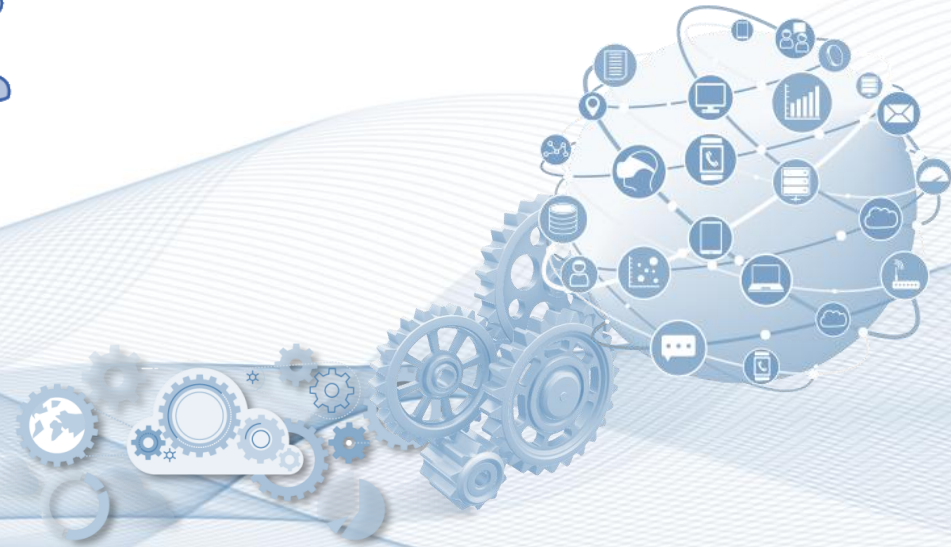


ΤΕΧΝΙΚΕΣ ΠΡΟΓΡΑΜΜΑΤΙΣΜΟΥ ΥΠΟΛΟΓΙΣΤΩΝ

Διάλεξη I

Scratch - Ξεκινώντας τον Προγραμματισμό

Γιάννης Τζήμας, Καθηγητής



Τι θα δούμε σήμερα...

- Ο πρώτος μας **στόχος** είναι να πάρετε μία πρώτη ιδέα σε σχέση με:
 - τον **Προγραμματισμό** και την **Επιστήμη των Υπολογιστών**.
- Θα γνωρίσουμε το **Scratch** και θα δούμε τις **γενικές αρχές**:
 - της **Αλληλουχίας** των εντολών,
 - των **Γεγονότων** (events),
 - της **Επιλογής** και
 - της **Επανάληψης**.



Το πρώτο πράγμα που πρέπει πάντα να κάνετε... είναι να ρωτάτε το γιατί;

- **Γιατί** να ασχοληθείτε με την **Πληροφορική**;
 - Για να ενισχύσετε τη **λογική** στον τρόπο που σκέφτεστε (logical thinking).
 - Για να επεκτείνετε το **μαθηματικό** σας υπόβαθρο.
 - Για να κερδίσετε **βαθύτερη γνώση** σε σχέση με την **τεχνολογία**:
 - Πώς να τη χρησιμοποιείτε.
 - Πώς να δημιουργείτε με αυτή.
 - Πώς να την αξιοποιείτε για υπολογισμούς.



Γιατί αρχές προγραμματισμού, όταν υπάρχει το AI;

- Τα εργαλεία Τεχνητής Νοημοσύνης γράφουν πλέον κώδικα γρήγορα και εύκολα.
- Κάποιος όμως πρέπει να τον διαβάσει, να τον ελέγξει και να αποφασίσει αν κάνει αυτό που πρέπει.
- Αυτός ο κάποιος είστε εσείς! Γι' αυτό χρειάζεται να καταλαβαίνετε τις αρχές του προγραμματισμού:
 - αλληλουχία, επιλογή, επανάληψη, μεταβλητές, αφαίρεση...
- Οι αρχές αυτές δεν αλλάζουν από γλώσσα σε γλώσσα, ούτε από εργαλείο σε εργαλείο.



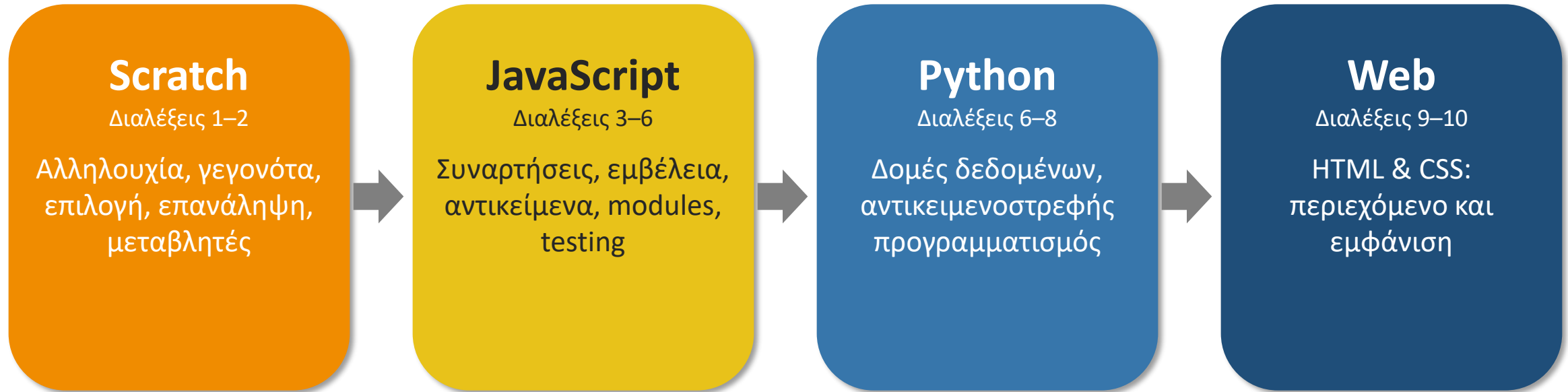
Γιατί βρεθήκαμε όλοι μαζί εδώ;

- Για να μάθουμε να **προγραμματίζουμε**.
- Για να το κάνουμε αυτό χρειαζόμαστε μία γλώσσα προγραμματισμού, ώστε να εκφράσουμε τις εντολές μας προς τον υπολογιστή.

Ποια γλώσσα όμως είναι η **σωστή**;



Η απάντηση: καμία... και όλες!



Ίδιες αρχές, σε τρεις γλώσσες.

Ο στόχος δεν είναι να μάθετε μία γλώσσα, αλλά να μάθετε να σκέφτεστε σαν προγραμματιστές.

Τι θα χρησιμοποιήσουμε για αρχή;

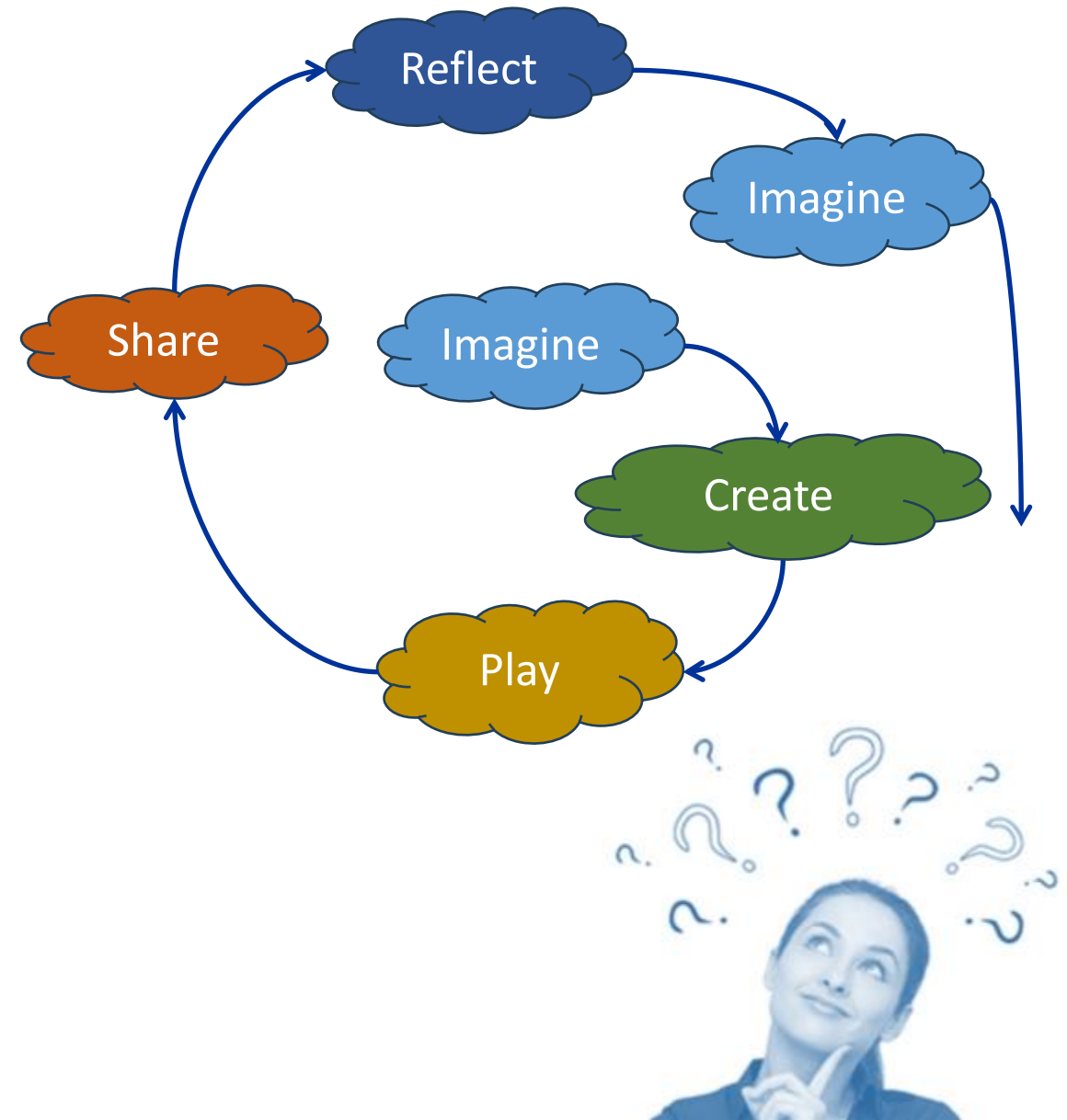
- Αναπτύχθηκε από μερικούς πολύ έξυπνους ανθρώπους @ MIT
- Λειτουργεί σε Linux, Mac, Windows και πλέον σε οποιαδήποτε συσκευή μέσω ενός Web Browser.

SCRATCH



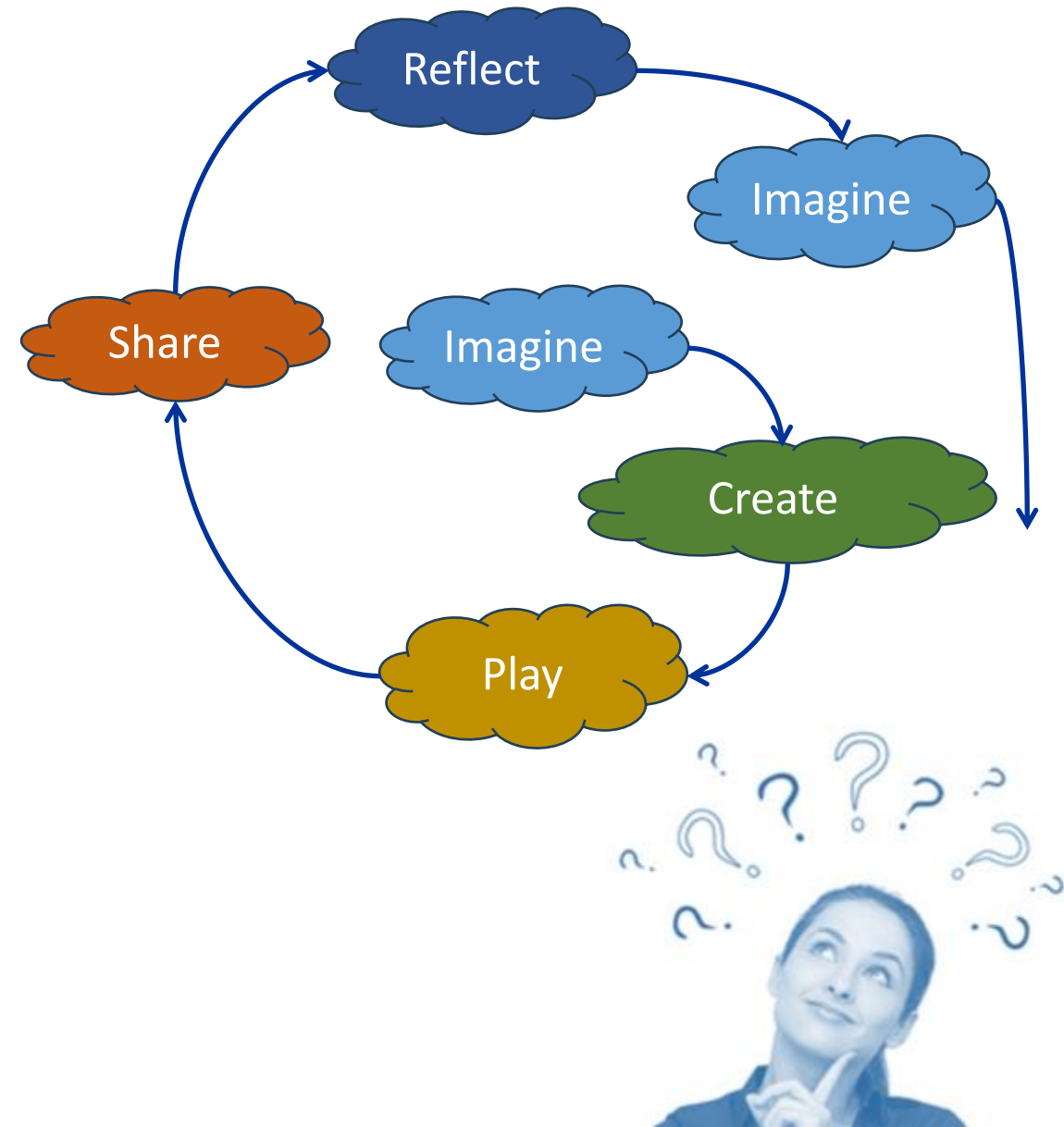
Γιατί Scratch; (1/2)

- Εντυπωσιακό
- Δωρεάν
- Χρησιμοποιείται από εκατομμύρια ανθρώπους παγκόσμια



Γιατί Scratch; (2/2)

- **Visual, drag & drop** (λιγότερη πληκτρολόγηση, περισσότερες δοκιμές)
- Μείξη διαφορετικών μέσων (γραφικά, ήχος και βίντεο) “**scratching**”
- **Διαμοίραση** και **συνεργασία**
- **Low floor, high ceiling, and wide walls**
 - **Εύκολο** από την αρχή.
 - Δεν είναι παιχνίδι – μπορεί να λύσει σύνθετα προβλήματα.
 - **Επεκτάσιμο** σε διάφορα πεδία εφαρμογών.



Εγκατάσταση!

- <http://scratch.mit.edu>
- Για να παίζετε στο PC σας μπορείτε να κατεβάσετε την offline από εδώ:
<https://scratch.mit.edu/download>

 Δημιούργησε Εξερεύνησε Ιδέες Σχετικά Αναζήτηση Εγγραφή Σύνδεση

Scratch Desktop

Μπορείτε να εγκαταστήσετε τον επεξεργαστή Scratch Desktop για να εργαστείτε σε έργα χωρίς σύνδεση στο διαδίκτυο. Αυτή η έκδοση λειτουργεί σε Windows και MacOS.



Απαιτήσεις

 Windows 10+  macOS 10.13+

Επιλέξτε το λειτουργικό σας σύστημα:  

Install Scratch Desktop

1

Download Scratch Desktop

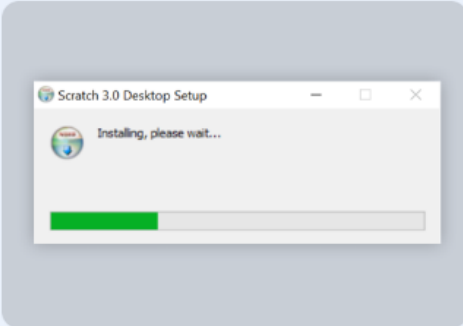
 Get it from Microsoft

or

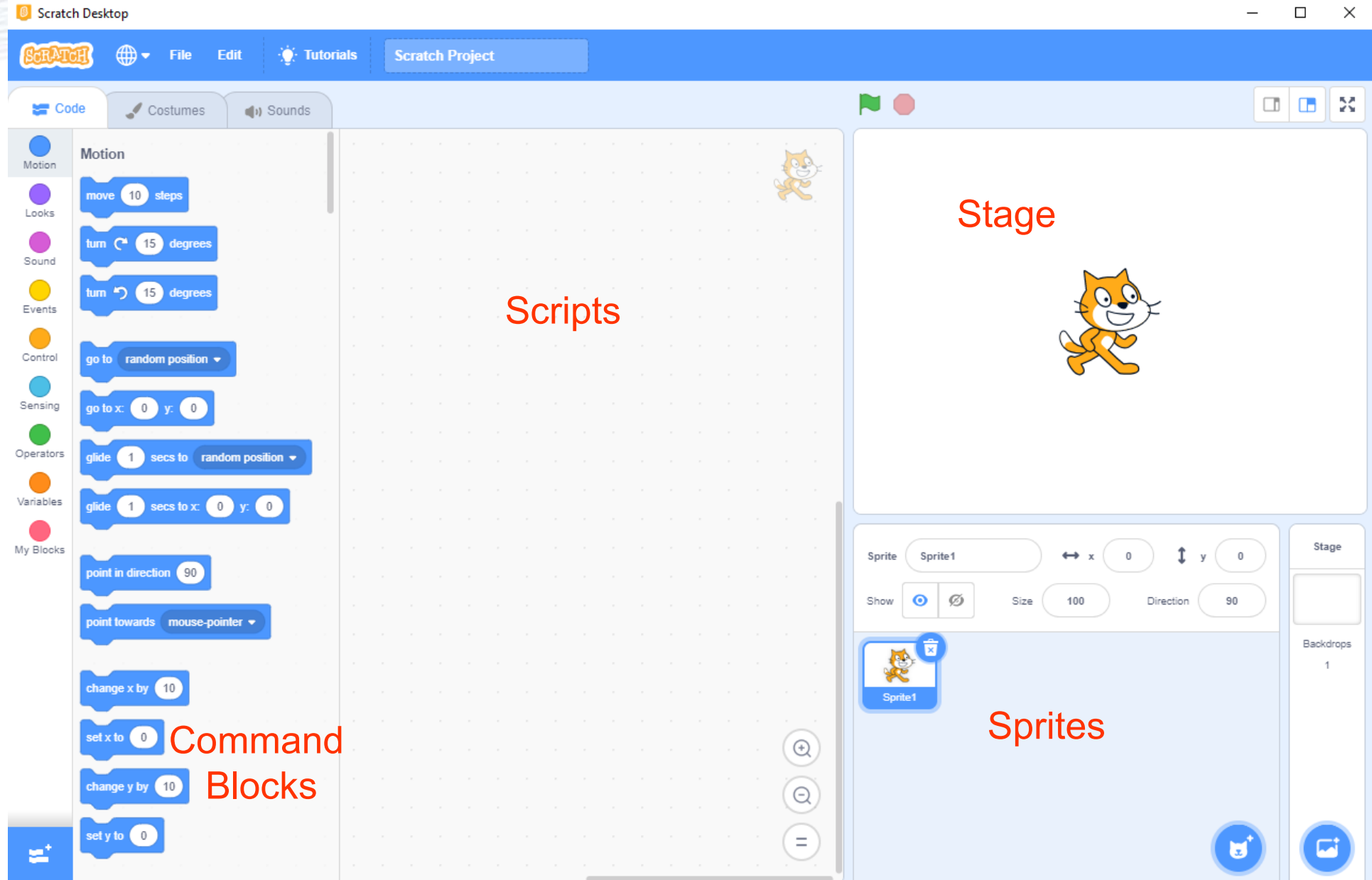
[Direct download](#)

