο η Atari πλήρωσε τα ανάλογα πρόστημα. Ο δημιουργός του παιχνιδιού αυτού εν τέλει τιμήθηκε από τον Τζορτζ Μπους με το μετάλλιο για τις τεχνολογικές του ανακαλύψεις και εφευρέσεις.

Όπως προείπαμε, υπάρχουν κάποιοι οι οποίοι αμφισβητούν την πρωτία του Pong, και θεωρούν ως «Αρχηγούς» των ηλεκτρονικών παιχνιδιών, κάποια άλλα δημιουργήματα.

Ένα από αυτά είναι και το «Cathode Ray Tube Amusement Device» , δημιούργημα του 1947 .



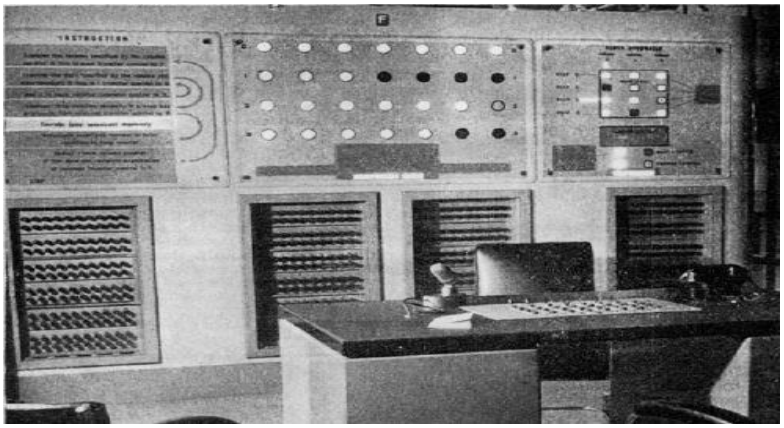
Εικόνα 3.4

Το παιχίδι αυτό αποτελούσε έναν προσομοιωτή πυραύλων, κάτι που εμπνεύστηκαν οι σχεδιαστές του από τις οθόνες ραντάρ του Δεύτερου Παγκοσμίου Πολέμου. Το παιχνίδι χρησιμοποιούσε κάποια αναλογικά κυκλώματα και όχι ψηφιακά, και μπορούσε κάποιος να ελέγξει την πορεία και την θέση της κουκίδας στην οθόνη.

Το ίδιο έτος, έγινε επίσης μια προσπάθεια για τη δημιουργία του πρώτου ηλεκτρονικου προσομοιωτή για παιχνίδια σκακιού (Chess) . Η προσπάθεια αυτή, αν και όχι εκατό της εκατό επιτυχής, έγινε από τους Alan Turing και Dietrich Prinz . Παρόλ'αυτά ήταν ένα ακόμη μεγάλο βήμα του ανθρώπου όσο αφορά την νοημοσύνη των μηχανών.

Εικόνα 3.5

Το 1951 γράφεται στην ιστορία ως το έτος που δημιουργήθηκε ο πρώτος υπολογιστής που φτιάχτηκε αποκλειστικά και μόνο για να παίζει ένα παιχνίδι. Ήταν στην ουσία ας πούμε η πρώτη κονσόλα ηλεκτρονικού παιχνιδιού, εάν μπορεί βέβαια να ονομαστεί έτσι λόγω των τεραστίων διαστάσεων που είχε. Το όνομα αυτής, NIMROD, και ο κατασκευαστής της ο Ferranti. Για την ιστορία, το παιχνίδι αυτό ονομαζόταν Nim.

