ΕΛΛΗΝΙΚΗ ΔΗΜΟΚΡΑΤΙΑ
Εθνικόν και Καποδιστριακόν
Πανεπιστήμιον Αθηνών

# Διάλεξη «Εισαγωγή στην πλατφόρμα Android»

Εισηγητής: Παντελής Μπαλαούρας

2021 - 2022

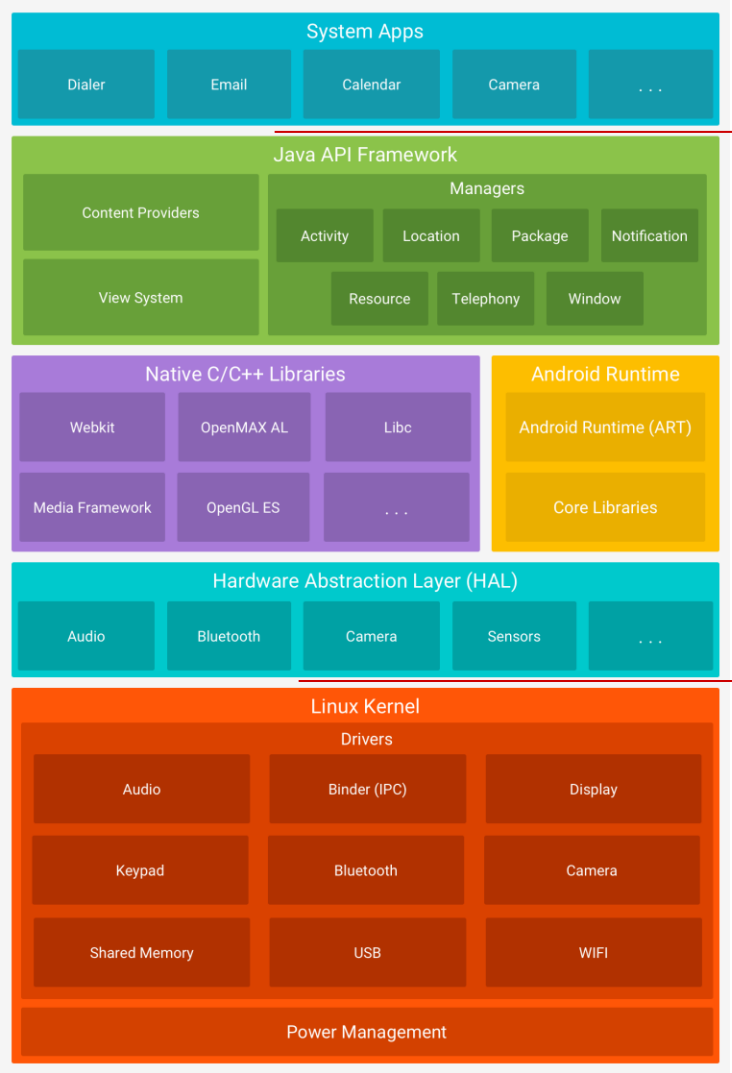
# ΑΡΧΙΤΕΚΤΟΝΙΚΗ ΚΑΙ ΔΟΜΙΚΑ ΣΥΣΤΑΤΙΚΑ ANDROID

# Τι είναι το Android

Ένα **Λειτουργικό Σύστημα**,

μια **στοίβα Λογισμικού** για φορητές συσκευές με, σχετικά

- χαμηλή επεξεργαστική ισχύ
- χαμηλή μνήμη
- οθόνη αφής ως μέσο διεπαφής
- ένας χρήστης