2

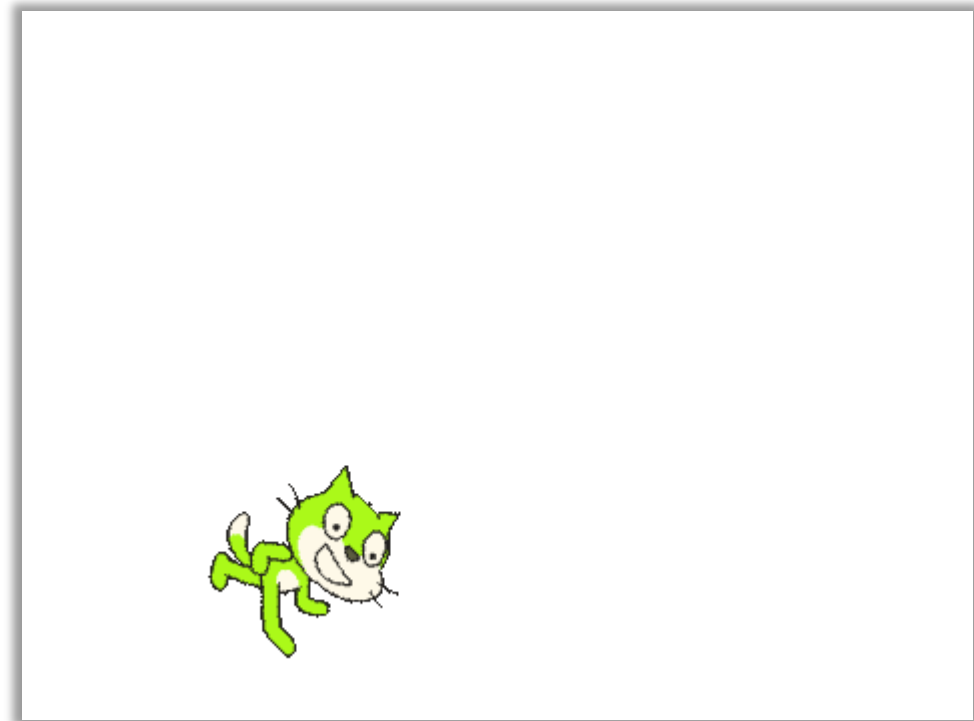
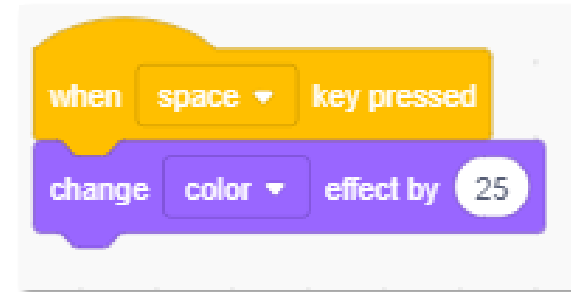
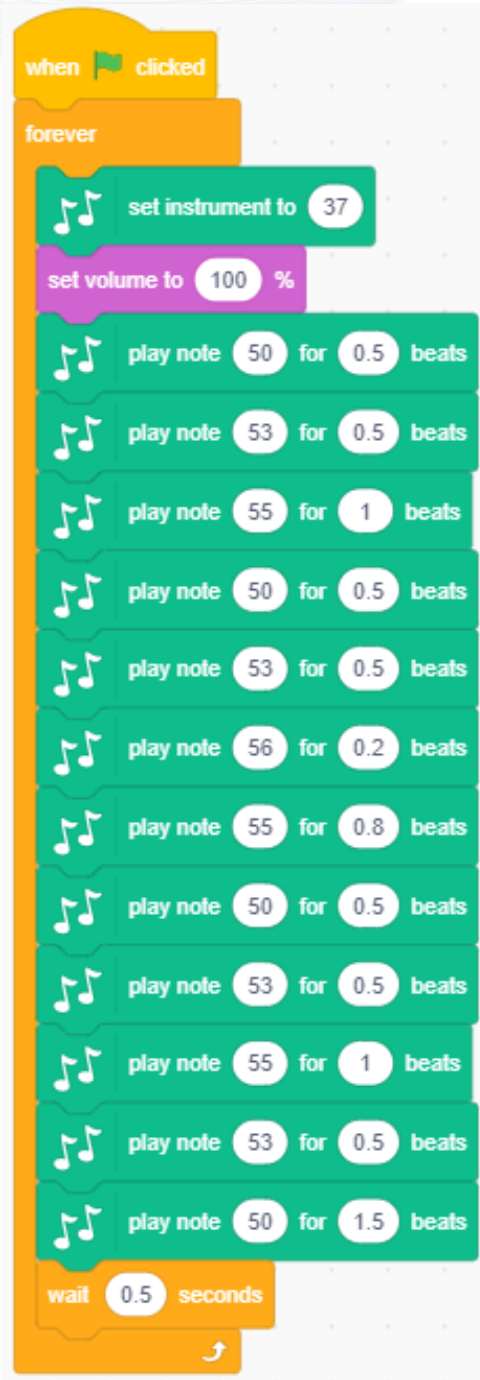
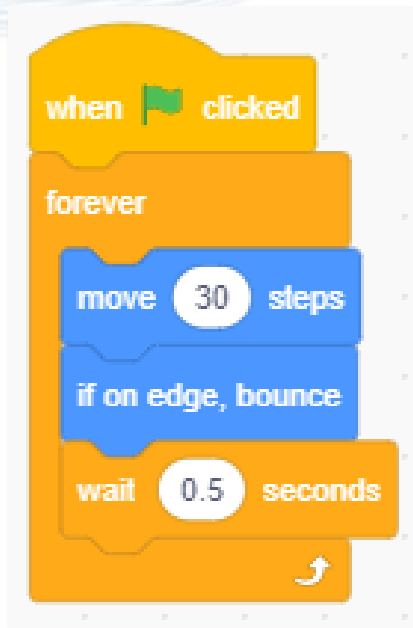
Εκτελέστε το αρχείο .exe.



...and Scratch!!!



Ένα μικρό demo...



Ένα μικρό demo...

Scratch Desktop

Scratch File Edit Tutorials lecture1_demo1

Code Costumes Sounds

Motion

- move 10 steps
- turn 15 degrees
- turn 15 degrees
- go to random position
- go to x: 0 y: 0
- glide 1 secs to random position
- glide 1 secs to x: 0 y: 0
- point in direction 90
- point towards mouse-pointer
- change x by 10
- set x to 0
- change y by 10
- set y to 0

when clicked

- forever loop:
 - move 30 steps
 - if on edge, bounce
 - wait 0.5 seconds

when space key pressed

- change color effect by 25

when clicked

- forever loop:
 - set instrument to 37
 - set volume to 100 %
 - play note 50 for 0.5 beats
 - play note 53 for 0.5 beats
 - play note 55 for 1 beats
 - play note 50 for 0.5 beats
 - play note 53 for 0.5 beats
 - play note 56 for 0.2 beats
 - play note 55 for 0.8 beats
 - play note 50 for 0.5 beats
 - play note 53 for 0.5 beats
 - play note 53 for 0.5 beats
 - play note 55 for 1 beats
 - play note 53 for 0.5 beats
 - play note 50 for 1.5 beats
 - wait 0.5 seconds

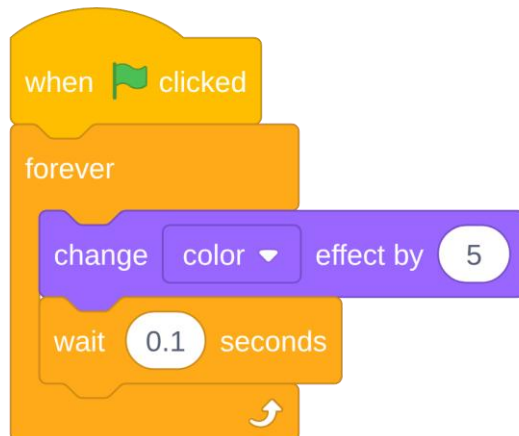
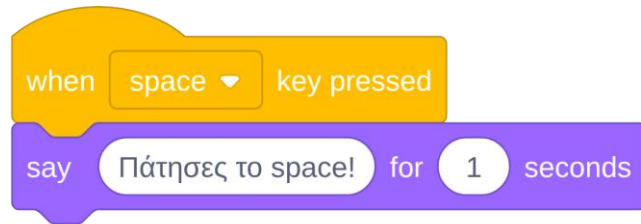
Stage

Sprite1

Sprite2

Backdrops 1

Πώς ξεκινάει ένα πρόγραμμα;

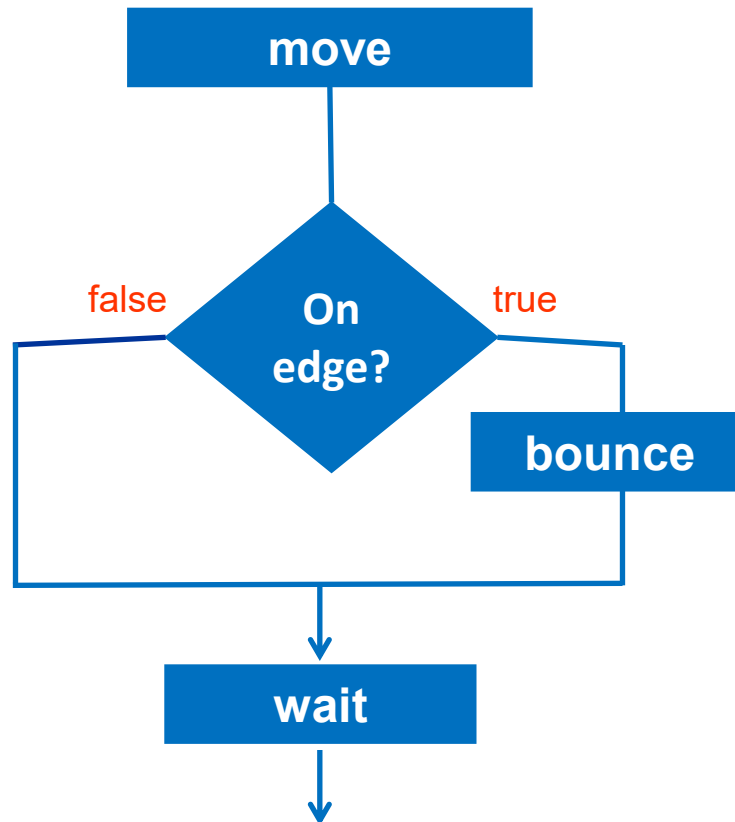


- **Γεγονότα (Events)**: τα blocks με «καπέλο» δεν κάνουν κάτι μόνα τους, **περιμένουν ένα γεγονός**.
- Όταν συμβεί το γεγονός (σημαία, πλήκτρο, κλικ...), εκτελείται το script που βρίσκεται από κάτω.
- Ένα sprite μπορεί να έχει **πολλά scripts που τρέχουν ταυτόχρονα**: εδώ η περιστροφή και η αλλαγή χρώματος γίνονται μαζί.
- Θα το ξαναβρούμε στη JavaScript: στο Web σχεδόν όλα ξεκινούν από γεγονότα (clicks, πλήκτρα, αιτήματα).

Τι είδαμε μέχρι τώρα;

- Στην επίλυση προβλημάτων με υπολογιστή υπάρχουν **τρεις** δεξιότητες τις οποίες θα πρέπει να καλλιεργήσετε:

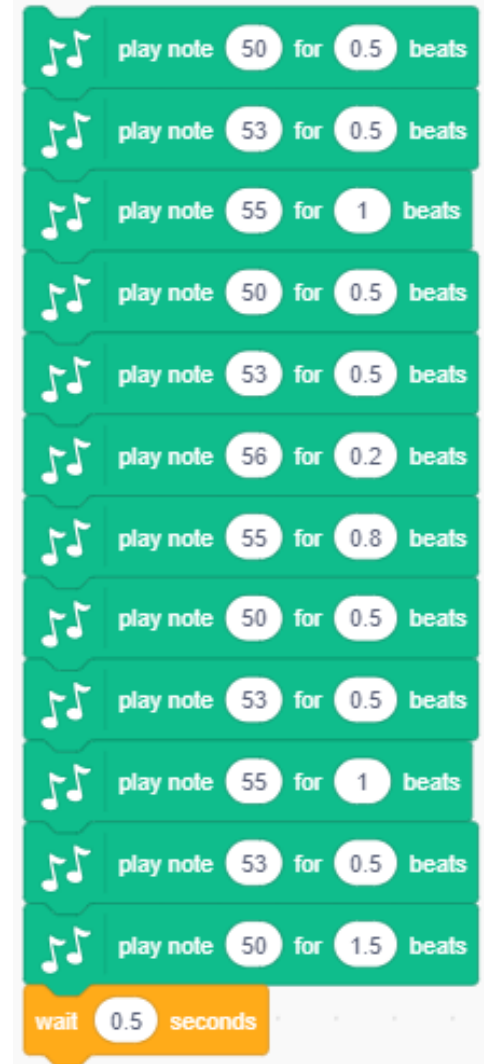
(2) Επιλογή ανάμεσα σε διάφορες εναλλακτικές



(3) Επανάληψη

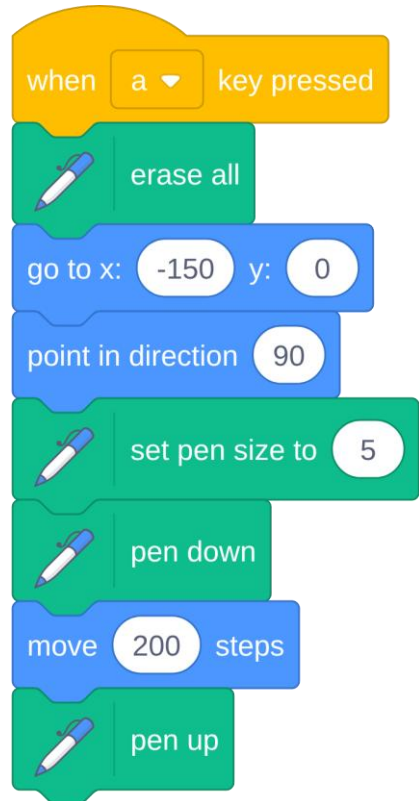


(1) Αλληλουχία
των εντολών

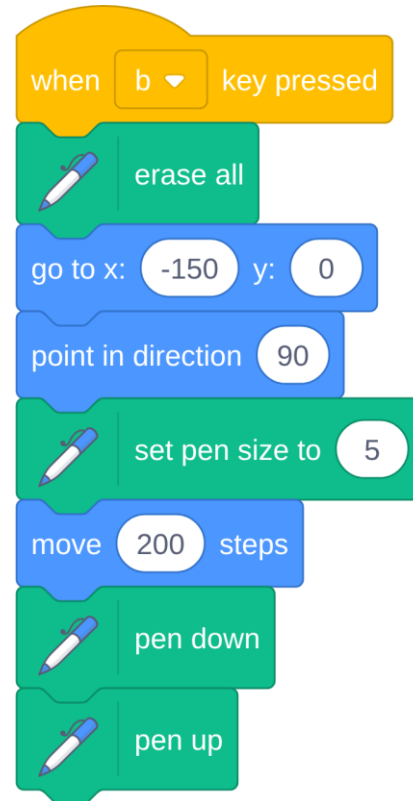


Γενική Αρχή #1 – Αλληλουχία

Script A



Script B



- **Αλληλουχία (sequence)**: οι εντολές εκτελούνται **μία-μία**, με τη **σειρά** που είναι γραμμένες: από πάνω προς τα κάτω.
- Η σειρά **έχει σημασία**: τα δύο scripts έχουν ακριβώς τις ίδιες εντολές, αλλά σε άλλη σειρά.

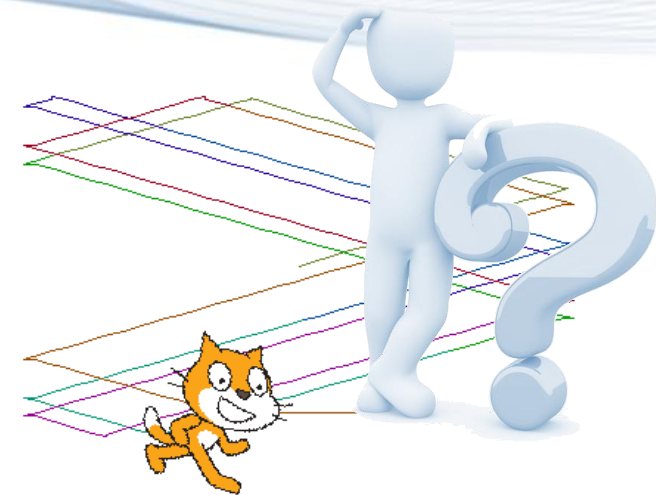
Ποιο από τα δύο θα ζωγραφίσει γραμμή;

Μόνο το **A**!

Στο B η πένα κατεβαίνει **αφού** έχει γίνει η κίνηση, οπότε ζωγραφίζει μόνο μια τελεία.

Και λίγη δουλειά για το σπίτι...

- Κατεβάστε και εγκαταστήστε το Scratch:
<http://scratch.mit.edu/download>
- Μπείτε στο <http://scratch.mit.edu> και δείτε μερικά projects από άλλους...
- Και μερικές **ασκήσεις**:
 - Δημιουργήστε ένα πρόγραμμα όπως αυτό που κάναμε μαζί χρησιμοποιώντας ένα διαφορετικό sprite image και βάζοντας τη δικιά σας μουσική (ίσως το αγαπημένο σας τραγούδι). Το πρόγραμμα θα πρέπει να τρέχει για πάντα εκτός και εάν το σταματήσει ο χρήστης.
 - Δημιουργήστε ένα πρόγραμμα το οποίο αρχικά τοποθετεί το sprite στη θέση (0,0), δείχνει σε μία τυχαία κατεύθυνση, κατεβάζει την πένα (pen down) και ξεκινάει να μετακινείται. Όταν φτάνει σε κάποια άκρη θα πρέπει να αλλάζει κατεύθυνση. Επίσης, χρησιμοποιήστε το πλήκτρο SPACE για να αλλάζετε το χρώμα της πέννας. Το αποτέλεσμα θα πρέπει να μοιάζει με τη διπλανή εικόνα:

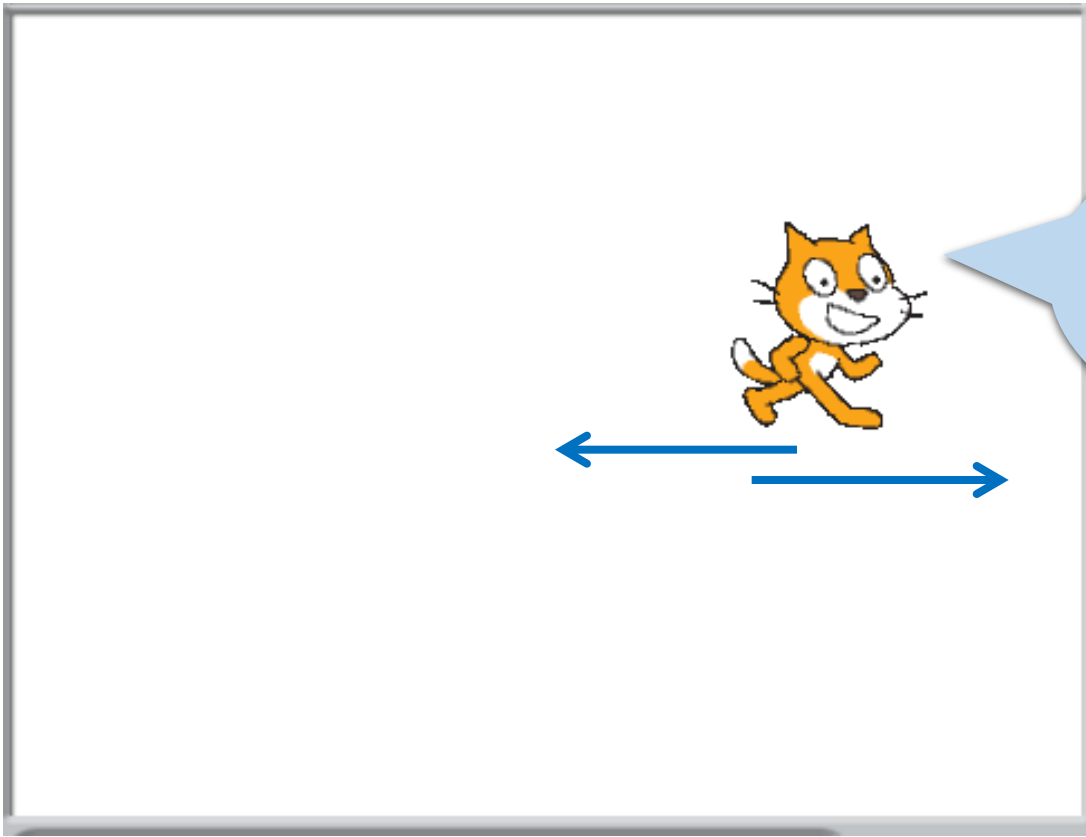


Στο Scratch 3 η πένα είναι **επέκταση**: «Add Extension» (κάτω αριστερά) → «Pen».