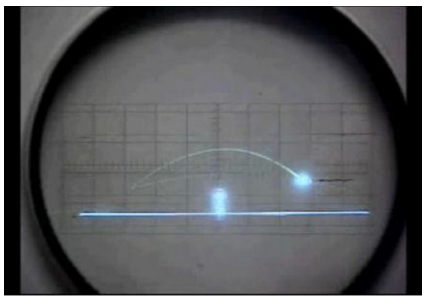


Εικόνα 3.6

Το 1952 έχουμε την εμφάνιση του OXO, ή αλλιώς Noughts and Crosses(Tic-Tac-Toe) , γνωστό στην Ελλάδα ως Τρίλιζα. Ο Alexander S. Douglas δημιούργησε για πρώτη φορά, μια ηλεκτρονική μορφή του παιχνιδιού «Τρίλιζα», εμφανίζοντας έτσι στη «σκηνή των ηλεκτρονικών παιχνιδιών» το πρώτο παιχνίδι στον υπολογιστή που χρησιμοποιούσε μια ψηφιακή οθόνη γραφικών. Επίσης η εφαρμογή αυτή πρωτοπορεί στο ότι μπορούσε να αποθηκευτεί. Για να παίξει κάποιος αυτό το παιχνίδι, χρησιμοποιούσε ένα περιστρεφόμενο μοχλό.

Εικόνα 3.7

Εξίσου σημαντική εφεύρεση επιτεύκτηκε το 1958, από τον William Higinbotham, ο οποίος κατασκεύασε έναν εξομοιωτή τένις, ή σύμφωνα με κάποιους έναν εξομοιωτή Ping-Pong, το «Tennis for two».



Εικόνα 3.8

Όπως προείπαμε, οι απόψεις διίστανται για το ποίο παιχνίδι πρέπει να θεωρηθεί ως το πρώτο ηλεκτρονικό παιχνίδι για υπολογιστή. Αυτό θα το κρίνει ο καθένας προσωπικά…

ΕΠΙΛΟΓΟΣ

Φτάνοντας λοιπόν στο τέλους αυτού του πονήματος, πιστεύω πως αναλύθηκαν επαρκώς τα στοιχεία και οι λεπτομέρειες που χρειάζεται να γνωρίζει κάποιος ώστε να μπορέσει να εισχωρήσει επιτυχώς στον κόσμο της δημιουργίας ηλεκτρονικών παιχνιδιών με το Game Maker. Μπορεί οι δυνατότητες του εργαλείου αυτού να είναι πολύ μεγάλες και να μην τις αξιοποιήσαμε πλήρως, αλλά πιστεύω πως με τις πληροφορίες που πήραμε μπορεί ο καθένας να τις ανακαλύψει από μόνος του.

Όπως είναι λογικό πριν φτάσουμε σε εφαρμογές προσωπικού ηλεκτρονικού υπολογιστή για την δημιουργία παιχνιδιών, όπως το Game Maker, έχει μεσολαβήσει μια ολόκληρη ιστορία που έχει να κάνει με τα ηλεκτρονικά παιχνίδια. Νομίζω πως είναι ενδιαφέρουσες οι αναφορές που έγιναν σε αυτήν την ιστορία και κινεί το ενδιαφέρον όλων μας.

Εν κατακλείδι, ελπίζω η εργασία αυτή να προκαλέσει το ενδιαφέρον των αναγνωστών της, να βοηθήσει αυτούς που έχουν την όρεξη να ασχοληθούν εκτενέστερα με το θέμα (πέρα από μια ανάγνωση) και να αποτελέσει ένα βασικό κομμάτι για την είσοδό τους σε αυτόν τον χώρο.

ΒΙΒΛΙΟΓΡΑΦΙΑ

ΒΙΒΛΙΑ

- Habgood J. , Overmars M. (2006**) The Game Maker's Apprentice: Game Development for Beginners**
- Lee Ford J. (2009) **Getting Started with Game Maker**
- Swamy Nanu, Swamy Naveena (2006) **Basic Game Design & Creation for Fun & Learning**

E-BOOKS

- Αγγελιδάκης Ν. (2012) **Δημιουργία παιχνιδιού με το Game Maker 8.1 Lite**
- Bailey M. (2011) **Using Game Maker 8**
- Overmars M. (2004) **Designing Games with Game Maker *version* 5.3**
- Trollsplatterer (2008) **Game Maker Language**

WEB PAGES

- http://wiki.yoyogames.com
- http://gamemaker.info/en/manual
- http://www.gamemakerlanguage.herobo.com/gmltutorial.php?page=begtut1
- http://asterakiaa.blogspot.gr/2012/02/blog-post_8811.html
- http://gadget-akia.blogspot.gr/2010/11/video-games.html
- http://en.wikipedia.org/wiki/Game_Maker_Language
- http://www.gmlscripts.com/gml/language_overview