*περιλαμβάνει*

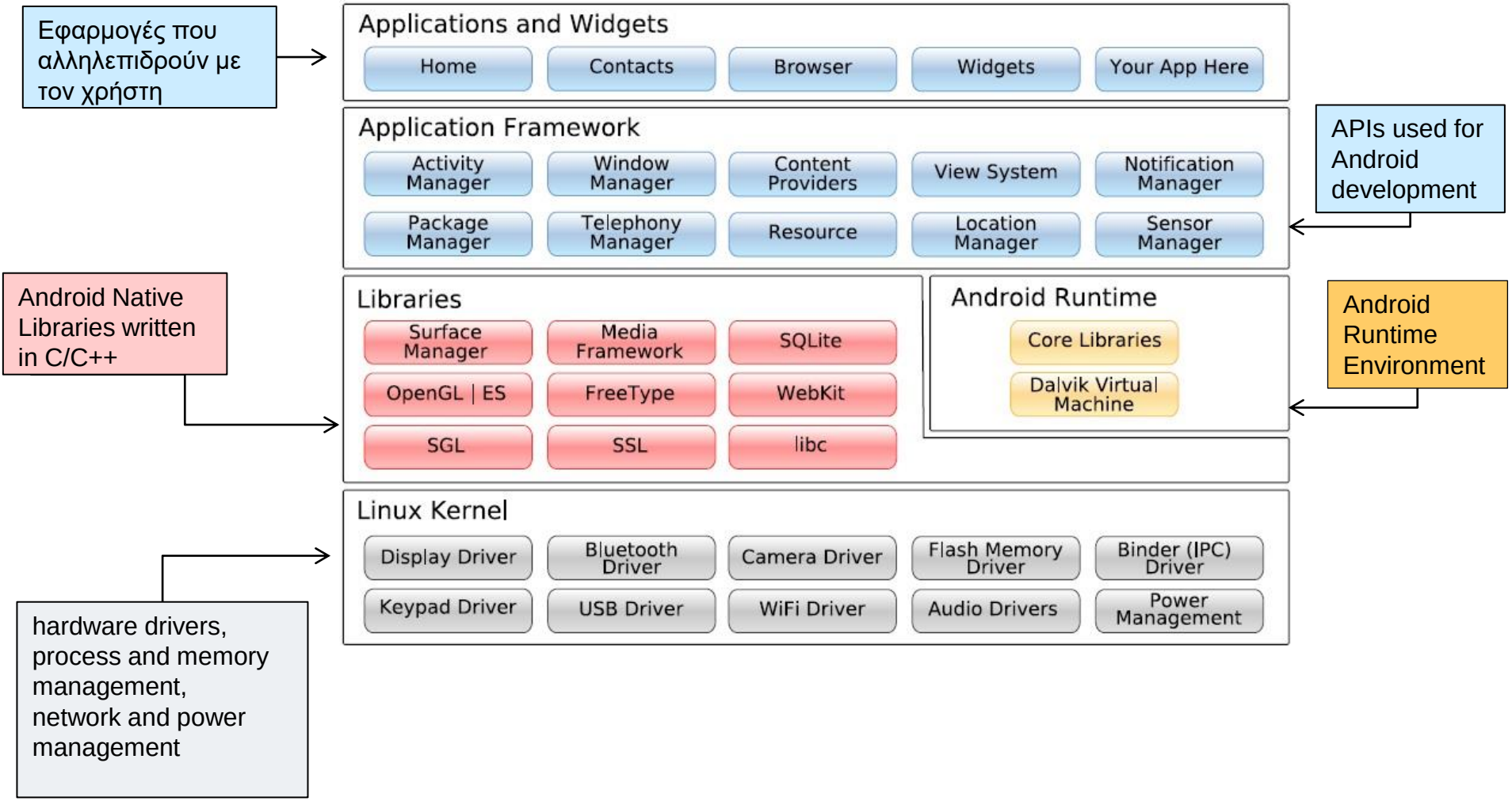
Βασικές εφαρμογές

Μεσισμικό (middleware)