Θέση = Συντεταγμένες

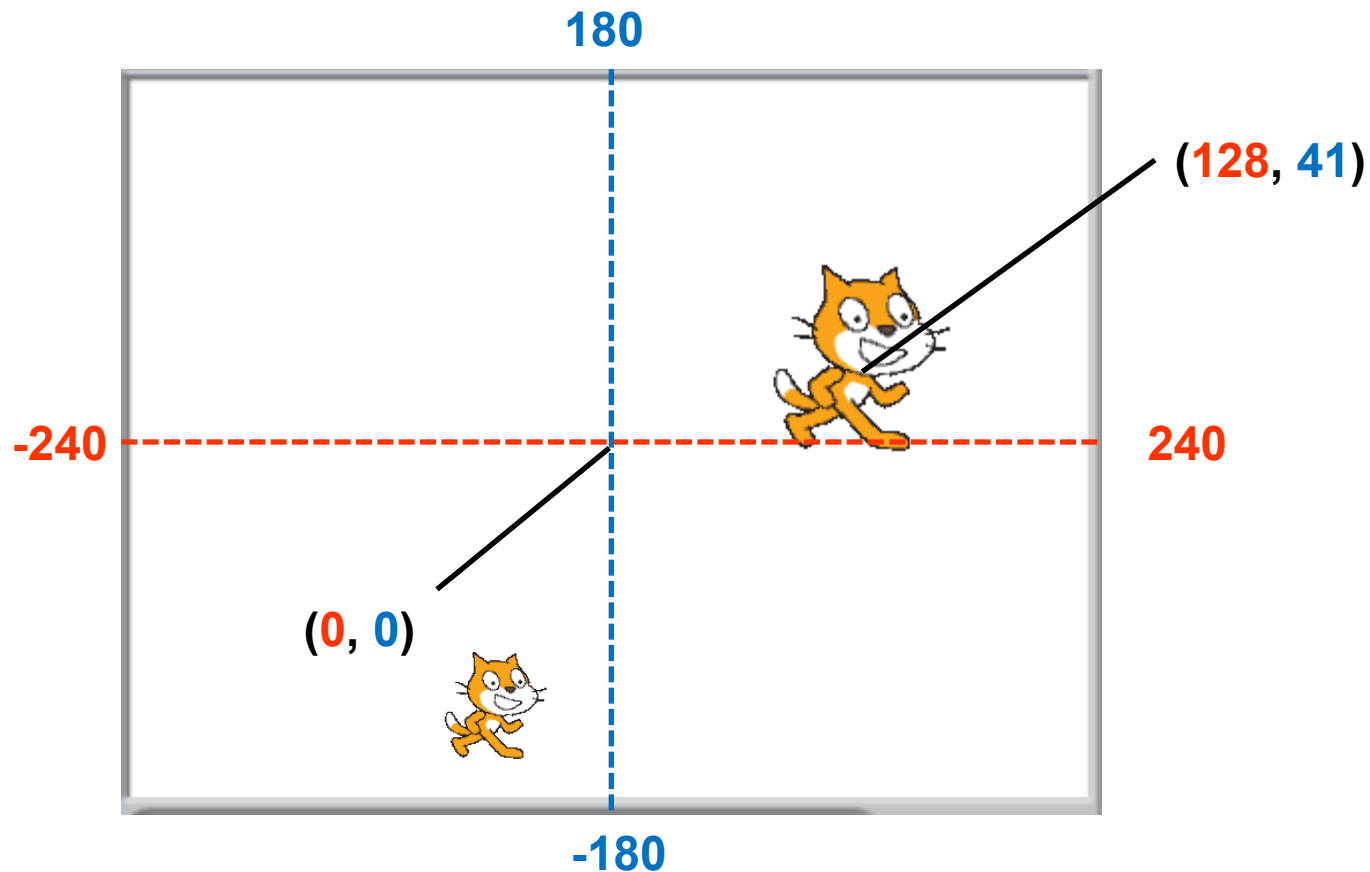
- Η **θέση** ενός sprite ορίζεται από δύο αριθμούς, τις **συντεταγμένες** (x, y): βήματα **δεξιά/αριστερά** και **πάνω/κάτω** από το κέντρο.
- **Γενική αρχή:**
 - Ο υπολογιστής **δεν μαντεύει** τι εννοούμε.
 - Οι εντολές μας πρέπει να είναι **σαφείς και ακριβείς**.



Όταν λέμε μετακινήσου
αριστερά εννοούμε το
δικό μας αριστερά, ή του
sprite;

Το Καρτεσιανό Σύστημα Συντεταγμένων

- Χρησιμοποιούμε **θετικούς** και **αρνητικούς** αριθμούς για τις ακριβείς συντεταγμένες.



Blocks εντολών σχετικά με Συντεταγμένες

go to random position ▾

go to x: 0 y: 0

glide 1 secs to random position ▾

glide 1 secs to x: 0 y: 0

change x by 10

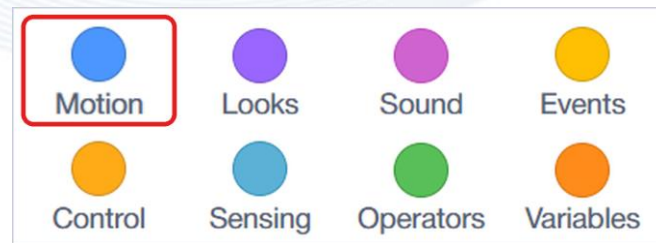
set x to 0

change y by 10

set y to 0

x position

y position



- Πήγαινε στο (x,y)
- Διολίσθησε-glide στο (x,y)
- Άλλαξε το x ή το y κατά...
- Όρισε το x ή το y ίσο με...

go to x: 0 y: 0

glide 1 secs to x: 100 y: 100

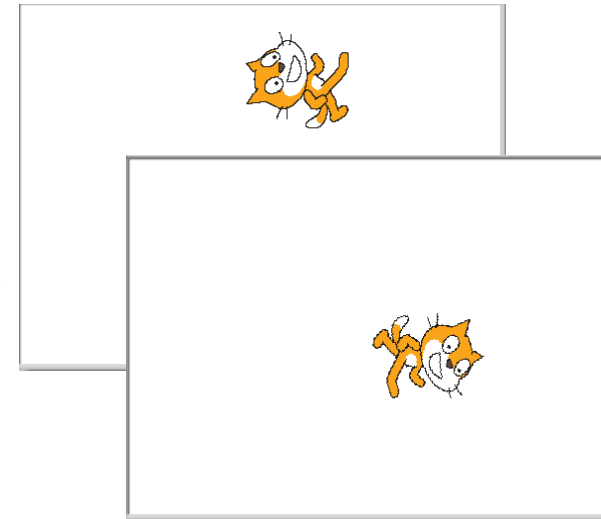
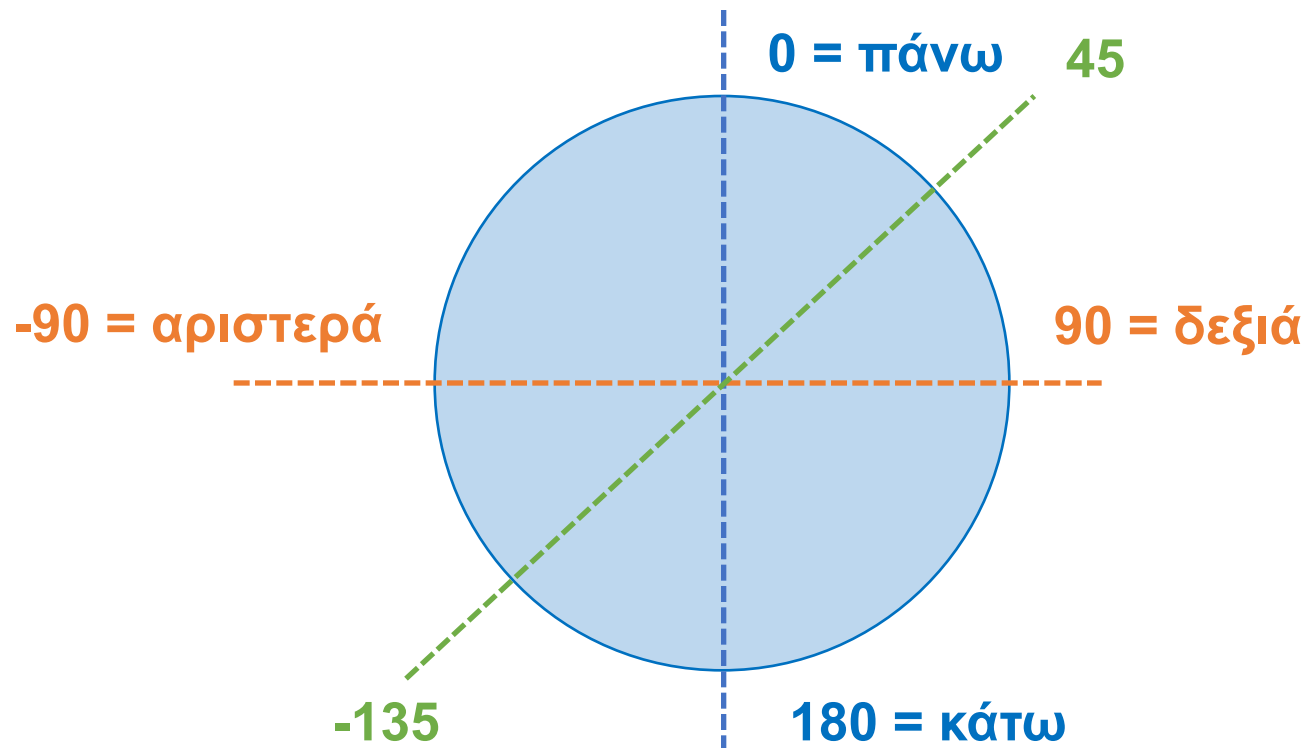
change x by 200

change y by 200



Κατεύθυνση - Direction

- Τα sprites μπορούν να πάρουν διάφορες κατευθύνσεις.
 - Όπως σε μια πυξίδα έτσι και εδώ, οι κατευθύνσεις αναπαρίστανται σε μοίρες, αλλά...



Blocks εντολών σχετικά με Κατεύθυνση

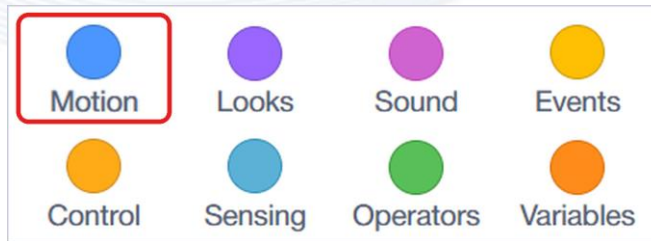
turn  15 degrees

turn  15 degrees

point in direction 90

point towards mouse-pointer ▼

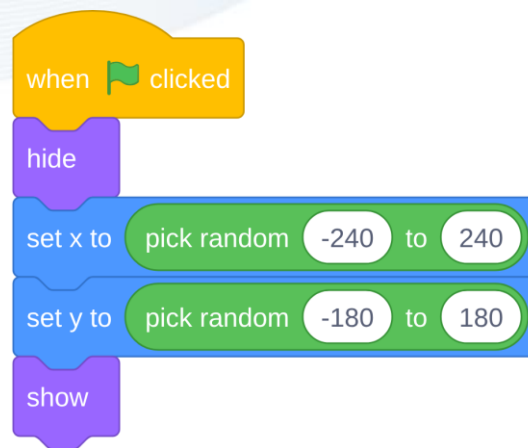
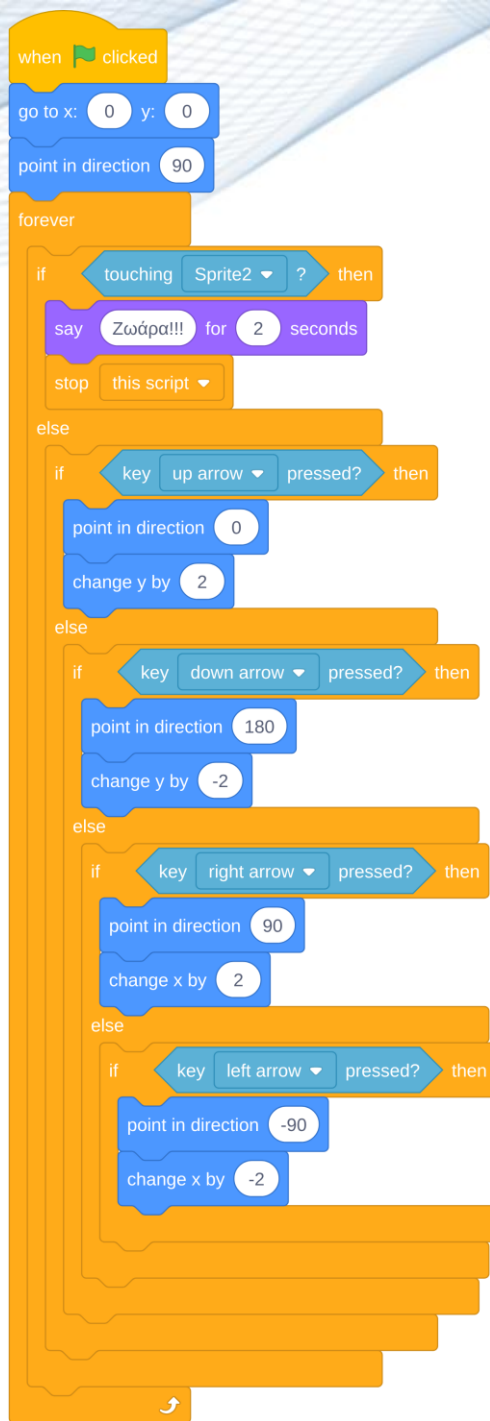
direction



- Στρίψε
 - δεξιόστροφα
 - αριστερόστροφα
 - προς την κατεύθυνση των...
 - προς...



Ένα μικρό demo...



Demos_2026-2027/lecture1_02.sb3



Τι είδαμε μέχρι τώρα;



- Είδαμε το **Καρτεσιανό Σύστημα Συντεταγμένων**.
Με τα (x,y) μπορούμε να ορίσουμε **θέσεις**.
- Είδαμε πως μπορούμε να καθορίσουμε την **κατεύθυνση** χρησιμοποιώντας **μοίρες**.
 - Σε αντίθεση με τις πυξίδες που έχουμε 0...360 μοίρες, στο Scratch έχουμε 0...180 και 0...-180
- Μέχρι στιγμής χρειαστήκαμε για ακόμα μία φορά κάθε μία από τις παρακάτω δεξιότητες:
 - **Αλληλουχία** των εντολών
 - **Επιλογή** ανάμεσα σε διάφορες εναλλακτικές
 - **Επανάληψη**

Και λίγη δουλειά για το σπίτι...



- Μερικές ασκήσεις:
 - Δημιουργήστε ένα πρόγραμμα όπως αυτό που κάναμε μαζί βάζοντας τη δικιά σας μουσική. Προσοχή, θα πρέπει να αντικαταστήσετε το “stop script” με ένα “stop all” block για να σταματήσει και ο ήχος. Στη συνέχεια επεκτείνετε το πρόγραμμα ώστε όταν πατάτε το πλήκτρο spacebar το μπολ να πηγαίνει σε νέα θέση.
 - Μετατρέψτε το πρόγραμμα σε παιχνίδι μετατρέποντας τον κώδικα στο μπολ. Το μπολ θα πρέπει να κινείται συνεχώς, ώστε η Scratch να πρέπει να το κυνηγήσει. Για αρχή κάντε την κίνηση απλή. Το μπολ μπορεί να πηγαίνει μόνο οριζόντια στον άξονα X.

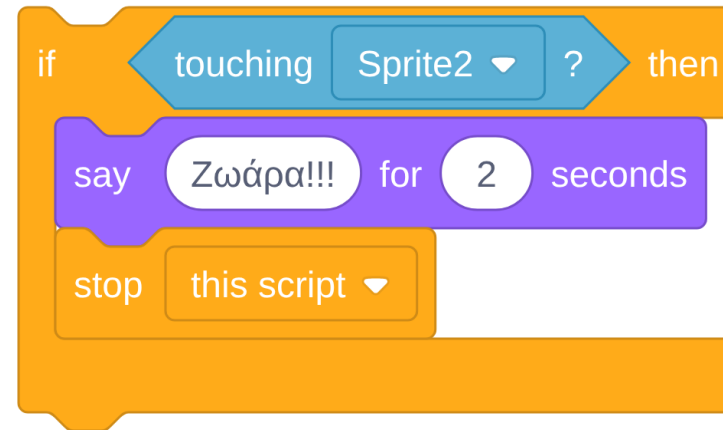
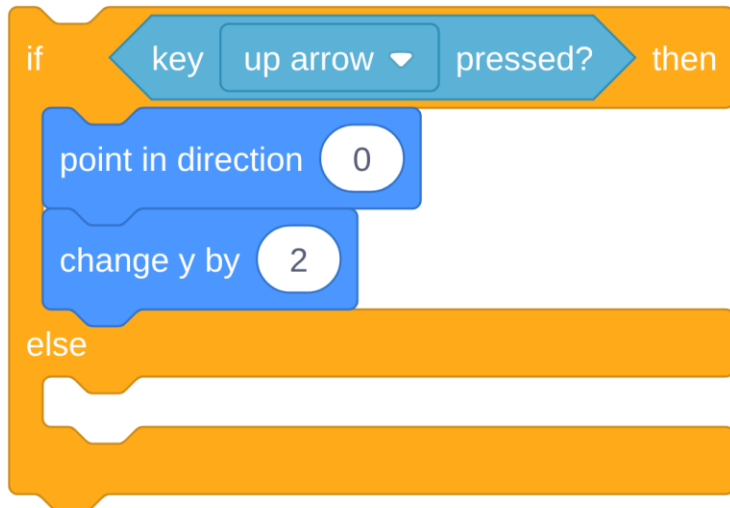
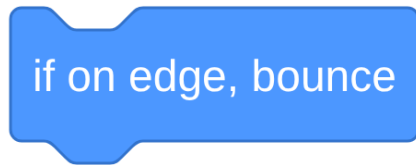
Επιλογή - Selection

- Συχνά χρειάζεται να **επιλέξουμε (select)** το τι θα κάνουμε βασιζόμενοι σε μία, ή περισσότερες **συνθήκες (conditions)**.
- Στην πραγματική ζωή:
 - **AN (IF)** πεινάσω **TOTE (THEN)** θα φάω κάτι.
 - **AN (IF)** δε βρέξει **TOTE (THEN)** θα κάνω ποδήλατο **ΑΛΛΙΩΣ (ELSE)** θα κοιμηθώ.
 - **AN (IF)** ο καθηγητής ρωτήσει μια ερώτηση **ΚΑΙ (AND)** ξέρω την απάντηση **TOTE (THEN)** θα σηκώσω το χέρι.



Η Επιλογή στο Scratch

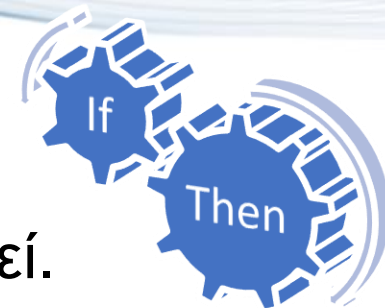
- Έχουμε ήδη δει μέχρι τώρα την διαδικασία της επιλογής **πολλές φορές**.



← έλεγξε τα υπόλοιπα κουμπιά κατευθύνσεων...

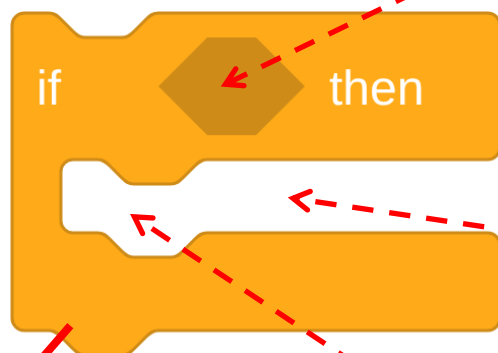
Πολλά if-else το ένα μέσα στο άλλο (όπως για τα βελάκια) λέγονται **else if** στις περισσότερες γλώσσες. Θα τα ξαναδούμε στη JavaScript.

Γενική Αρχή #2



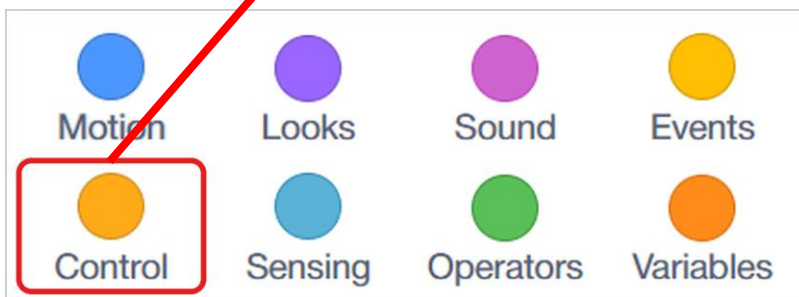
- Το **If-Then**

Συνθήκη που θα πρέπει να ελεγχθεί.

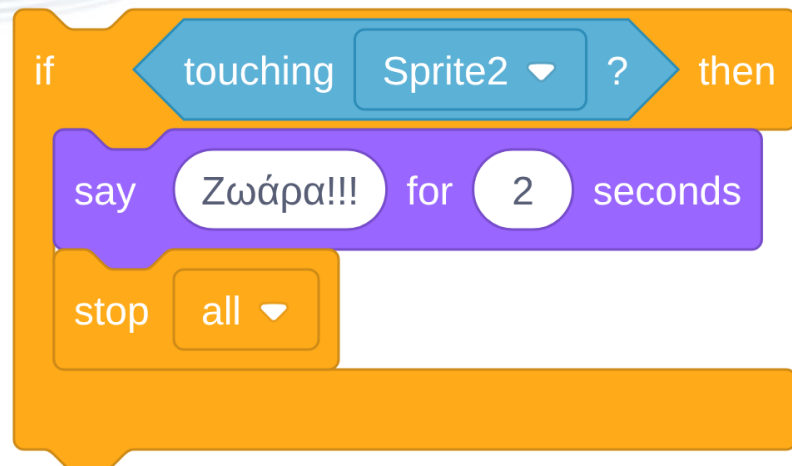


Εντολή/ές που θα πρέπει να εκτελεστούν «Αν» (if) η συνθήκη ισχύει (true).

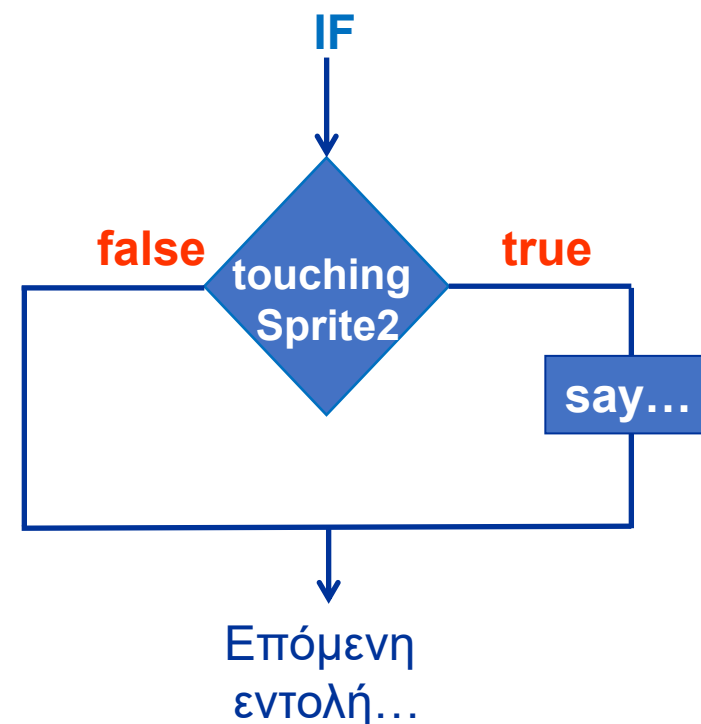
Αναφερόμαστε στις εμπειριεχόμενες εντολές σαν το τμήμα «**Τότε**» (then).



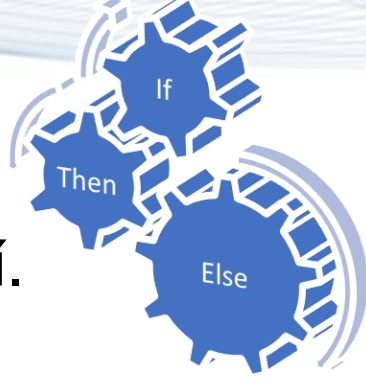
Μοντέλο Εκτέλεσης - Ένα Παράδειγμα...



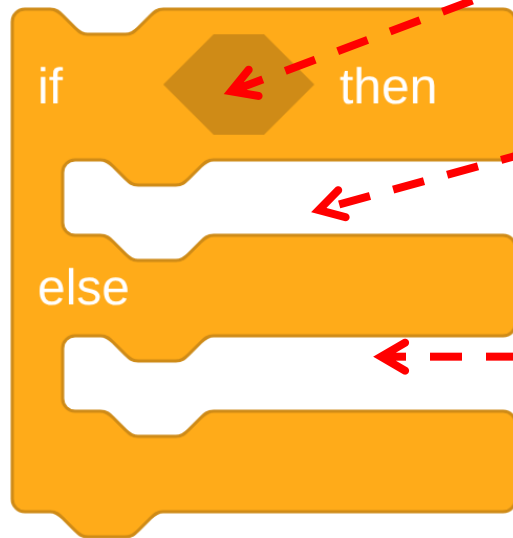
Τι θα συμβεί αν η συνθήκη δεν ισχύει;
Απολύτως τίποτα – Αυτό το τμήμα παραλείπεται.



Γενική Αρχή #3



- Το **If-Then-Else**



Συνθήκη που θα πρέπει να ελεγχθεί.

Εντολή/ές που θα πρέπει να εκτελεστούν «Αν» (if) η συνθήκη ισχύει (true).

«Τότε» (then)

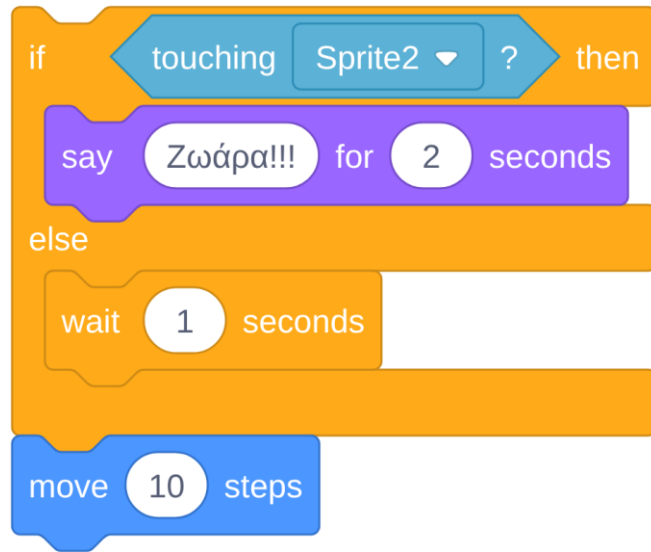
Εντολή/ές που θα πρέπει να εκτελεστούν «Αν» (if) η συνθήκη δεν ισχύει (false).

«Αλλιώς» (else)

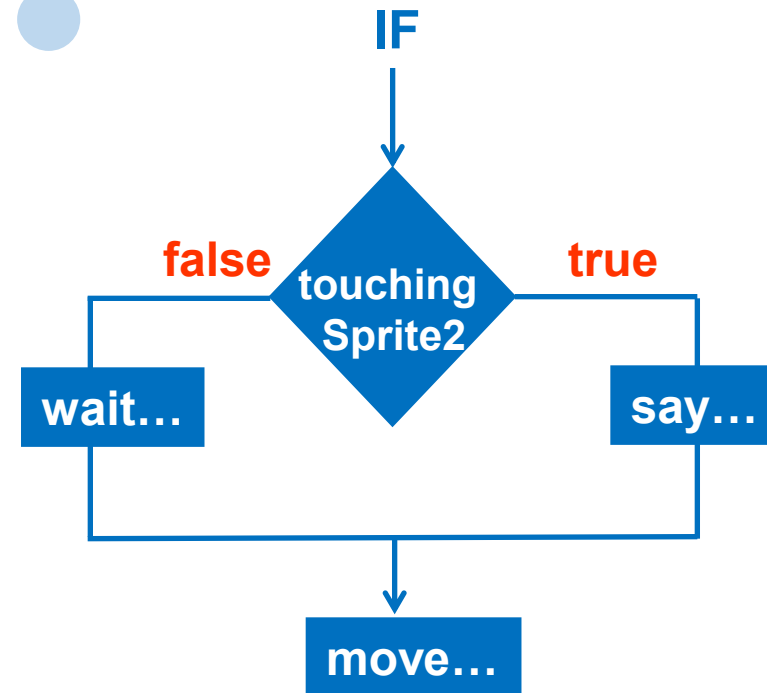
- **Πότε** το χρησιμοποιούμε;

- Όταν υπάρχουν ενέργειες που μπορούν να εκτελεστούν και στις δύο περιπτώσεις...(true, false)

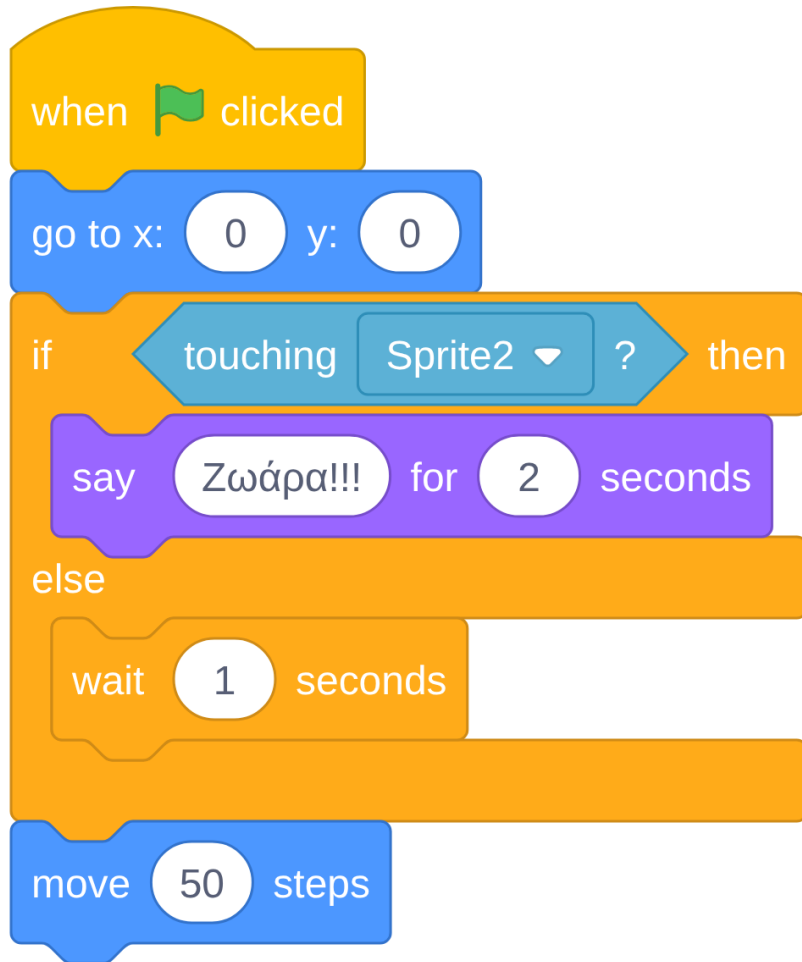
Μοντέλο Εκτέλεσης - Ένα Ακόμα Παράδειγμα...



Εκτελείται το **ένα** τμήμα ή το **άλλο**, αλλά ποτέ και τα δύο μαζί!



Σκεφτείτε πριν τρέξετε! (1/3)



Η Scratch ξεκινάει στο (0, 0) και **δεν** ακουμπάει το μπολ.

Ποιες εντολές θα εκτελεστούν και με ποια σειρά;

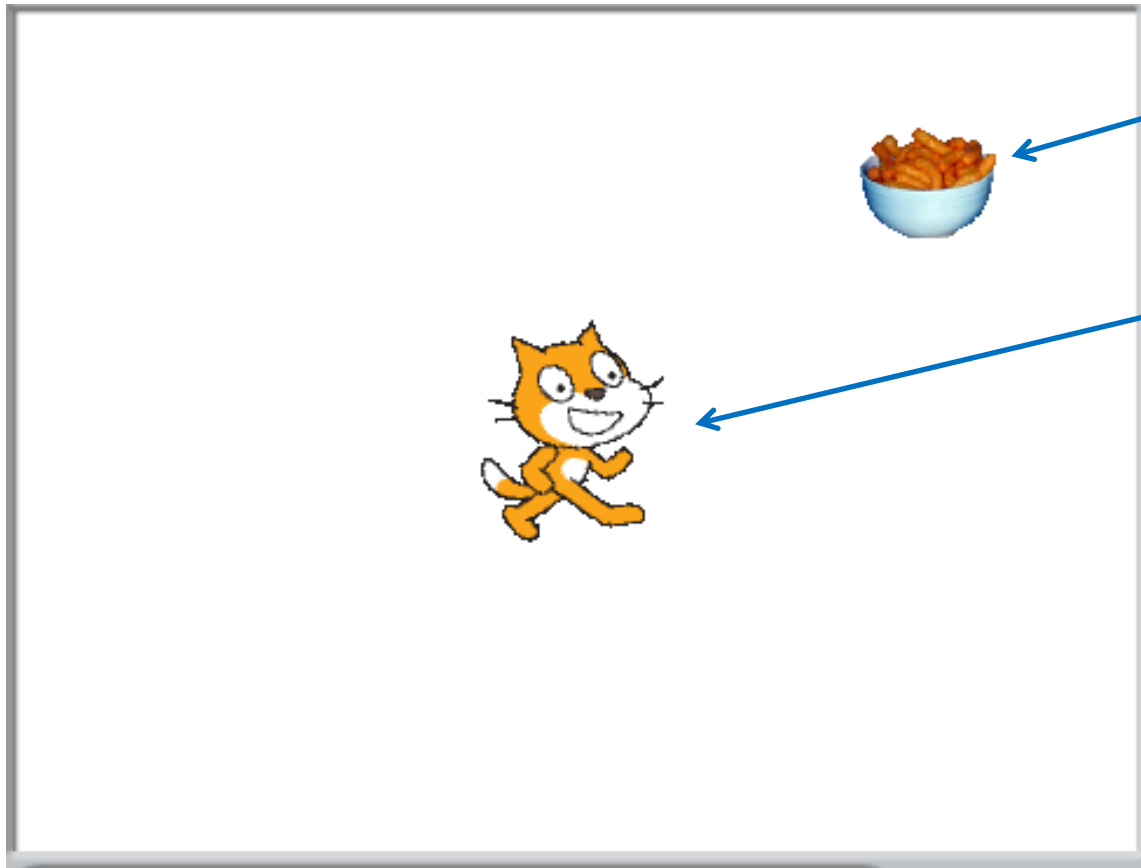
go to x: 0 y: 0 → **wait 1 seconds** → **move 50 steps**.

Το say **δεν** εκτελείται.

Το move εκτελείται **πάντα**, γιατί βρίσκεται μετά το if-else, όχι μέσα του.

Ένα μικρό demo που έχουμε ξαναδεί...

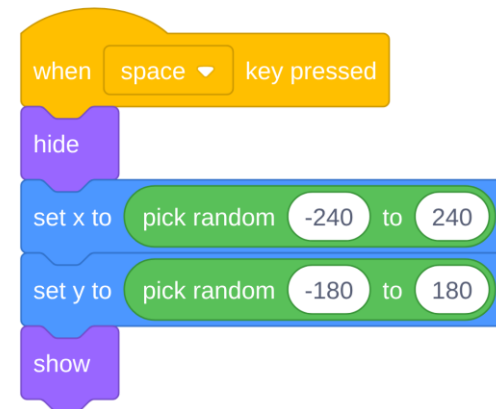
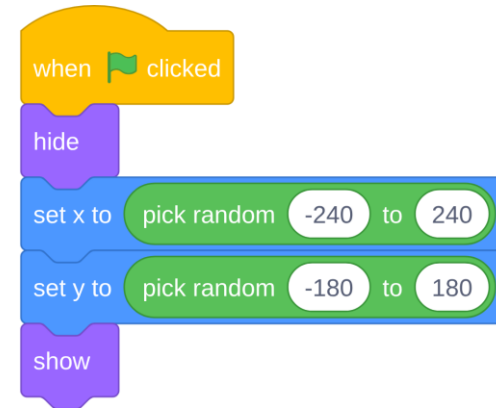
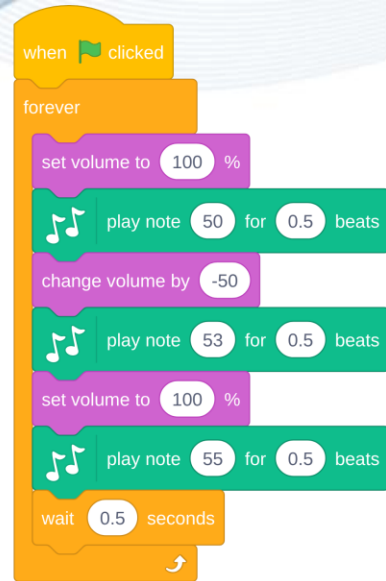
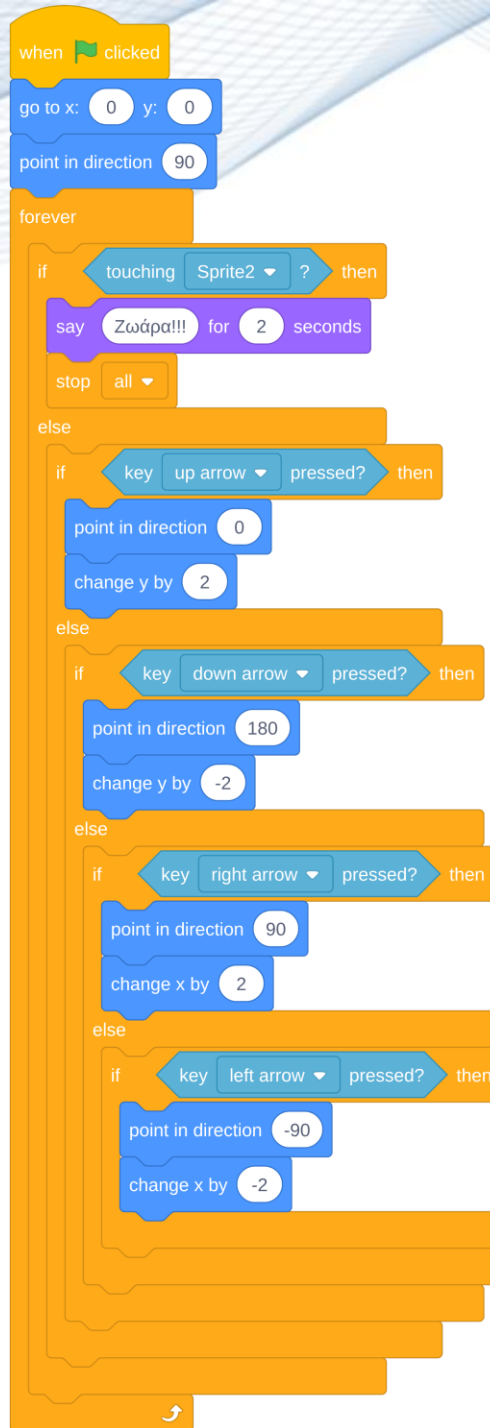
- Νωρίτερα φτιάξαμε ένα πρόγραμμα με το οποίο βοηθήσαμε τη Scratch να βρει φαγητό.