Λειτουργικό σύστημα

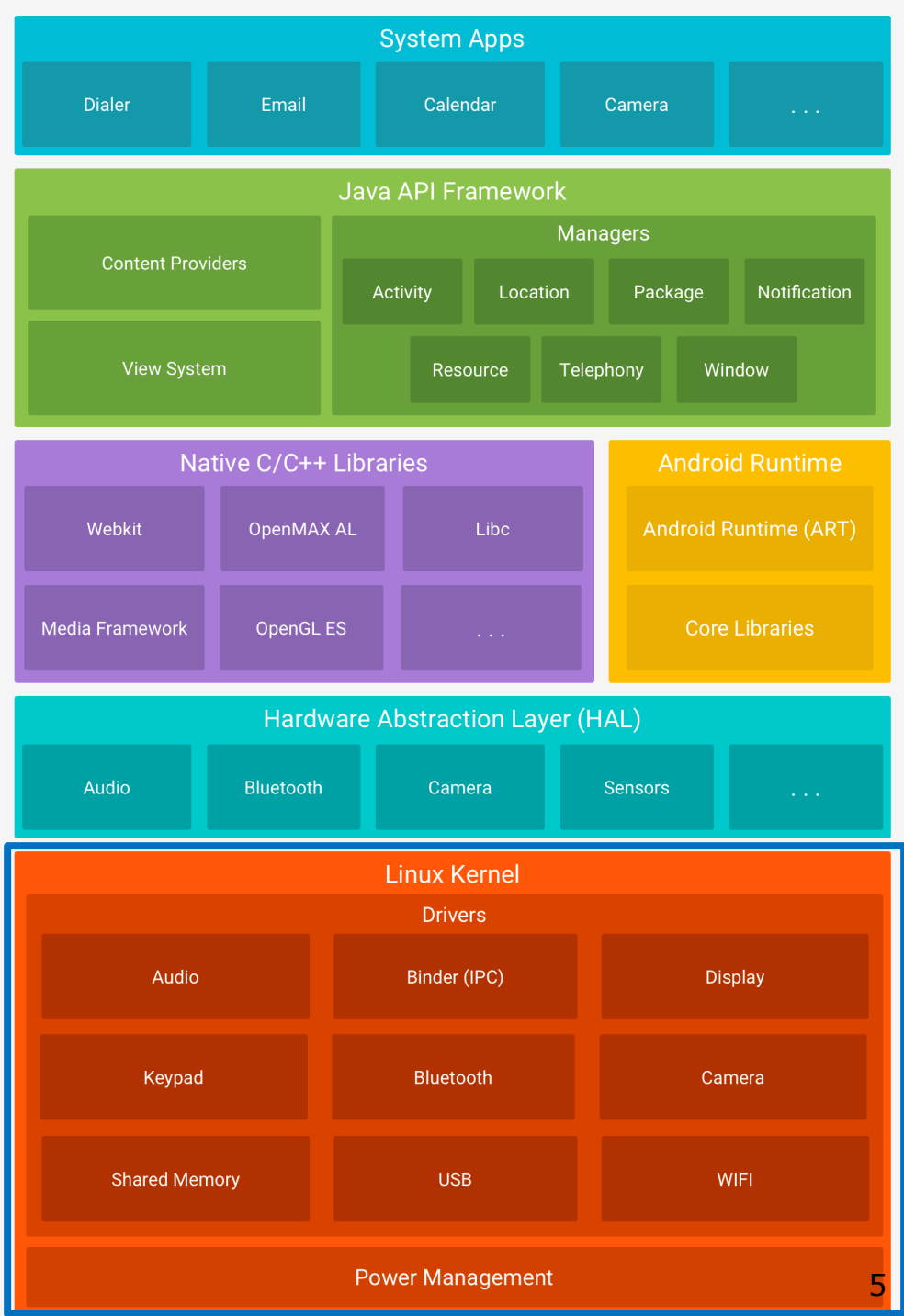
https://developer.android.com/guide/

# Αρχιτεκτονική Android



# Linux kernel (LK)

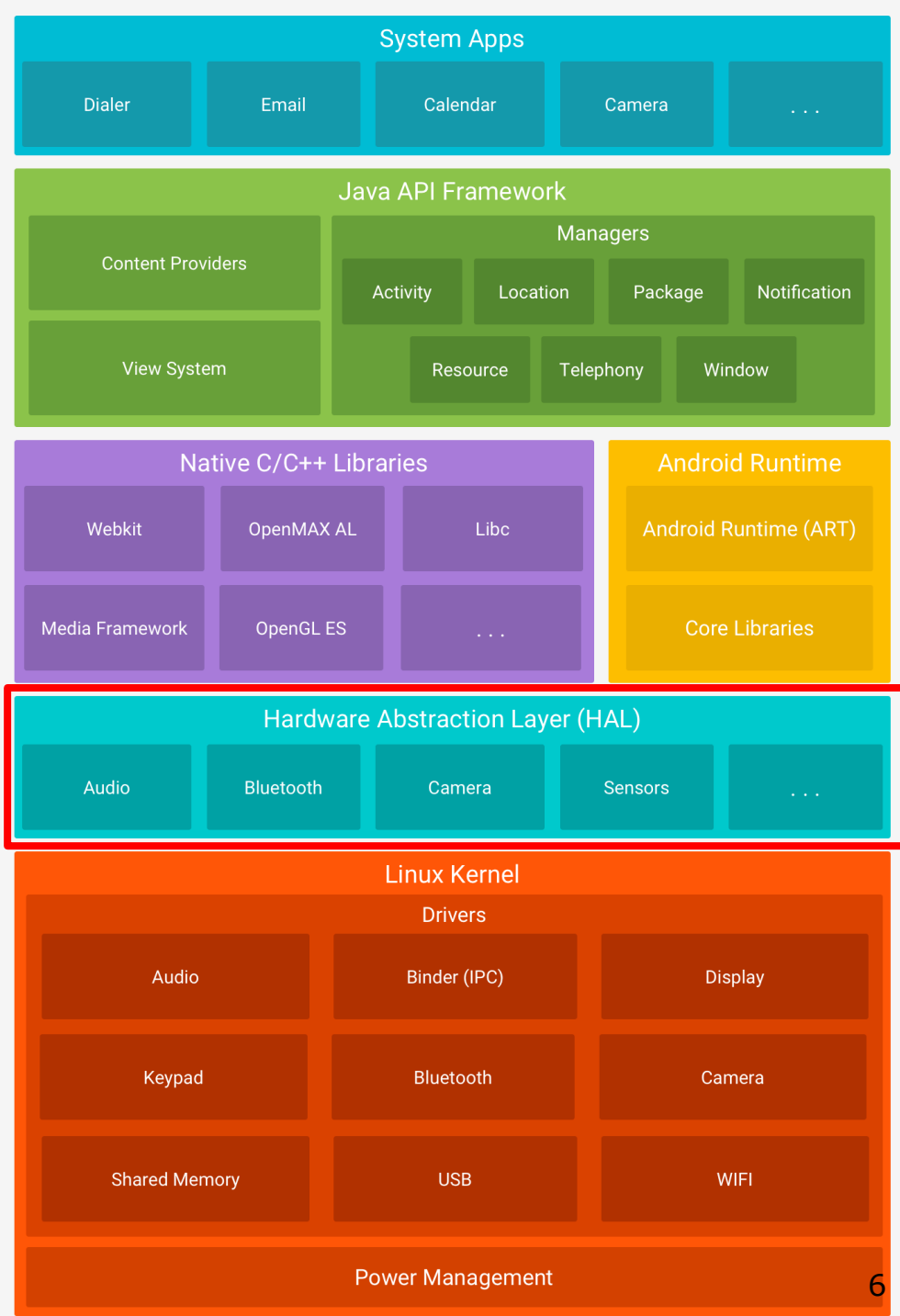
- ✓ Είναι η βάση του Android π.χ. το **Android Runtime (ART)** βασίζεται στο Linux kernel για:
  - Διαχείριση μνήμης (low-level memory management)
  - Διαχείριση διεργασιών και νημάτων
  - Διαχείριση ενέργειας
- ✓ Επιτρέπει στο Android **να εκμεταλλευτεί βασικά χαρακτηριστικά ασφάλειας** του Linux Kernel
- ✓ Επιτρέπει στους κατασκευαστές να **αναπτύσσουν drivers** σε ένα γνωστό και καλά ορισμένο kernel
- ✓ Διαδεδομένο και δοκιμασμένο ΛΣ ανοικτού κώδικα με **κατάλληλη άδεια χρήσης για το business model** των κατασκευαστών κινητών συσκευών




https://developer.android.com/guide/platform/

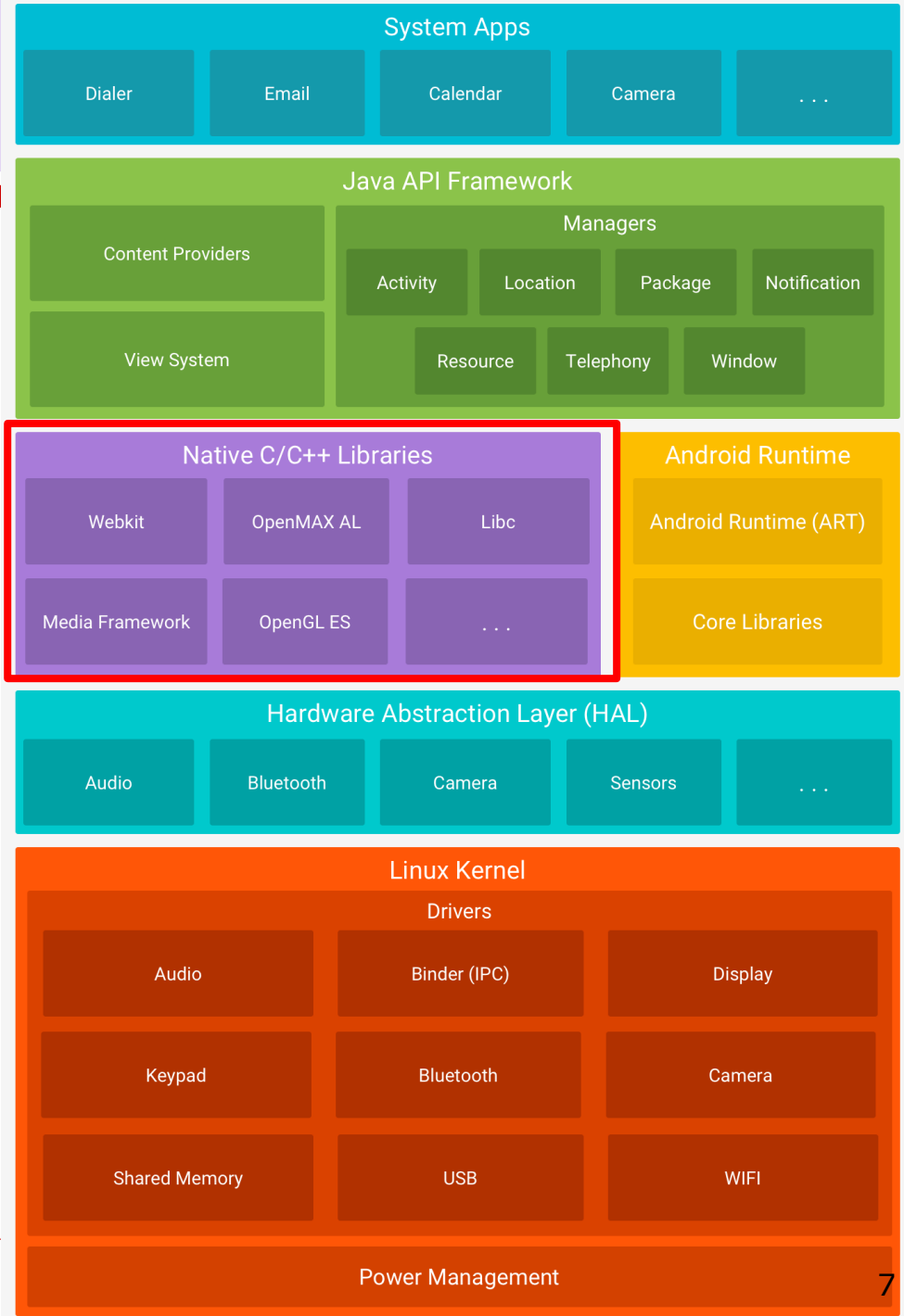
# Hardware Abstraction Layer (HAL)

- Παρέχει διεπαφές (standard interfaces) για χρήση των δυνατοτήτων των συστατικών **hardware** από το Java API framework
- Πολλαπλές βιβλιοθήκες, κάθε μία υλοποιεί μία συγκεκριμένη διεπαφή για **ένα τύπο συστατικού hardware**
- Όταν ένα framework API καλεί για χρήση μία συσκευή, το Android φορτώνει τη βιβλιοθήκη για αυτή τη συσκευή
- Απομονώνει το Linux Kernel από το API Framework



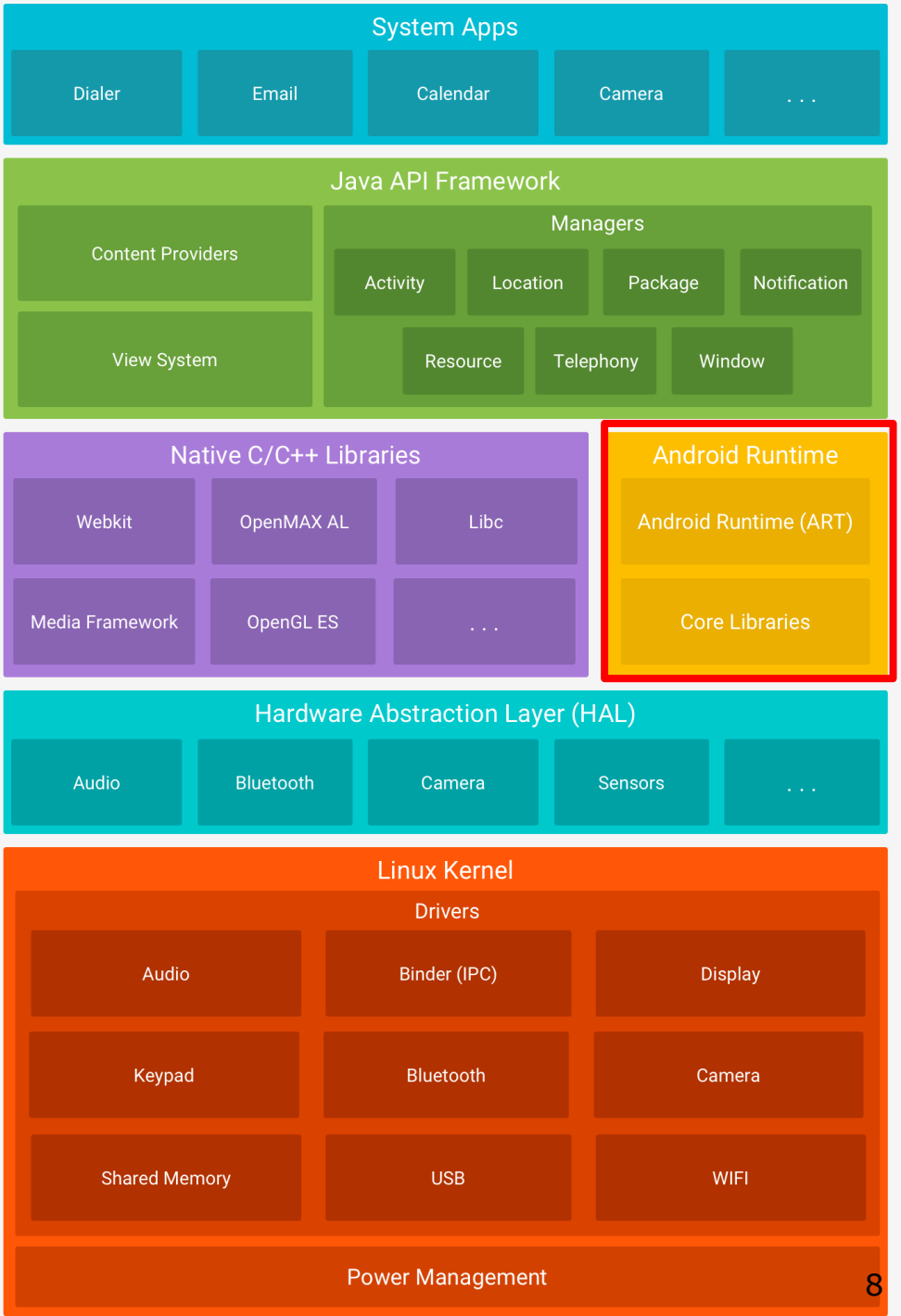
# Native C/C++ Libraries

- Πολλά συστατικά του Android system, π.χ., ART και HAL, αναπτύσσονται με **βάση κώδικα και απαιτούν βιβλιοθήκες γραμμένες σε C και C++.**
- Το Android παρέχει Java framework APIs για να διαθέτει τη λειτουργικότητα των libraries στα apps
  - π.χ. χρήση **OpenGL** ES μέσω του Android framework's Java OpenGL API για το **σχεδιασμό και χειρισμό γραφικών 2D και 3D σε μία εφαρμογή.**