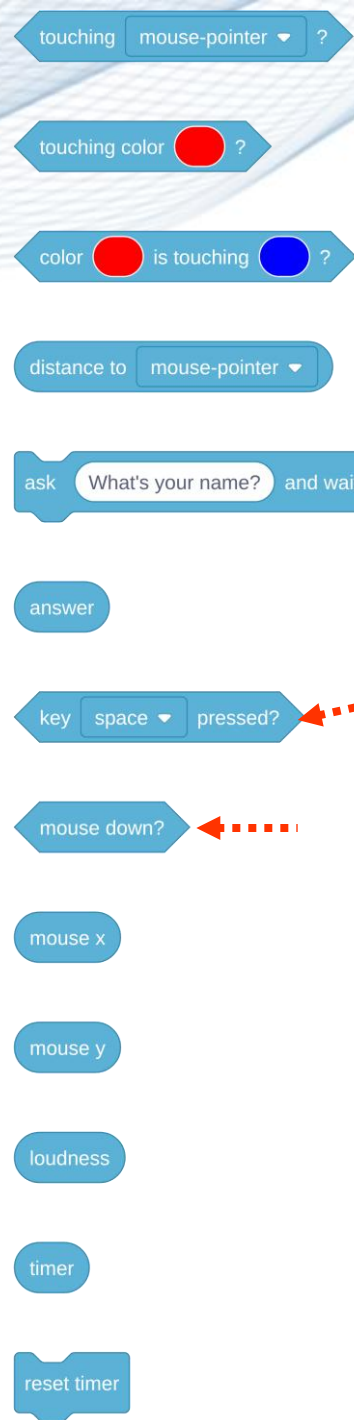


Το μπολ εμφανίζεται σε τυχαία θέση.

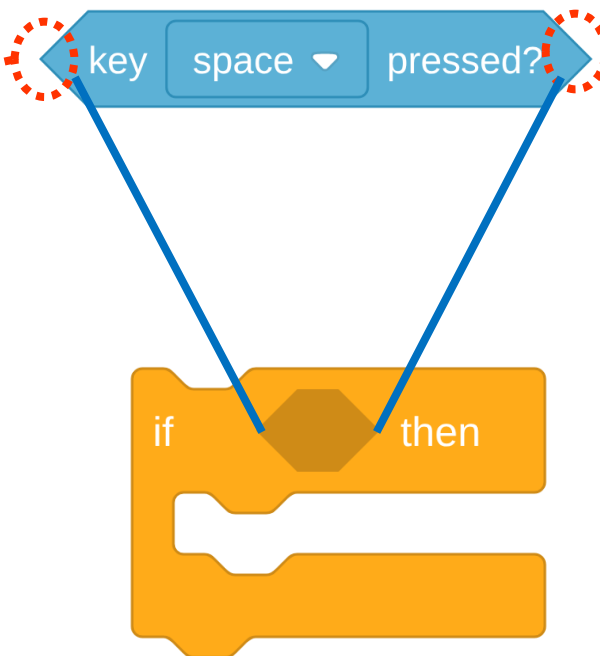
- Η Scratch εμφανίζεται στο σημείο (0,0).
- Χρησιμοποιούμε τα πλήκτρα για να οδηγήσουμε τη Scratch στο φαγητό.
- Το πρόγραμμα τερματίζεται όταν η Scratch ακουμπήσει το φαγητό.

Ένα μικρό demo που έχουμε ξαναδεί...





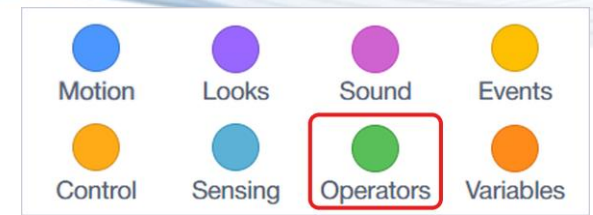
Η κατηγορία “**Sensing**” περιέχει έτοιμα πρότυπα **συνθηκών**. Παρατηρήστε ότι όλες οι συνθήκες οδηγούν σε μία απάντηση ναι ή όχι (true/false).



Παρατηρήστε ότι όσα blocks έχουν **μυτερές άκρες** αφορούν **συνθήκες**.

Δημιουργώντας τις δικές σας συνθήκες...

- Μερικές φορές χρειάζεται να δημιουργήσετε τις δικές σας συνθήκες.
 - Οι σχεδιαστές του Scratch ήταν αρκετά έξυπνοι για να ξέρουν ότι **δε μπορούν να τα προβλέψουν όλα**,
 - έτσι έφτιαξαν ένα **γενικό μηχανισμό** για να δημιουργείτε τις **δικές σας συνθήκες**.



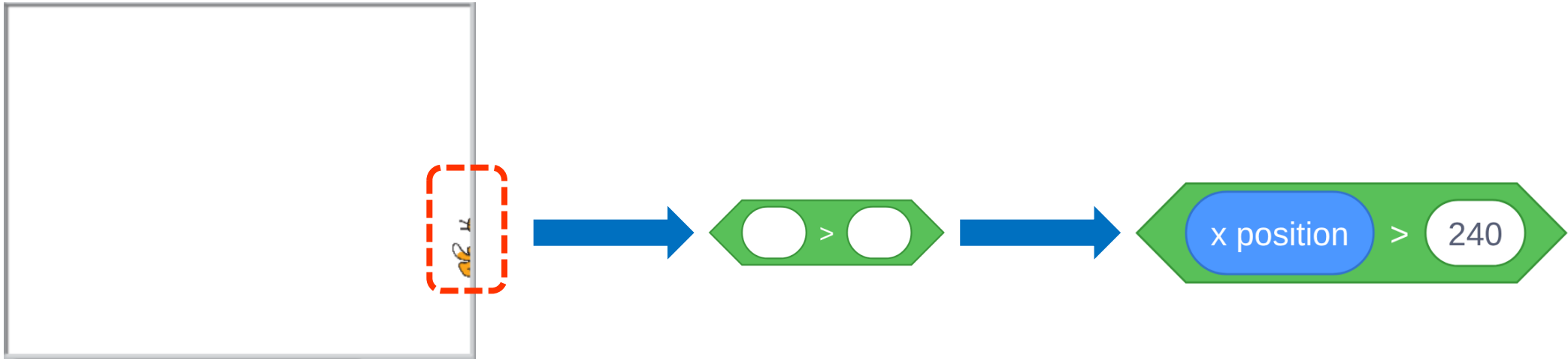
Τελεστές

Σύγκριση

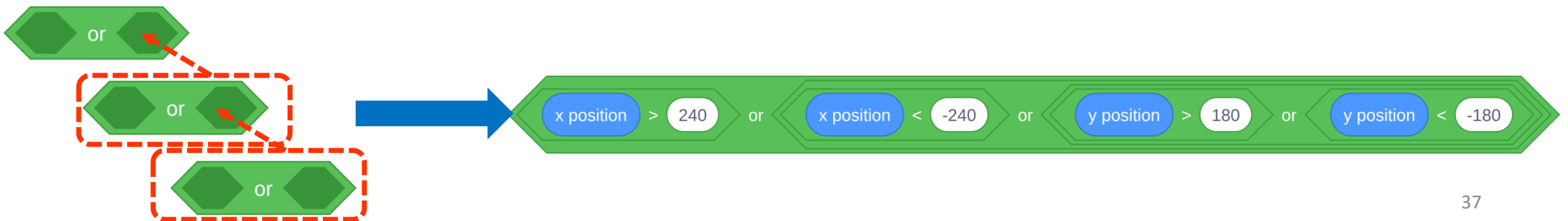
Σύνθεση

Παραδείγματα

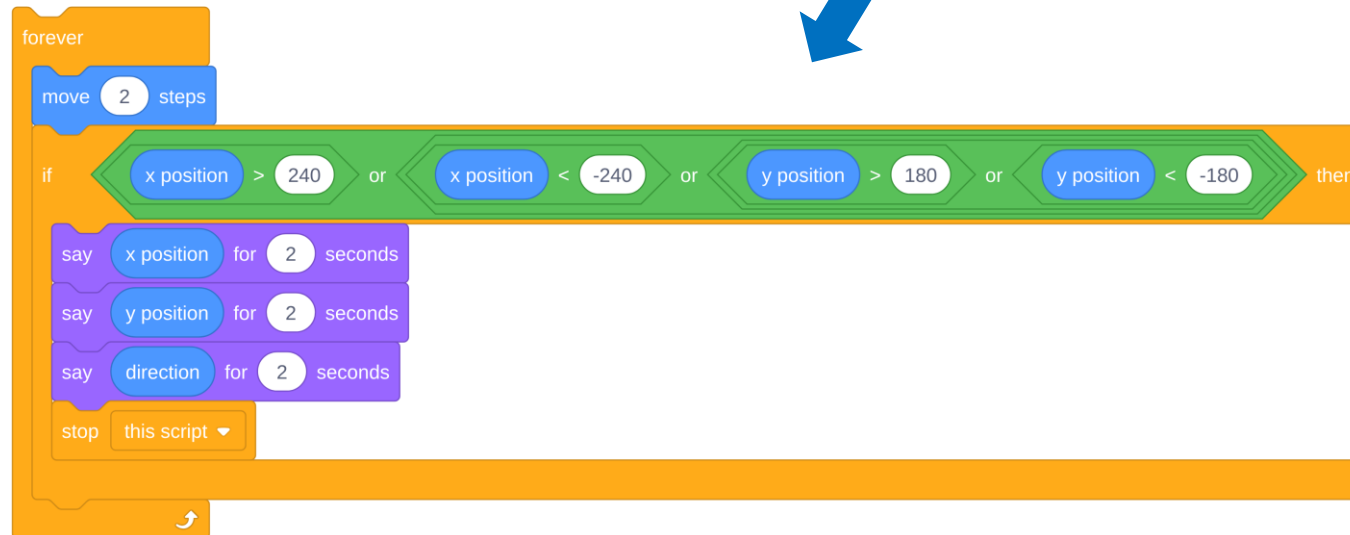
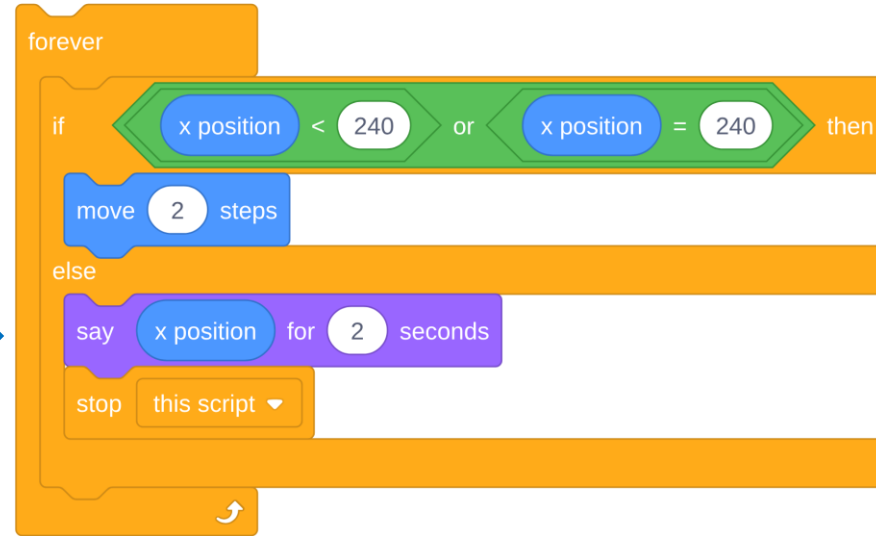
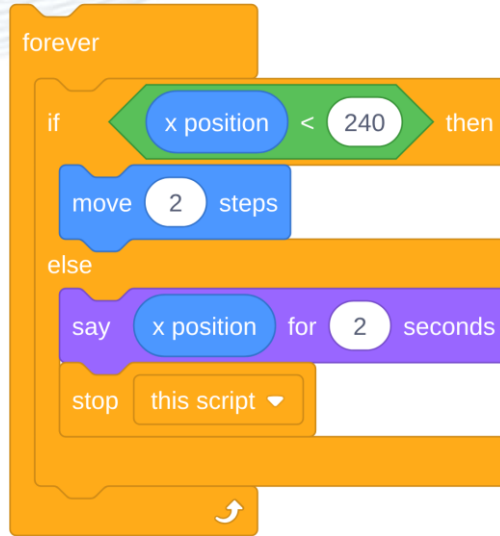
- Είναι η **συντεταγμένη X** του sprite μεγαλύτερη από αυτή της **δεξιάς γωνίας** της σκηνής;



- Είναι η **συντεταγμένες (X, Y)** του sprite εκτός **οποιασδήποτε πλευράς** της σκηνής;
 - Αυτό ισχύει αν $X > 240$ **OR** $X < -240$ **OR** $Y > 180$ **OR** $Y < -180$

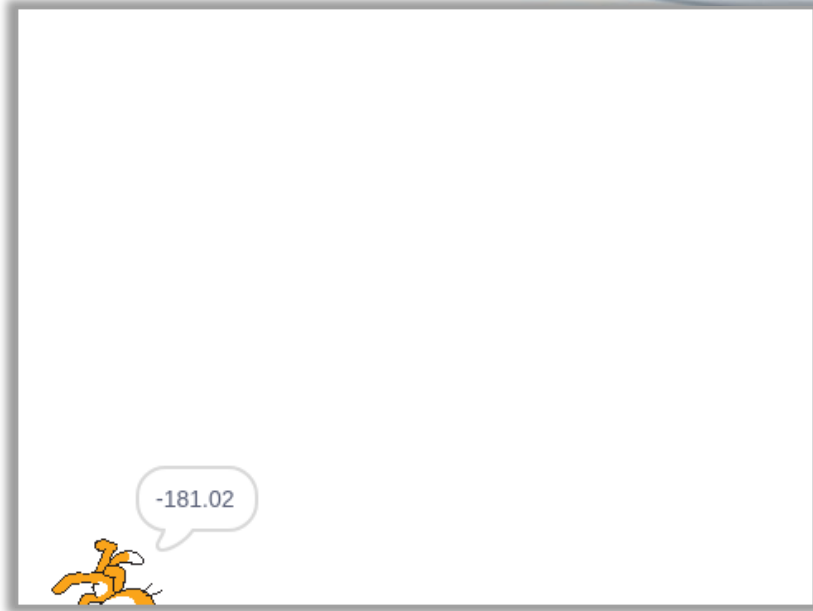
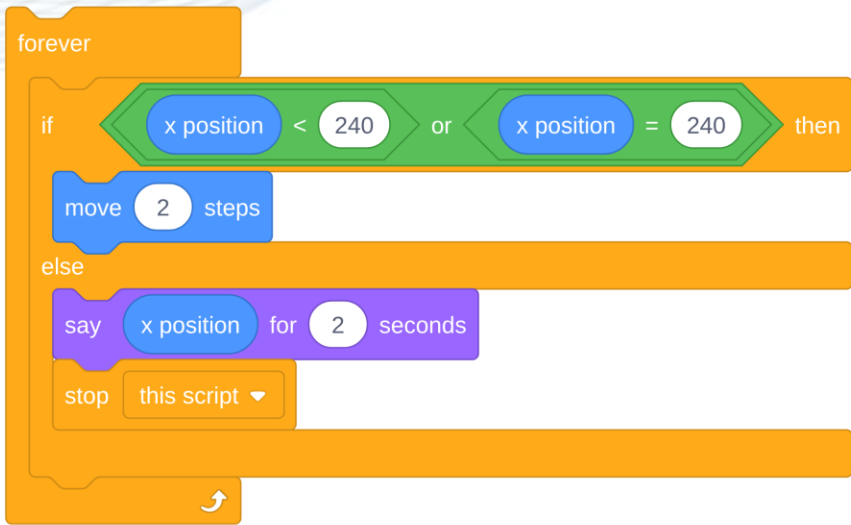


Ένα μικρό demo... στην πράξη (1/2)



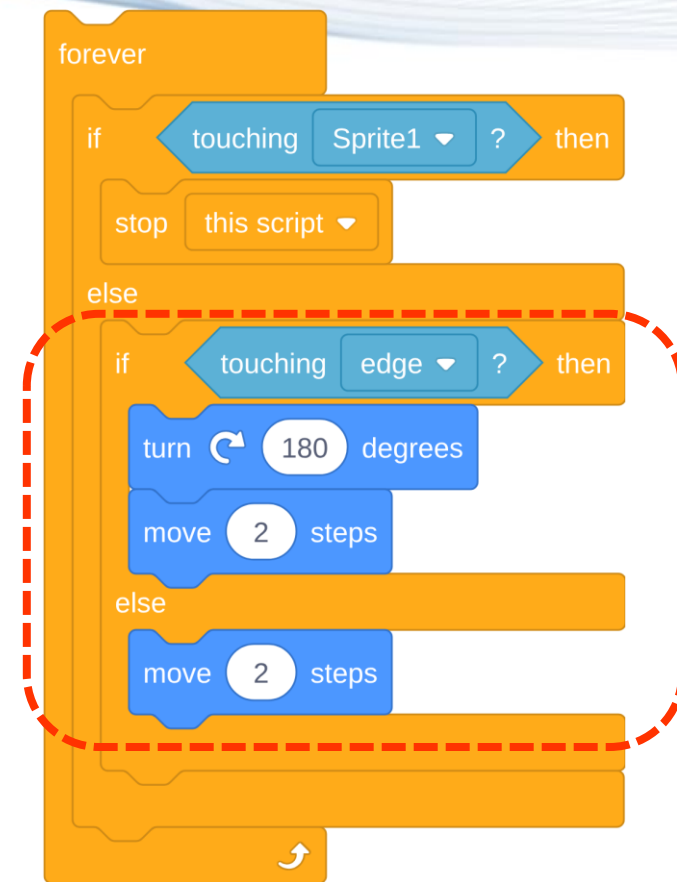
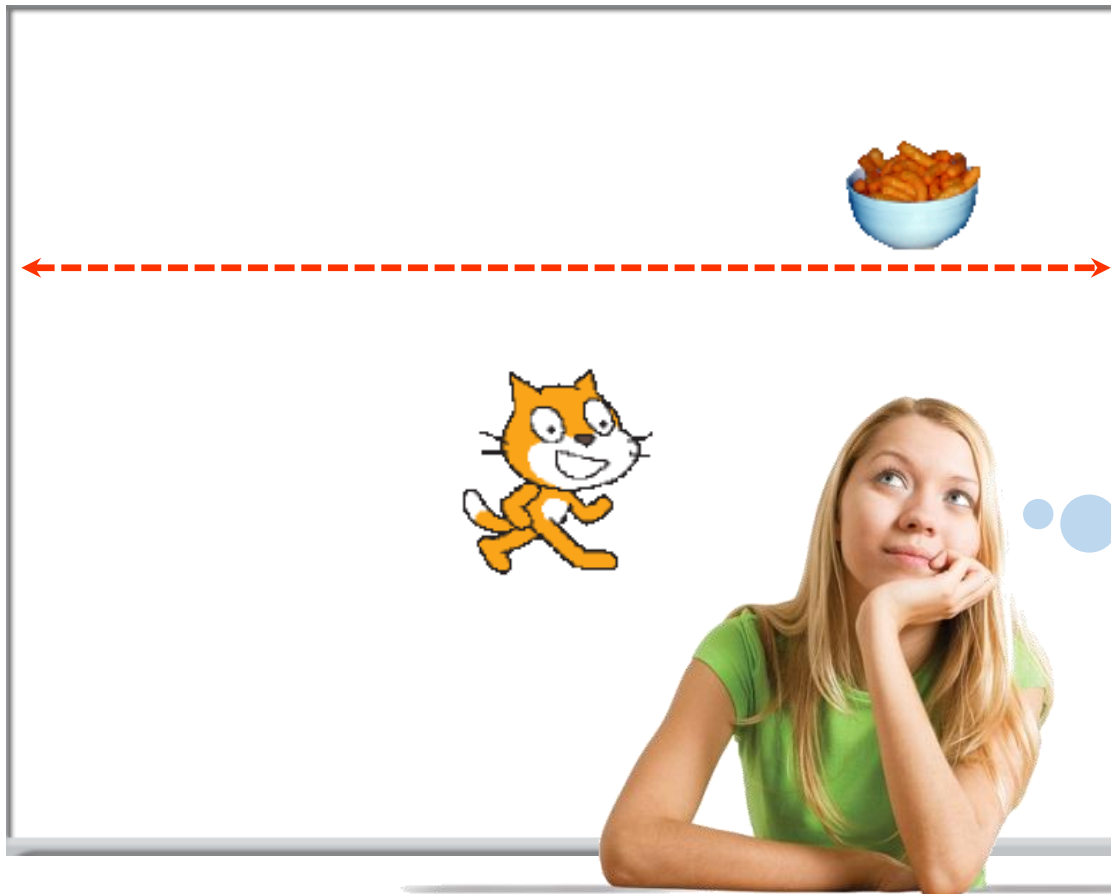
Ένα μικρό demo... στην πράξη (2/2)

go to x: 0 y: 0



Βάζοντάς τα όλα σε μία σειρά!