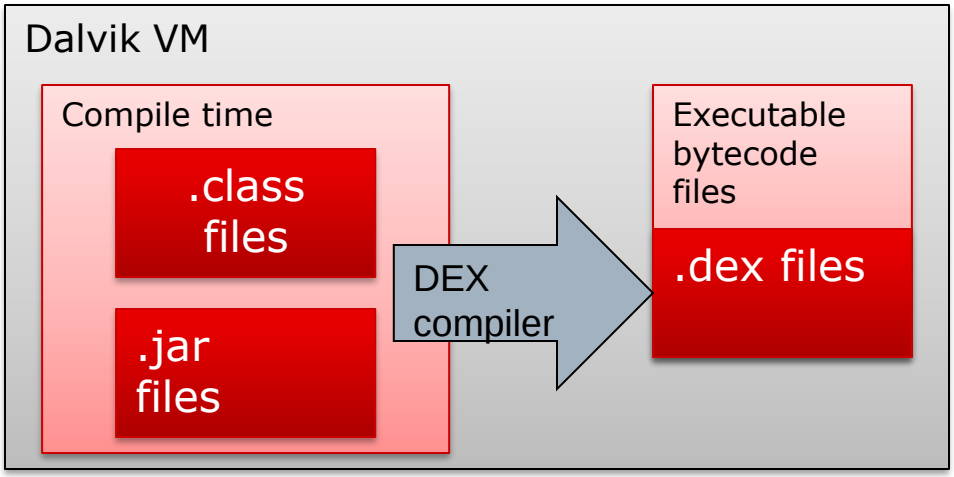
# Android Runtime Environment

- Τρέχει το εκτελέσιμο κώδικα των εφαρμογών
- **Dalvik** ή **ART** για version 5.0+ (API level 21+)
- **Κάθε app τρέχει ως διακριτή διεργασία** και σε δικό του στιγμιότυπο (instance) του ART.
- Το ART (και το Dalvik) τρέχει σε **πολλαπλά virtual machines** σε συσκευές χαμηλής μνήμης **εκτελώντας DEX files** (των apps)



# Dalvik Virtual Machine (VM)

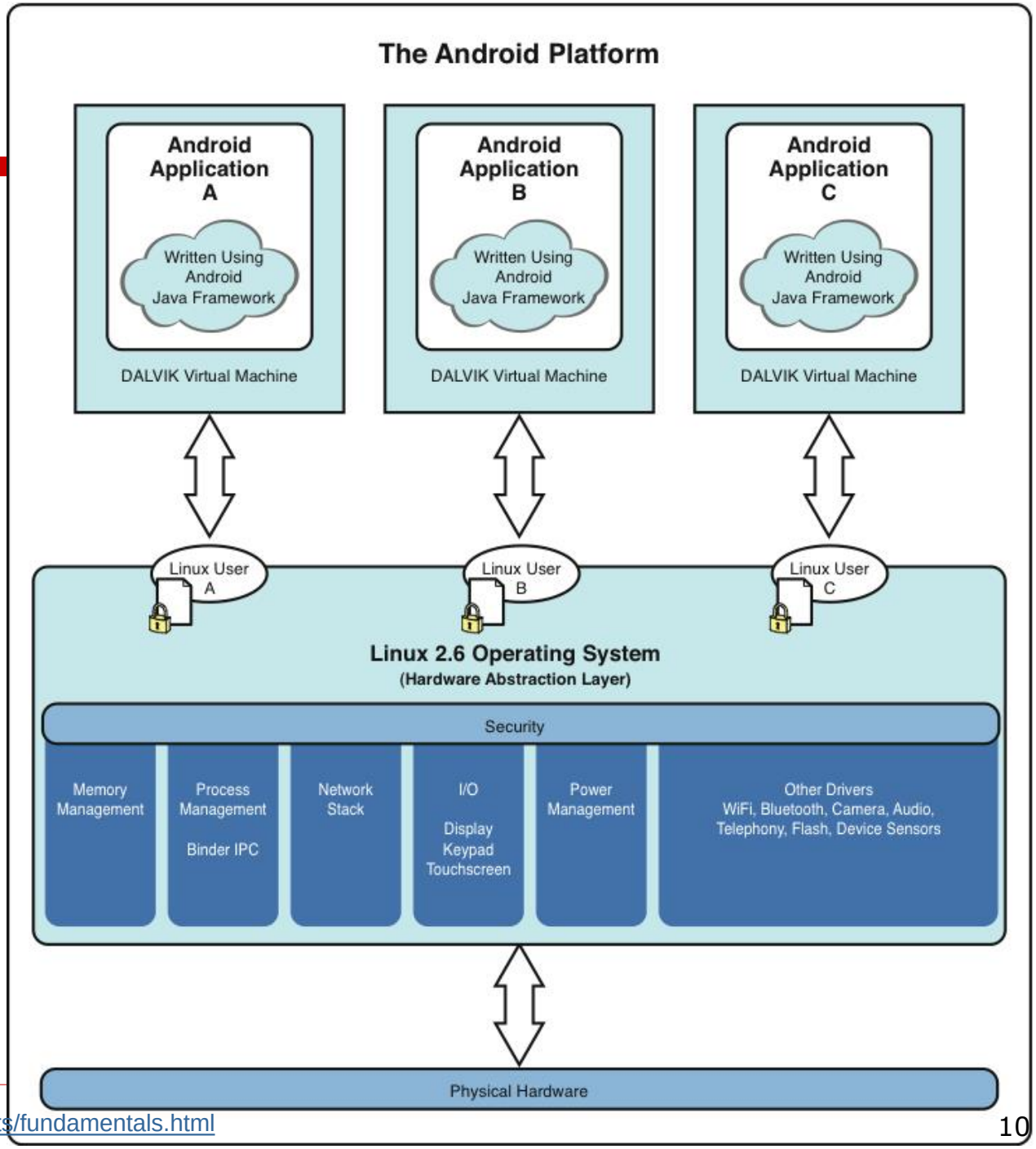
- Βελτιστοποιημένο Java VM με χαμηλές απαιτήσεις μνήμης
- Μεταγλώττιση κώδικα σε machine-independent εντολές (bytecodes) οι οποίες εκτελούνται από το Dalvik VM στη κινητή συσκευή
- Just-in-time (JIT) compilation
  - Ο DEX JIT compiler (dx tool) μεταγλωττίζει τα Java bytecode σε native machine language **κατά την εκτέλεση του κώδικα (run time)** και όχι πρωτύτερα
- Βασίζεται στο Linux Kernel για
  - Threading
  - Low-level διαχείριση μνήμης
- Δυνατότητα πολλαπλών VM στιγμιοτύπων που επωφελούνται από το Linux OS για απομόνωση ασφάλειας και διεργασιών




https://en.wikipedia.org/wiki/Dalvik_(software) https://www.javatpoint.com/dalvik-virtual-machine

# Αρχιτεκτονική Android

- Κάθε εφαρμογή τρέχει στη δική της διαδικασία Linux
- Κάθε διαδικασία έχει τη δική της Dalvik VM
- Σε κάθε εφαρμογή δίνεται ένα μοναδικό Linux user ID
- Τα Permissions καθορίζονται ώστε μια εφαρμογή να έχει πρόσβαση σε συγκεκριμένα αρχεία



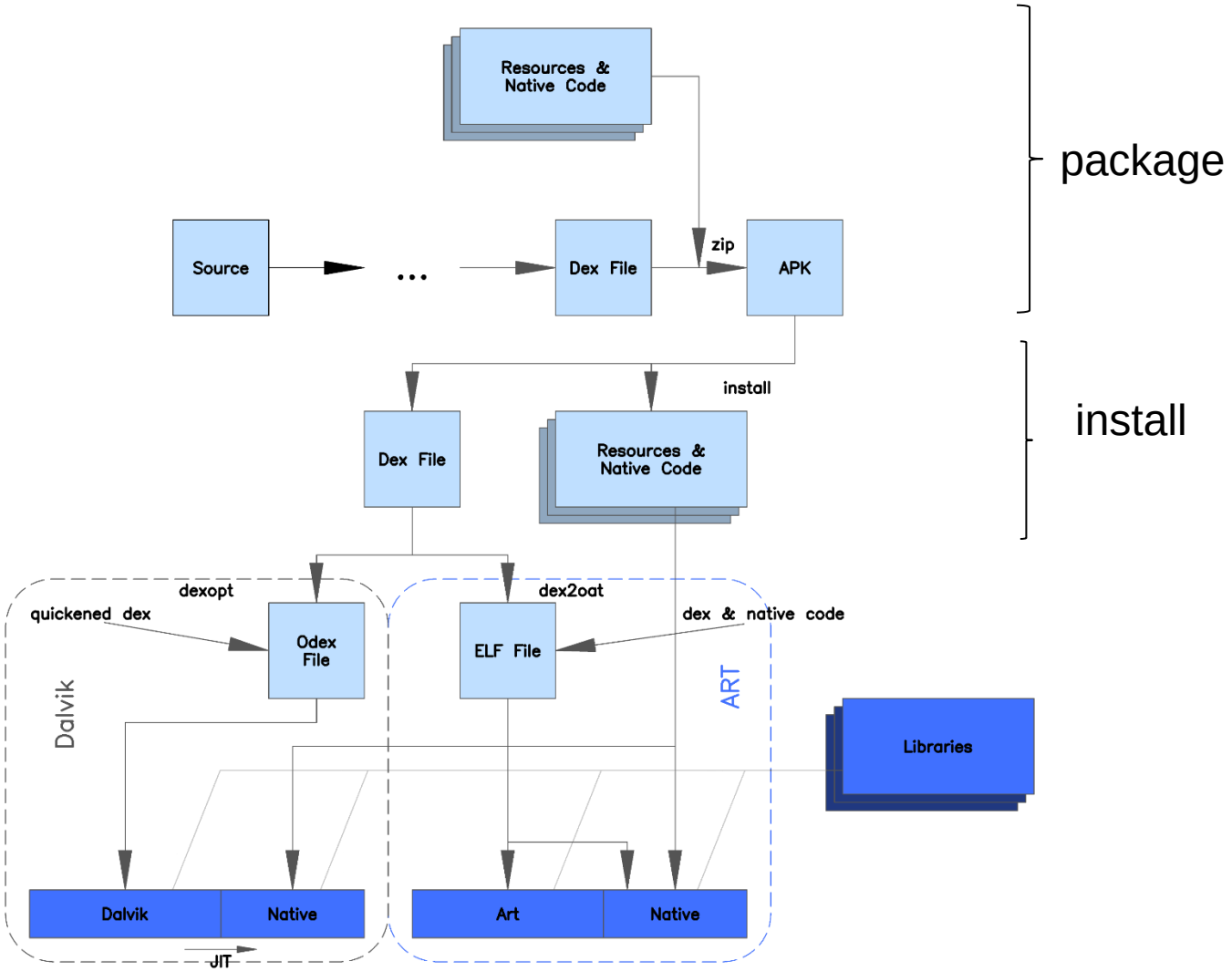

https://developer.android.com/guide/components/fundamentals.html

# Android RunTime (ART) αντί Dalvik

- Default από Android Lolipop (5.0)
- **Ahead Of Time** (AOT) compilation αντί JIT compilation
  - AOT μετατρέπει το bytecode ενός VM σε machine code **πριν την εκτέλεση** (κατά την εγκατάσταση της εφαρμογής με το dex2oat tool)
  - Καλύτερη απόδοση
  - Μικρότερη κατανάλωση ενέργειας
  - Καλύτερο garbage collection
- Καλύτερο περιβάλλον ανάπτυξης και debugging
- ART και το Dalvik είναι συμβατά, έτσι apps που αναπτύθηκαν για το Dalvik θα πρέπει να τρέχουν και με ART