- Σε μία από τις προηγούμενες ασκήσεις για το σπίτι σας ζητήθηκε να φτιάξετε ένα πρόγραμμα στο οποίο το μπουλ **να κινείται συνεχώς στην κατεύθυνση Χ**.

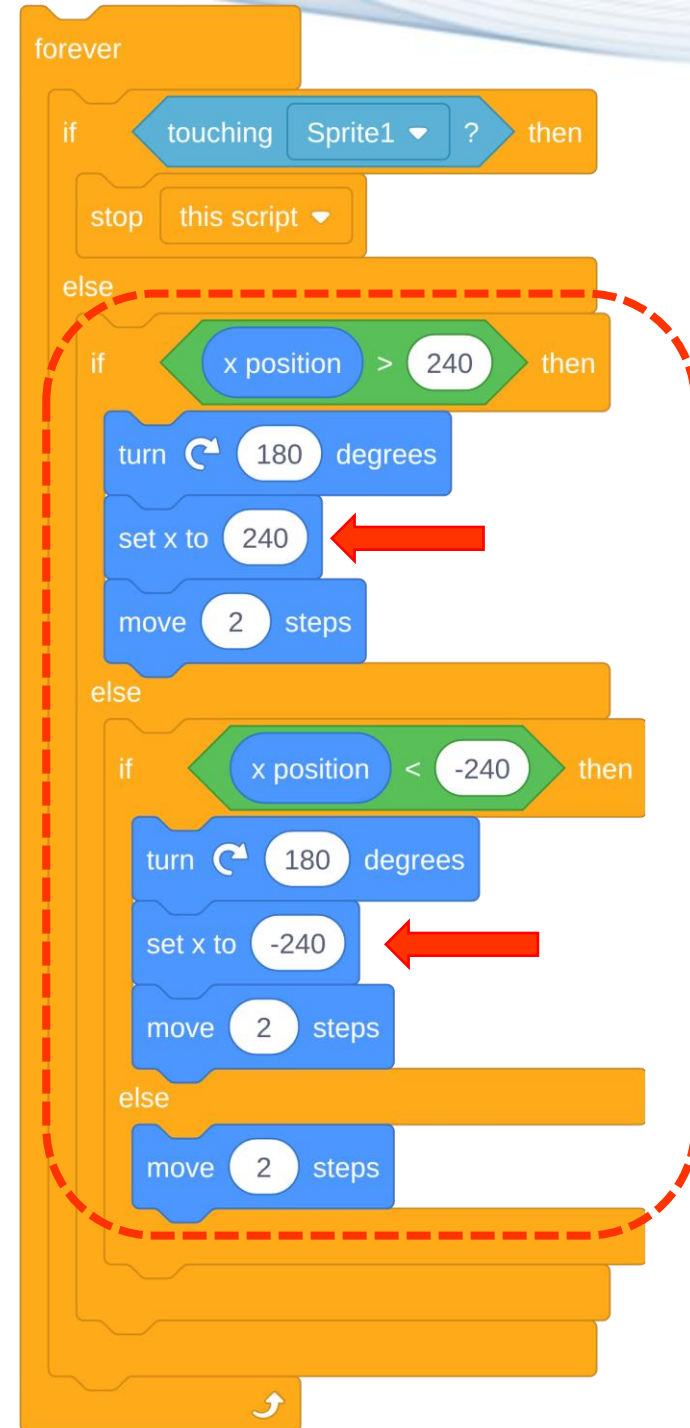
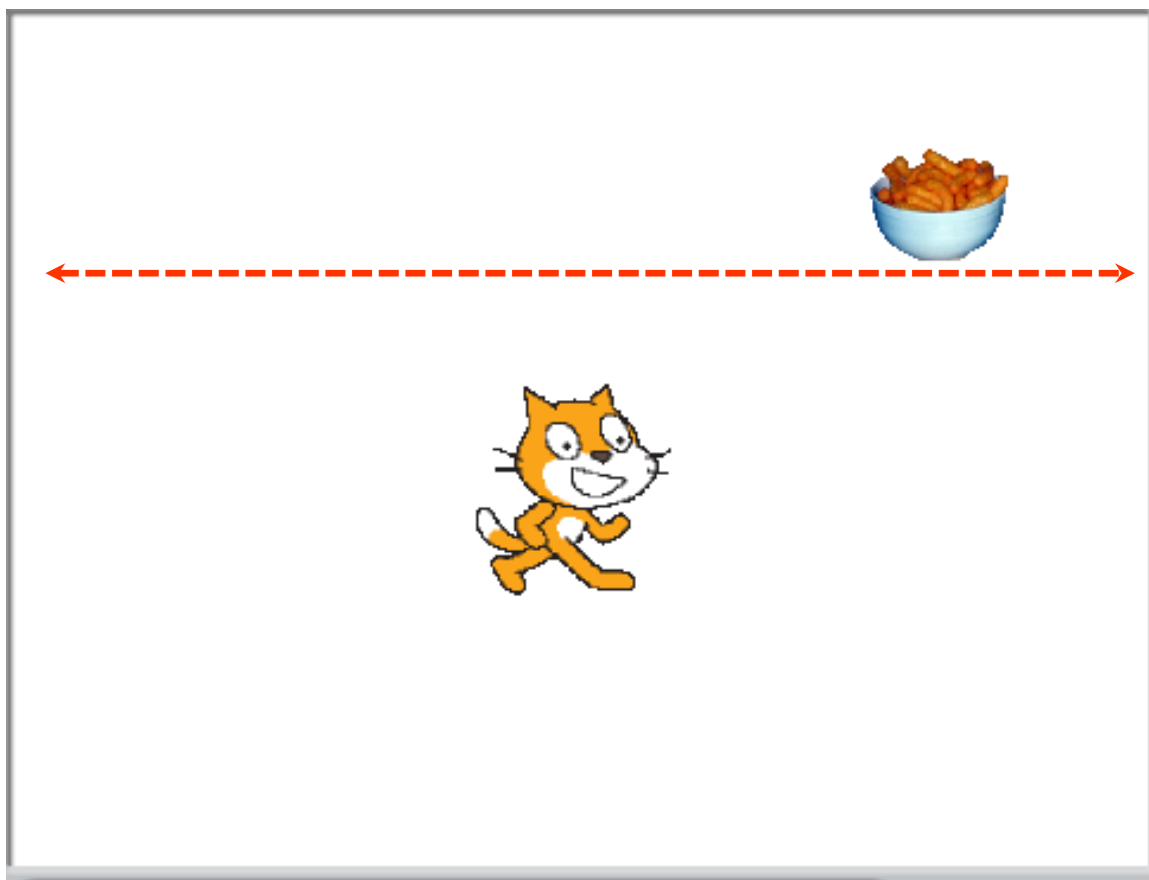


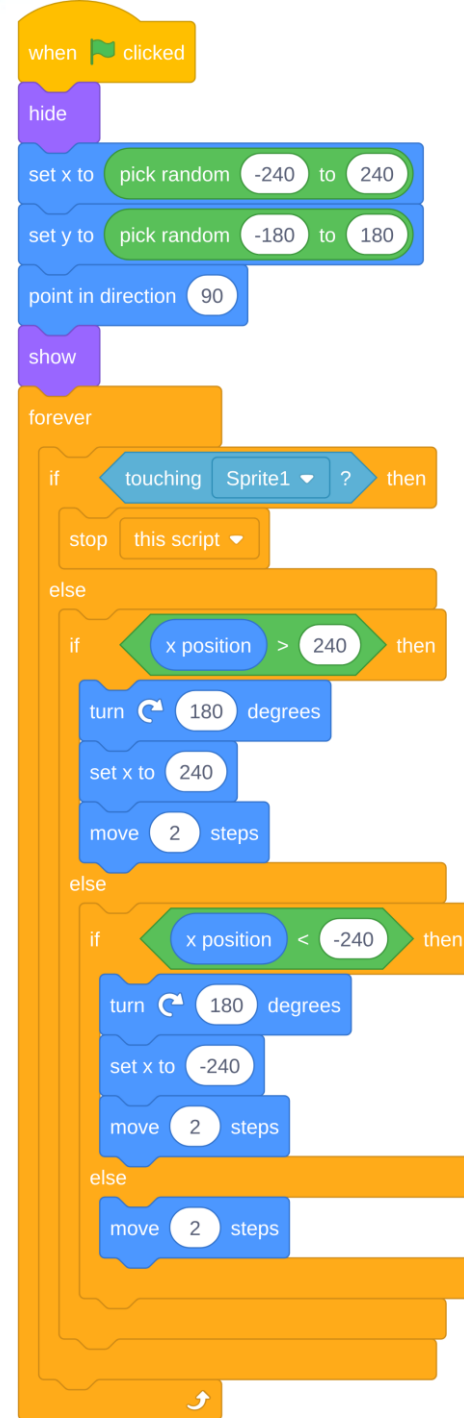
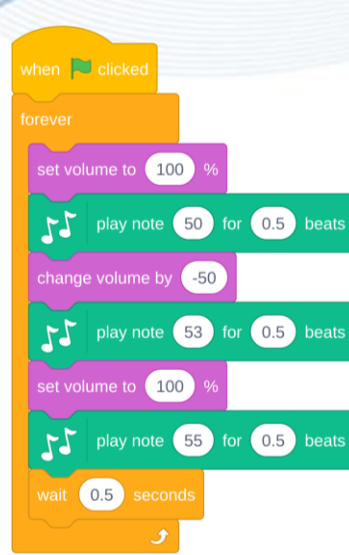
Αυτό θα λειτουργήσει,
αρκεί το μπουλ να μην
ξεκινήσει από τη γωνία.
Τότε θα κολλήσει!

Γιατί;

Η λύση...

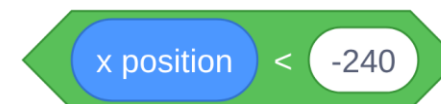
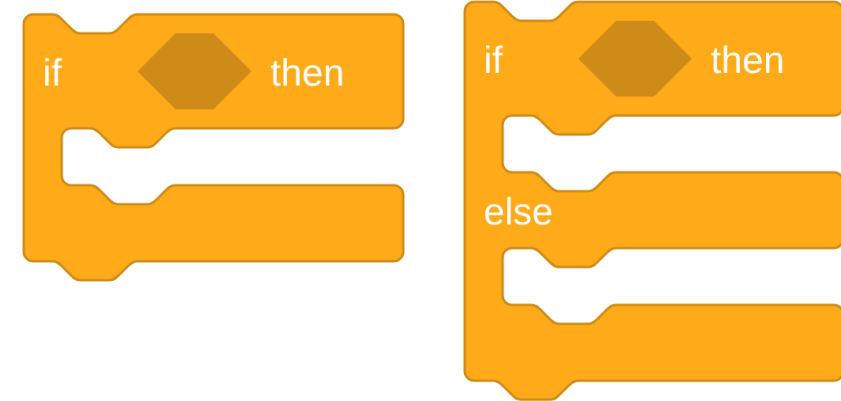
- Μια καλύτερη λύση είναι η ακόλουθη:





Τι είδαμε μέχρι τώρα;

- Η **επιλογή** (**selection**) είναι μία από τις **τρεις** προγραμματιστικές δεξιότητες που θα πρέπει να καλλιεργήσετε:
 - **Αλληλουχία** των εντολών
 - **Επιλογή** ανάμεσα σε διάφορες εναλλακτικές
 - **Επανάληψη**
- Στις περισσότερες γλώσσες προγραμματισμού η επιλογή παίρνει τη μορφή **If-Then**, ή **If-Then-Else** εντολών.
- Τέλος, η επιλογή βασίζεται στην αξιολόγηση **συνθηκών** (**conditions**). Είναι σημαντικό να καταλάβετε ότι οι συνθήκες είναι **εκφράσεις ή δηλώσεις**, των οποίων η αξιολόγηση έχει σαν αποτέλεσμα το **true** ή **false**.



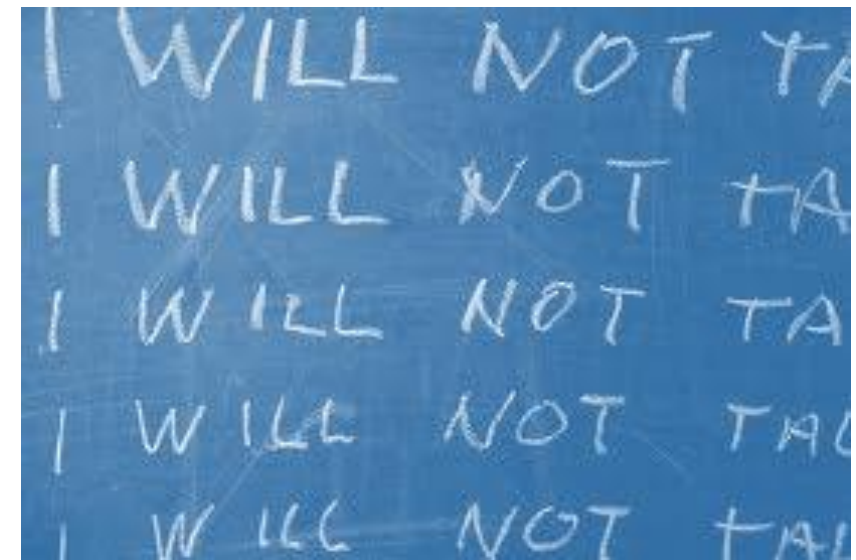
Και λίγη δουλειά για το σπίτι...

- Και μία **άσκηση**:
 - Όπως και στις προηγούμενες ασκήσεις, δημιουργήστε ένα πρόγραμμα το οποίο να περιέχει δύο sprites, τη Scratch και το μπολ.
Όπως και πριν το μπολ θα πρέπει να τοποθετείται σε τυχαίο σημείο στη σκηνή.
Παρόλα αυτά δε θα οδηγείτε εσείς τη Scratch για να βρει το μπολ, αλλά θα πρέπει να το κάνει μόνη της. Όταν το βρει θα πρέπει να φαίνονται η X και Y συντεταγμένες της και το πρόγραμμα να τελειώνει.
- Ακούστε λίγη μουσική μέχρι η Scratch να βρει το φαγητό 😊
Μπορεί να πάρει κάποια ώρα!



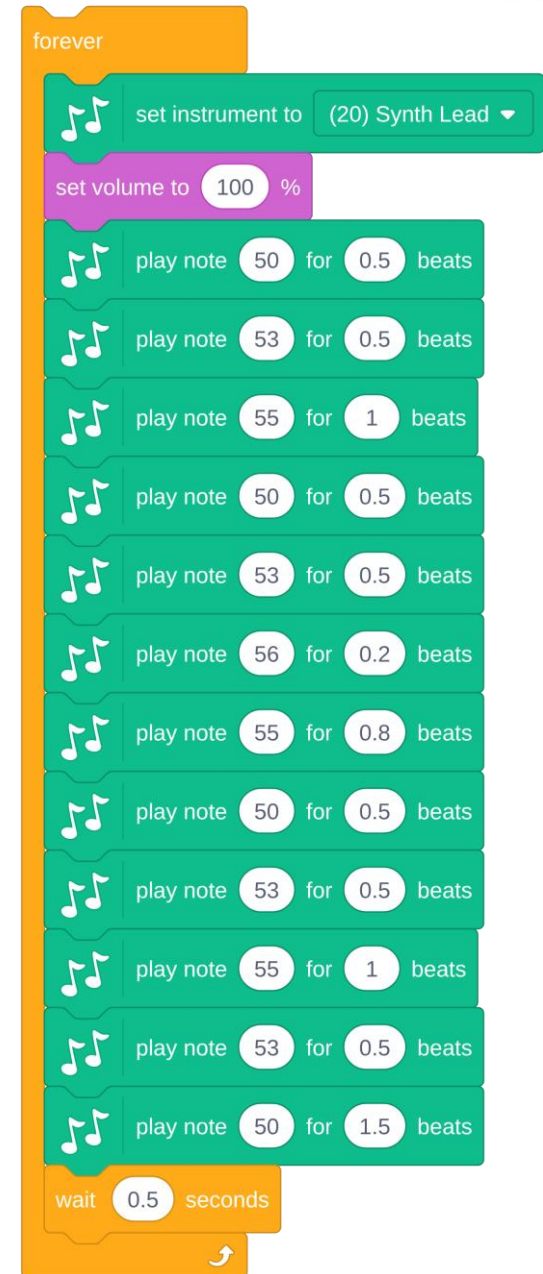
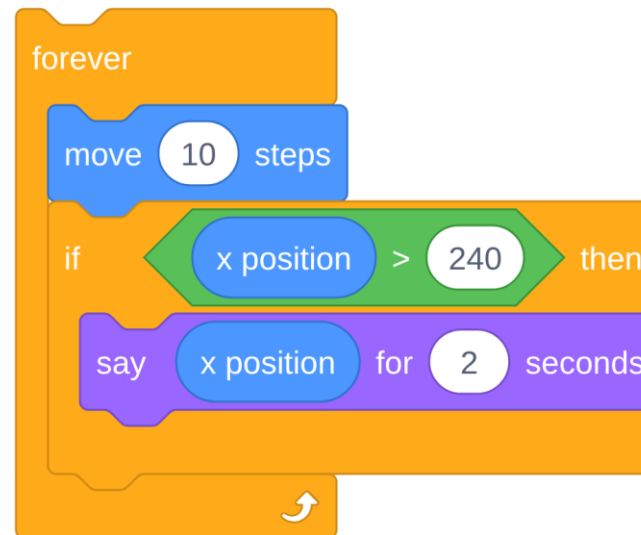
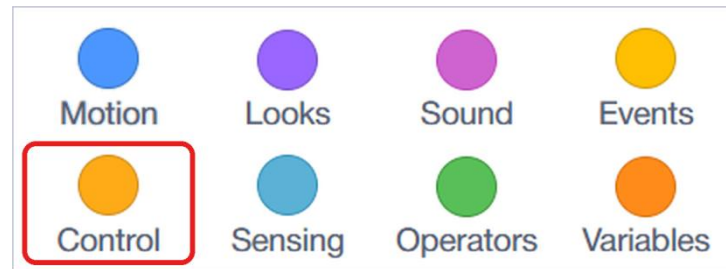
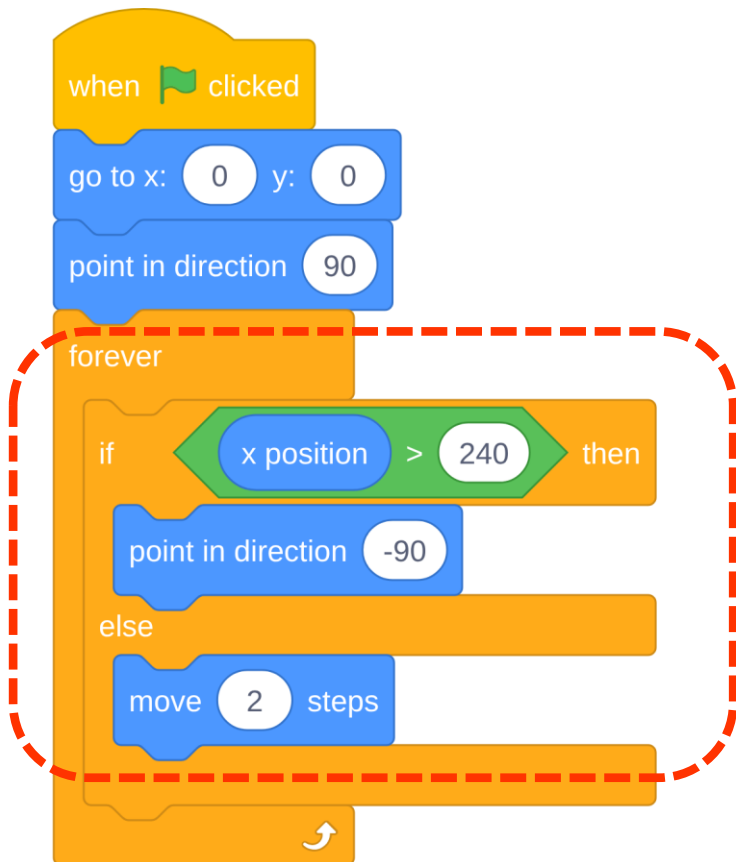
Επανάληψη - Repetition

- Πολλά πράγματα στην πραγματική ζωή επαναλαμβάνονται λόγω της φύσης τους.
 - Συνέχισε να ανακατεύεις **μέχρι (until)** να γίνει ο απαραίτητος αφρός.
 - Τρέξε **μέχρι (until)** να πονέσουν τα πόδια σου.
 - Γράψε **100 φορές** «δε θα ξαναστείλω SMS την ώρα του μαθήματος»



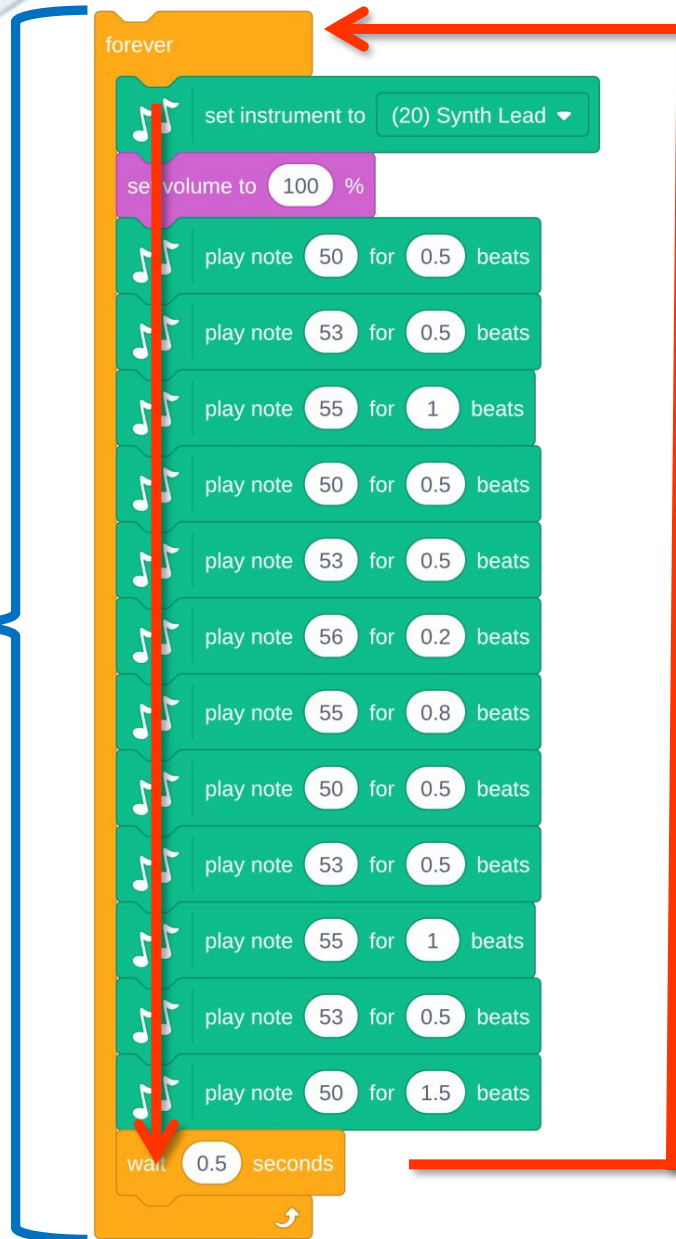
Η Επανάληψη στο Scratch

- Το έχουμε δει ήδη πολλές φορές μέχρι τώρα!



Ορολογία και Μοντέλο Εκτέλεσης

Loop - Βρόχος



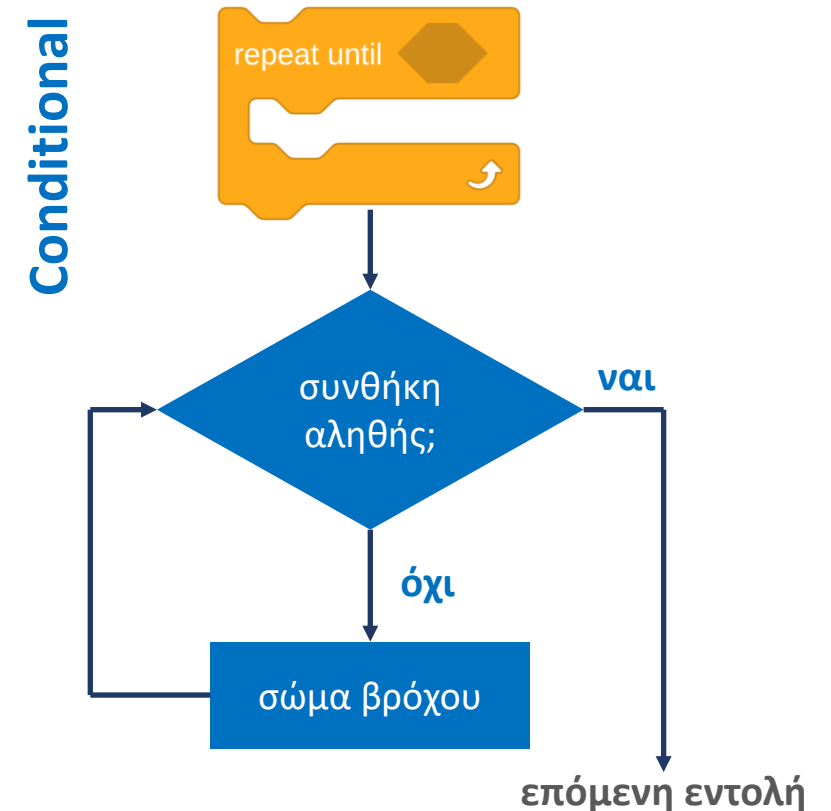
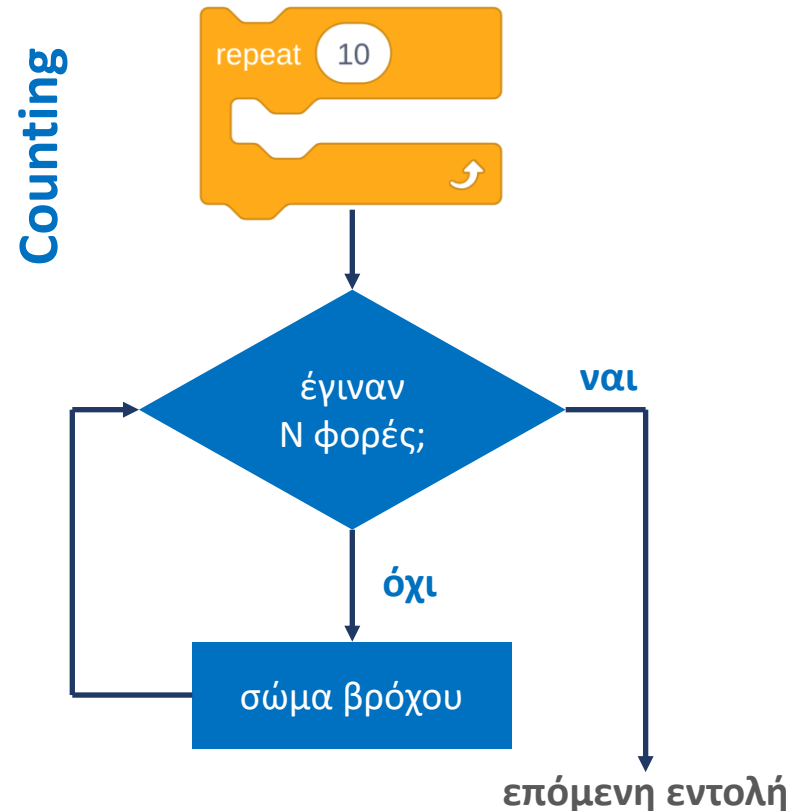
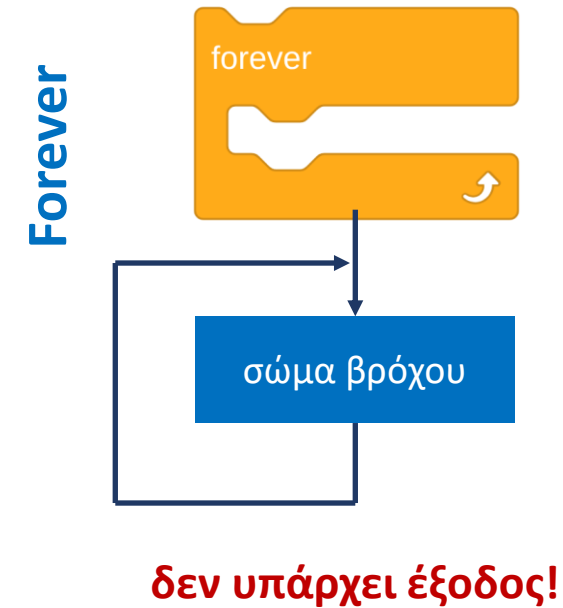
Loop body – Σώμα βρόχου

Το Scratch
«εκτελεί» το
loop body ξανά
και ξανά...



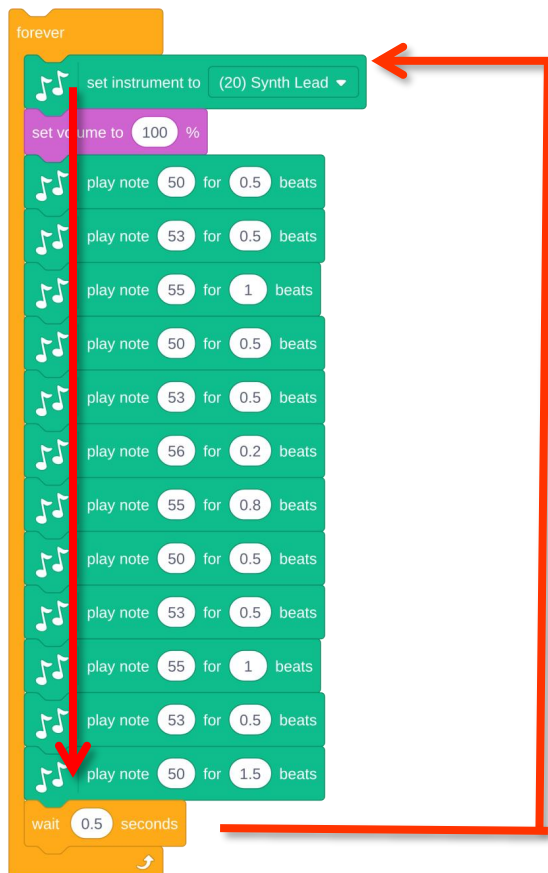
Γενικές Αρχές

- Υπάρχουν τρεις τύποι επανάληψης (ή **looping**):
 - Forever** – Για πάντα (ατέρμονο)
 - Counting** – Με μέτρηση
 - Conditional** – Υπό συνθήκες



Forever loop

- Ένα **forever loop** επαναλαμβάνει το σώμα του βρόχου **για πάντα**.



Πώς μπορεί το loop **να σταματήσει;**

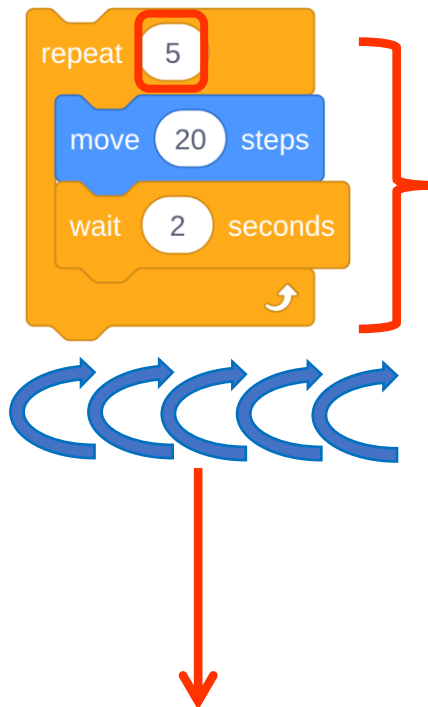
Δε μπορεί, εκτός

- εάν το σταματήσει ο χρήστης,
- ή υπάρχει ένα σταμάτημα εντός του loop,
 - ή κλείσει ο υπολογιστής



Counting loop

- Ένα **counting loop** επαναλαμβάνει το σώμα του βρόχου **για συγκεκριμένες επαναλήψεις**.
 - 1, 10 ή 88 φορές...



Επανάληψη του σώματος **5 ακριβώς φορές**.

Ένα μικρό demo...

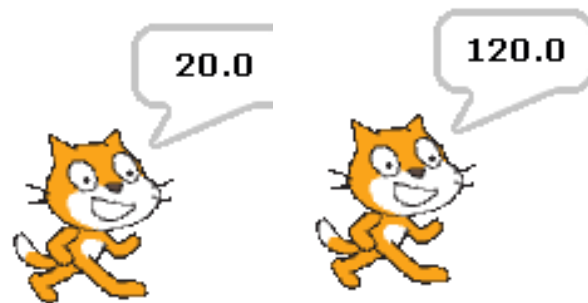


- Counting loops...
 - Μετακίνηση ακριβώς 100 steps
 - Στροφή 90 μοιρών
 - “Flash” π.χ. γρήγορο μεγάλωμα και συρρίκνωση 3 φορές

Ένα μικρό demo... στην πράξη

Demos_2026-2027/lecture1_06.sb3

```
say x position for 2 seconds
repeat 5
  move 20 steps
  wait 0.5 seconds
say x position for 2 seconds
```



```
say direction for 2 seconds
repeat 90
  turn 1 degrees
say direction for 2 seconds
```

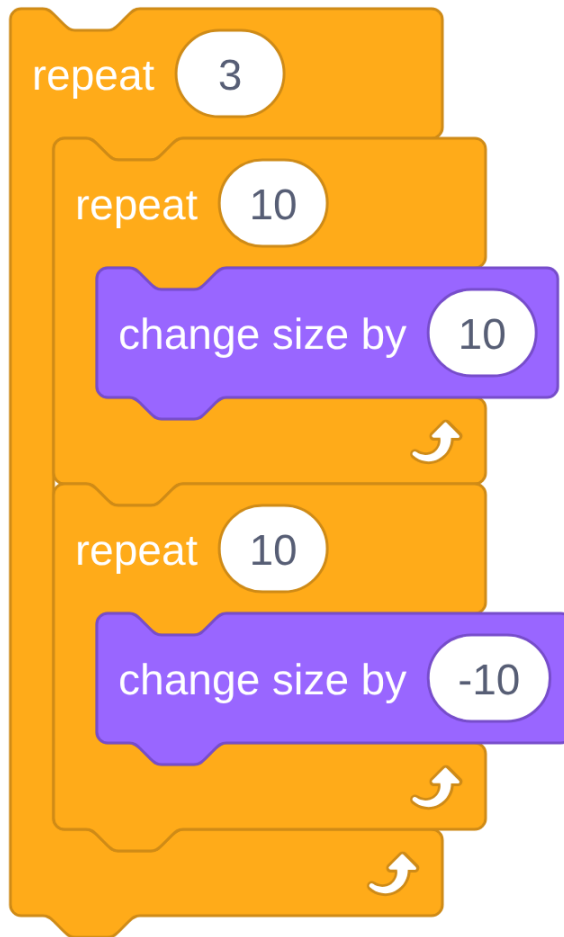


```
repeat 3
  repeat 10
    change size by 10
  repeat 10
    change size by -10
```



Doubly nested loop

Σκεφτείτε πριν τρέξετε! (2/3)



Βρόχος μέσα σε βρόχο (nested loop): ο εσωτερικός βρόχος εκτελείται **ολόκληρος** σε κάθε επανάληψη του εξωτερικού.

Πόσες φορές θα εκτελεστεί η εντολή change size by (10);
Πόσες συνολικά οι εντολές change size;

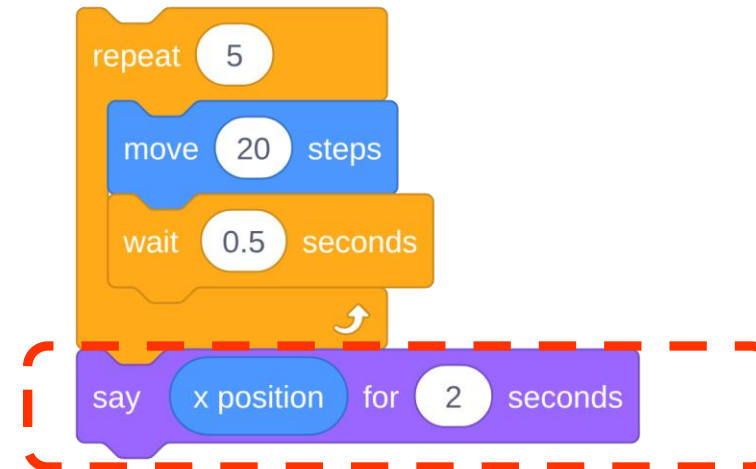
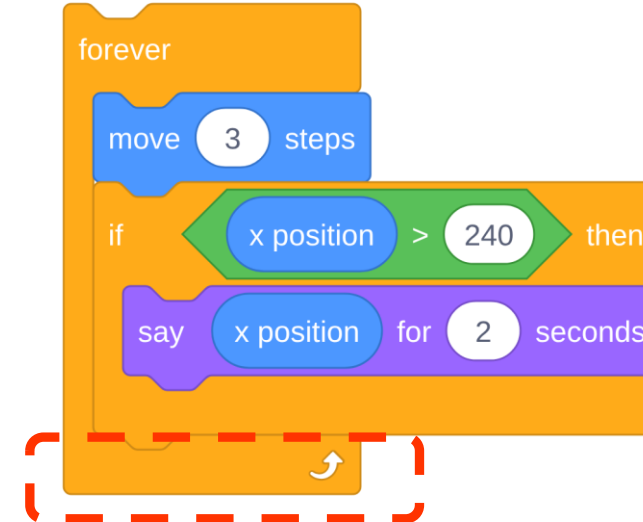
$3 \times 10 = 30$ φορές το change size by (10).