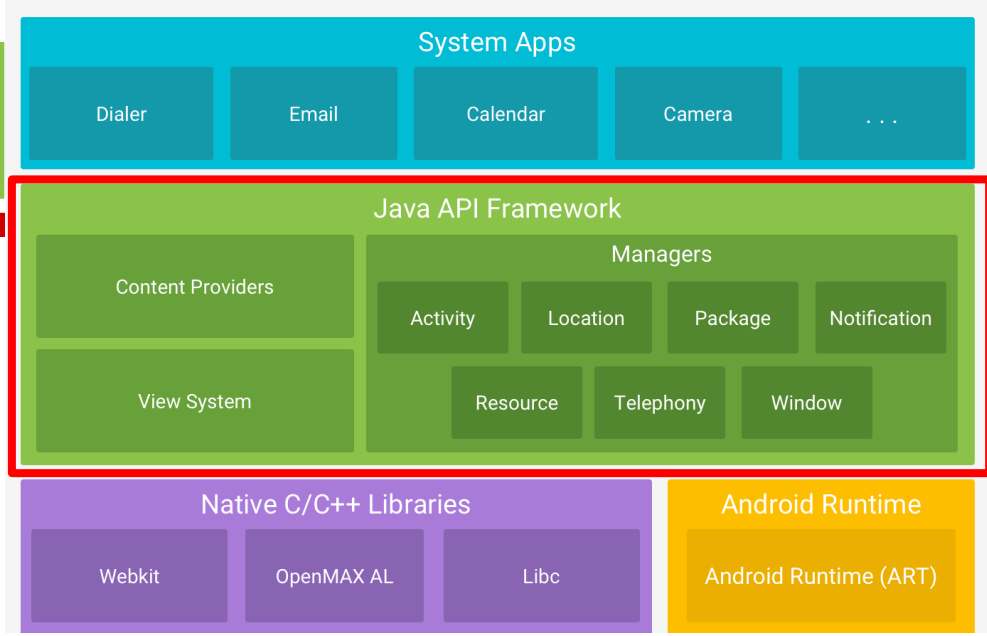
https://source.android.com/devices/tech/dalvik/

# ART vs Dalvik




https://en.wikipedia.org/wiki/Android_Runtime

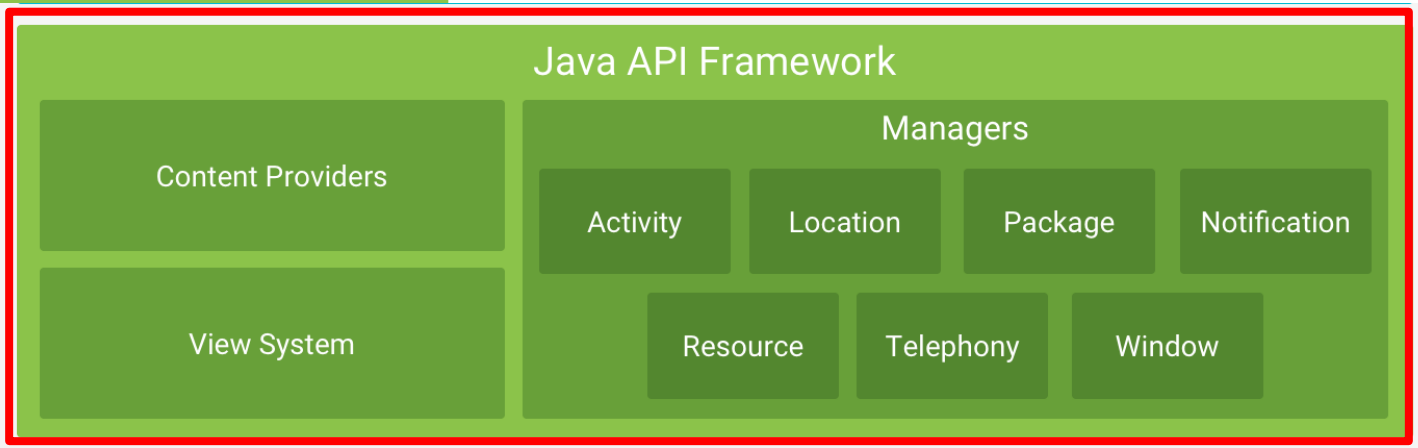
# Java API Framework




- **Όλο το set των δυνατοτήτων & χαρακτηριστικών του Android OS** είναι διαθέσιμο στον προγραμματιστή μέσω APIs γραμμένα σε Java
- Τα APIs είναι τα **βασικά building blocks** που χρειάζεται ένας προγραμματιστής για να δημιουργήσει Android εφαρμογές (apps) επαναχρησιμοποιώντας
  - βασικά δομικά συστατικά
  - υπηρεσίες
- Οι προγραμματιστές έχουν πλήρη πρόσβαση στο ίδιο Developers framework APIs που χρησιμοποιούν οι Android system apps

https://developer.android.com/guide/components/fundamentals.html

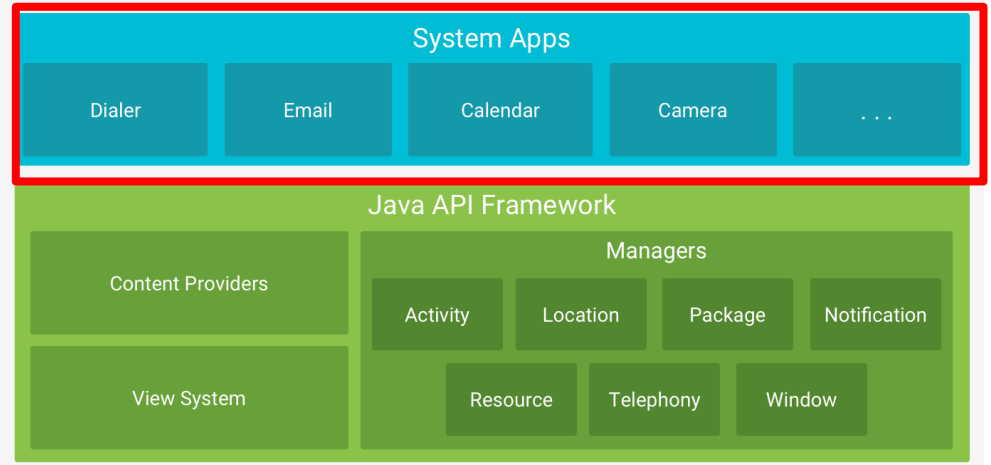
# Java API




## Κυριότερα δομικά συστατικά & υπηρεσίες:

- ✓ **View System** για την ανάπτυξη της **διεπαφής χρήσης** της εφαρμογής (app's UI) π.χ., lists, grids, text boxes, buttons  embeddable web browser
- ✓ **Content Providers:** επιτρέπει στις εφαρμογές να έχουν πρόσβαση σε δεδομένα από άλλες εφαρμογές (π.χ. εφαρμογή επαφών) ή να μοιράζουν τα δικά τους δεδομένα
- ✓ Ο **Resource Manager** παρέχει πρόσβαση σε πόρους, **αρχεία πλην των αρχείων κώδικα,** π.χ. τοπικές συμβολοσειρές (localized strings), γραφικά (graphics), και αρχεία που ορίζουν τη διάταξη (layout files)
- ✓ Ο **Notification Manager** επιτρέπει σε όλες τις εφαρμογές να εμφανίζουν μηνύματα στη γραμμή κατάστασης (custom alerts in the status bar)
- ✓ Ο **Activity Manager** που διαχειρίζεται το κύκλο ζωής των εφαρμογών και παρέχει την κοινή στοίβα πλοήγησης από εφαρμογή σε εφαρμογή με το back (navigation back stack)