$3 \times (10 + 10) = 60$ εντολές change size συνολικά.

Στο τέλος το sprite έχει πάλι το αρχικό του μέγεθος.

Παρατήρηση!

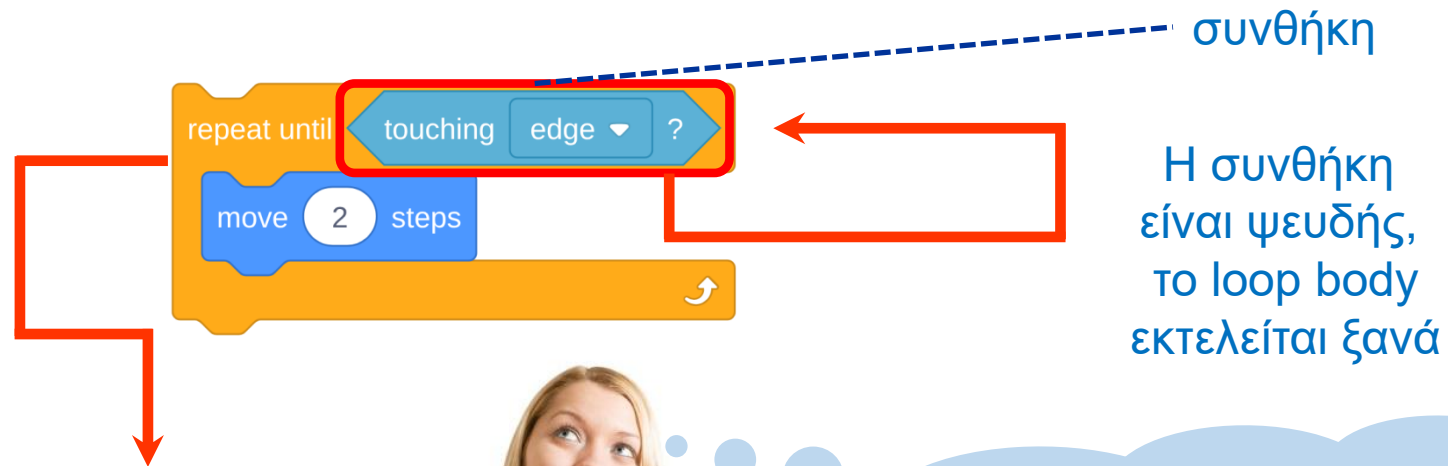
- Παρατηρήστε ότι ένα **forever** **δεν μπορεί** να ακολουθηθεί από άλλη εντολή,
- ενώ ένα **repeat** επιτρέπει επόμενες εντολές.



Conditional Loop

- Ένα **conditional loop** επαναλαμβάνει το σώμα του βρόχου με βάση μία **συνθήκη**.
 - Αν η συνθήκη είναι **αληθής-true** ο βρόχος σταματάει.
 - Αν η συνθήκη είναι **ψευδής-false** ο βρόχος επαναλαμβάνεται.

Η συνθήκη
είναι αληθής,
σταματάει το
loop



Η συνθήκη
είναι ψευδής,
το loop body
εκτελείται ξανά

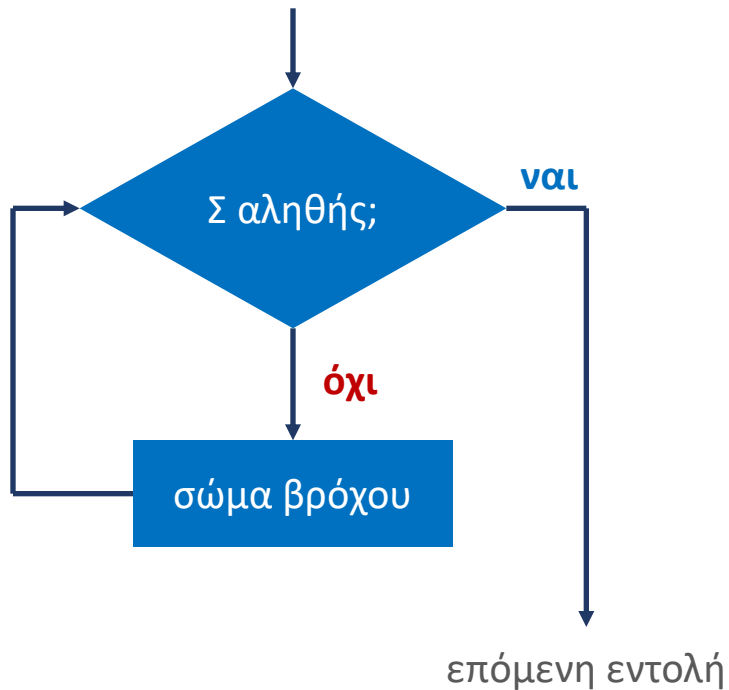


Ένα Conditional Loop
επαναλαμβάνεται **0** ή
περισσότερες φορές.

Προσοχή! repeat until $\neq$ while

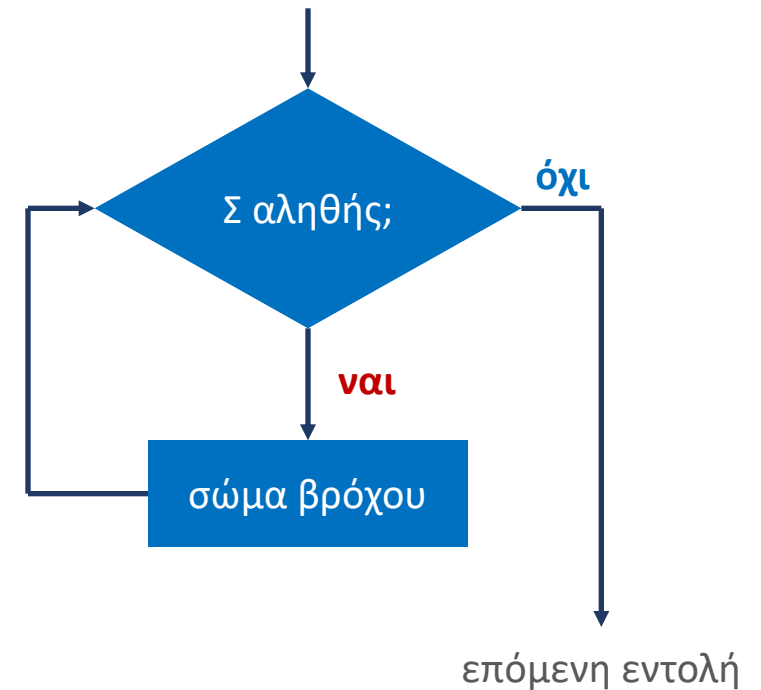
Scratch: repeat until $\langle \Sigma \rangle$

επαναλαμβάνει **ΜΕΧΡΙ** η Σ να γίνει αληθής



JavaScript, Python, ...: while Σ

επαναλαμβάνει **ΟΣΟ** η Σ είναι αληθής



repeat until $\langle \Sigma \rangle \equiv \text{while } (\text{not } \Sigma)$

`repeat until <touching edge?>  $\equiv$  while (!touchingEdge()) {...}  $\equiv$  while not touching_edge():`

...

Ένα μικρό demo...

- Conditional Loops...

- Κινήσου μέχρι να φτάσεις στην άκρη της σκηνής.
- Στρίψε μέχρι το sprite να κοιτάζει προς τα πάνω (direction = 0).
- Ανίχνευσε το ποντίκι και ζωγράφισε στη σκηνή όσο το κουμπί του ποντικιού είναι πατημένο...

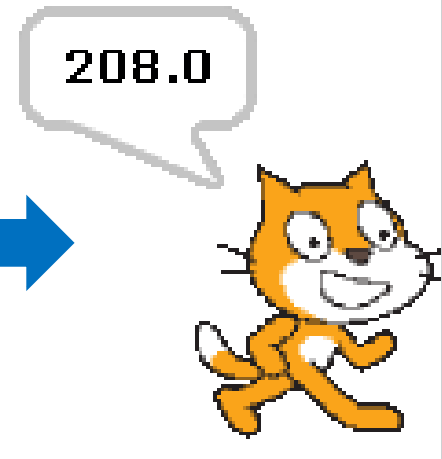
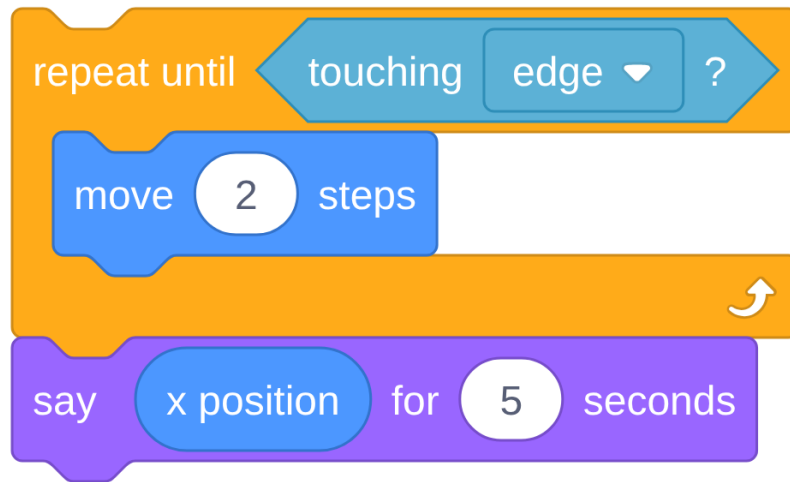


Ένα μικρό demo... στην πράξη (1/3)

DEMO

- Conditional Loops...
 - Κινήσου μέχρι να φτάσεις στην άκρη της σκηνής.

Demos_2026-2027/lecture1_07.sb3



Σκεφτείτε πριν τρέξετε! (3/3)



Η Scratch ξεκινάει **ήδη ακουμπώντας την άκρη** ($x = 240$).
Πόσες φορές θα εκτελεστεί το σώμα του βρόχου; Τι θα πει η Scratch;

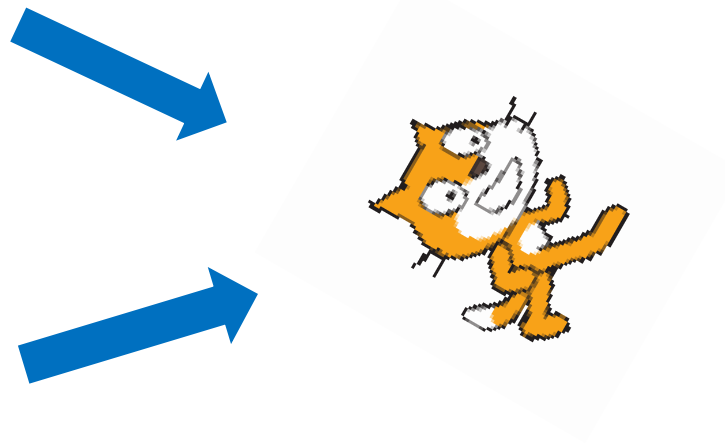
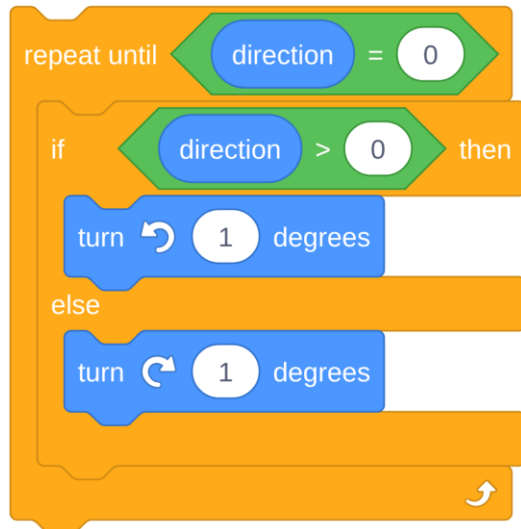
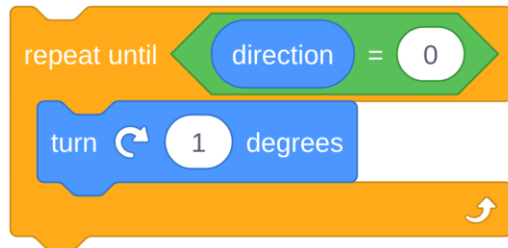
Καμία! Η συνθήκη ελέγχεται **πριν** από κάθε επανάληψη, οπότε ο βρόχος τελειώνει αμέσως: η Scratch λέει «Ξεκινάω!» και μετά «Τέλος!». Γι' αυτό ένα conditional loop εκτελείται **0 ή περισσότερες φορές**.

Ένα μικρό demo... στην πράξη (2/3)

DEMO

- Conditional Loops...

- Στρίψε μέχρι το sprite να κοιτάζει προς τα πάνω (direction = 0).



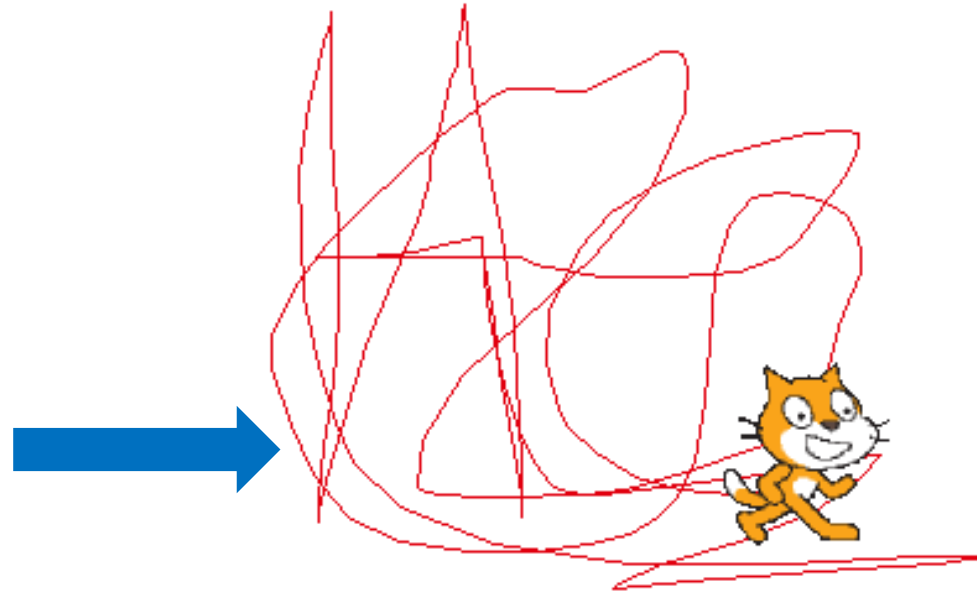
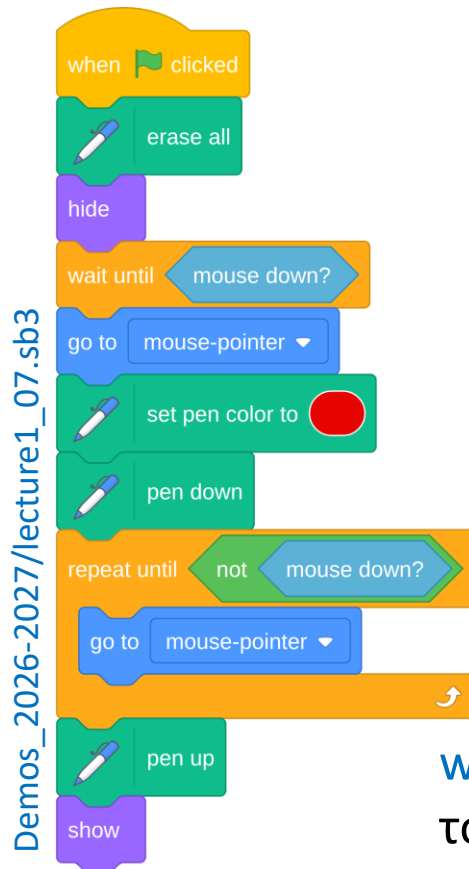
Demos_2026-2027/lecture1_07.sb3

Ένα μικρό demo... στην πράξη (3/3)

DEMO

- Conditional Loops...

- Ανίχνευσε το ποντίκι και ζωγράφισε στη σκηνή όσο το κουμπί του ποντικιού είναι πατημένο...

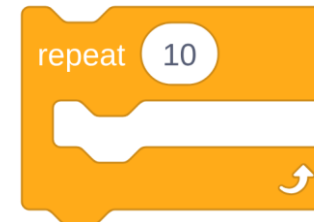


wait until <συνθήκη>:

το script περιμένει (δεν κάνει τίποτα) μέχρι η συνθήκη να γίνει αληθής. 61

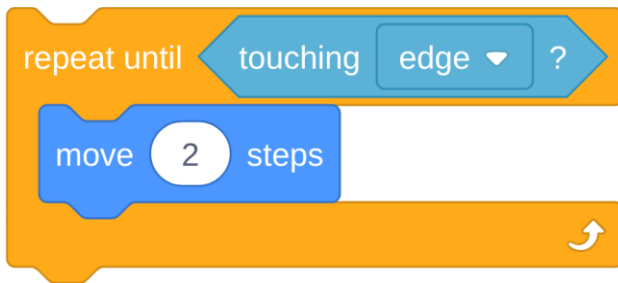
Τι είδαμε μέχρι τώρα;

- Η επανάληψη (**repetition**) είναι η τρίτη από τις προγραμματιστικές δεξιότητες που θα πρέπει να καλλιεργήσετε:
 - Αλληλουχία των εντολών
 - Επιλογή ανάμεσα σε διάφορες εναλλακτικές
 - Επανάληψη
- Στις περισσότερες προγραμματιστικές γλώσσες η επανάληψη έχει μία λίγο διαφορετική μορφή:
 - Τα counting loops ονομάζονται “for” loops – για (for) έναν αριθμό επαναλήψεων...
 - Τα conditional loops συχνά ονομάζονται “while” loops – όσο (while) η συνθήκη είναι αληθής (true) – το αντίθετο από το repeat until!



Η ίδια αρχή σε τρεις γλώσσες

Scratch



JavaScript

```
for (let i = 0; i < 10; i++) {  
  move(10);  
}
```

```
while (!touchingEdge()) {  
  move(2);  
}
```

```
if (touching("Sprite2")) {  
  say("Zwάpa!!!");  
} else {  
  move(10);  
}
```

Python

```
for i in range(10):  
    move(10)
```

```
while not touching_edge():  
    move(2)
```

```
if touching("Sprite2"):  
    say("Zwάpa!!!")  
else:  
    move(10)
```

Άλλη σύνταξη, ίδια λογική! (Οι move, say, touching... είναι υποθετικές συναρτήσεις, μόνο για τη σύγκριση.)

Και λίγη δουλειά για το σπίτι...

- Και μία άσκηση:
 - Στην Αγγλία, όταν βρέχει καταρρακτωδώς, χρησιμοποιείται η ακόλουθη έκφραση για να το περιγράψει: “**it’s raining cats and dogs**”.
Φτιάξτε ένα πρόγραμμα με 10 ή περισσότερα sprites (μία μείξη από γάτες και σκύλους) που πηγαίνουν από την κορυφή προς το κάτω μέρος της σκηνής, συνέχεια...
 - Το αποτέλεσμα θα πρέπει να φαίνεται όπως στην παρακάτω εικόνα 😊



Remember!
This is only the start!!!



forever
imagine
program
share