# Εφαρμογές συστήματος (System Apps)




- Το Android συνοδεύεται από ένα σύνολο από βασικές εφαρμογές: email, SMS messaging, calendars, internet browsing, contacts, κ.α.
- Αυτές **δεν έχουν κάποια διάκριση** σε σχέση με αυτές που επιλέγει να εγκαταστήσει ο χρήστης.
  - Έτσι, εφαρμογές τρίτων (third-party app) μπορούν να επιλεγούν ως οι προκαθορισμένες (default) από το χρήστη για web browser, SMS messenger, πληκτρολόγιο.
  - **Εξαιρούνται οι ρυθμίσεις.**
- Παρέχουν **επαναχρησιμοποιήσιμες βασικές δυνατότητες** που μπορούν να χρησιμοποιήσει ένας προγραμματιστής σε μια νέα εφαρμογή *(πολλαπλές εισόδους)*.
  - Π.χ. εάν η εφαρμογή σας θέλετε να στείλει ένα SMS, δεν απαιτείται να αναπτύξετε εξ αρχής τη λειτουργικότητα αλλά να καλέσετε την SMS app που είναι ήδη εγκατεστημένη.

# ΔΟΜΙΚΑ ΣΥΣΤΑΤΙΚΑ ΕΦΑΡΜΟΓΩΝ (APPS)

# Βασικά δομικά συστατικά εφαρμογών (App components)

- Τι είναι τα App components
  - είναι **βασικά δομικά συστατικά** μιας εφαρμογής
  - σημείο εισόδου προς την εφαρμογή σας από
    - το σύστημα Android
    - ένα χρήστη
- Τέσσερις (4) διαφορετικοί τύποι συστατικών:
  1. Activities
  2. Services
  3. Broadcast receivers
  4. Content providers

https://developer.android.com/guide/components/fundamentals.html

# Βασικά δομικά συστατικά εφαρμογών (App components)

- Κάθε τύπος συστατικού
  - εξυπηρετεί ένα **διακριτό σκοπό**
  - έχει ένα **διακριτό κύκλο ζωής** που ορίζει το πώς κάθε συστατικό
    - δημιουργείται (created)
    - καταστρέφεται (destroyed)
- Μία εφαρμογή αποτελείται από ένα σύνολο συστατικών:
  - **Μία** ή περισσότερες activities
  - Καμία, μία ή περισσότερες
    - Services
    - Broadcast receivers
    - Content providers

# Βασικά Δομικά Συστατικά Android

- **Activity**
  - Το στρώμα (layer) παρουσίασης της εφαρμογής
  - Μια εφαρμογή μπορεί να έχει περισσότερα του ενός activities για διαχείριση διαφορετικών φάσεων του προγράμματος
  - Κάθε activity είναι υπεύθυνο να αποθηκεύει την δική του κατάσταση
- **Intent**
  - Μηχανισμός που καθορίζει ποια συγκεκριμένη ενέργεια (action) πρέπει να εκτελεστεί
  - Σε android σχεδόν ΟΛΕΣ οι αλληλοεπιδράσεις γίνονται μέσω intents
- **Service**
  - Διεργασία που τρέχει στο παρασκήνιο χωρίς τη παρέμβαση του χρήστη.
  - Δεν παρέχει UI στο χρήστη
    - Παρόμοιο με Unix daemon
  - Υπάρχουν πολλά built-in services σε Android
- **Content Providers**
  - Επιτρέπουν σε μια εφαρμογή να έχει πρόσβαση σε δεδομένα άλλων εφαρμογών (π.χ. Contacts)
  - Επιτρέπουν σε μια εφαρμογή να διαμοιράζει τα δεδομένα που παράγει σε άλλες εφαρμογές

# Activity ένα συστατικό εφαρμογής (1)

- Αφορά το στρώμα παρουσίασης (**presentation layer**) της εφαρμογής.
  - Αντιστοιχεί σε **μία οθόνη (screen**) της εφαρμογής
- Είναι το **σημείο εισόδου** για διάδραση με το χρήστη
- Μια εφαρμογή μπορεί να έχει περισσότερα του ενός activities για διαχείριση διαφορετικών οθονών της εφαρμογής
  - π.χ. μία εφαρμογή email μπορεί να έχει μία activity που δείχνει τη λίστα με νέα emails, μία άλλη για τη σύνθεση email, μία τρίτη για την ανάγνωση των emails κ.α.
- **Παρόλου που παρέχουν μία ενιαία εμπειρία χρήσης, κάθε activity είναι ανεξάρτητη από την άλλη**
- Κάθε activity είναι υπεύθυνο να αποθηκεύει την δική του κατάσταση

https://developer.android.com/guide/components/fundamentals.html
https://developer.android.com/guide/components/activities/intro-activities

# Activity ένα συστατικό εφαρμογής (2)

- **Επαναχρησιμοποιήσιμο δομικό συστατικό**
  - Μια άλλη εφαρμογή μπορεί να καλέσει και ξεκινήσει μία από αυτές τις activities (σημείο εισόδου), αρκεί να τις έχουν παραχωρηθεί τα απαραίτητα δικαιώματα
  - Χρησιμοποιεί το μηχανισμό intent
    - Explicit: “Start the Send Email activity in the Gmail app.”
    - Implicit: “Start a Send Email screen in any activity that can do the job.”

# Activity ένα συστατικό εφαρμογής (3)

- Κάθε activity έχει το δικό του **κύκλο ζωής** και είναι υπεύθυνο να αποθηκεύει την δική του κατάσταση κατά το κύκλο ζωής του
- Ο **Activity Manager** παρακολουθεί τις ενεργές activities
  - διαχειρίζεται αυτή που είναι στο προσκήνιο και όσες είναι στο παρασκήνιο
  - τις καταστάσεις που βρίσκονται (διαχειρίζεται το κύκλο ζωής των εφαρμογών) και
  - παρέχει την **κοινή στοίβα** πλοήγησης από εφαρμογή σε εφαρμογή με το back (navigation back stack)

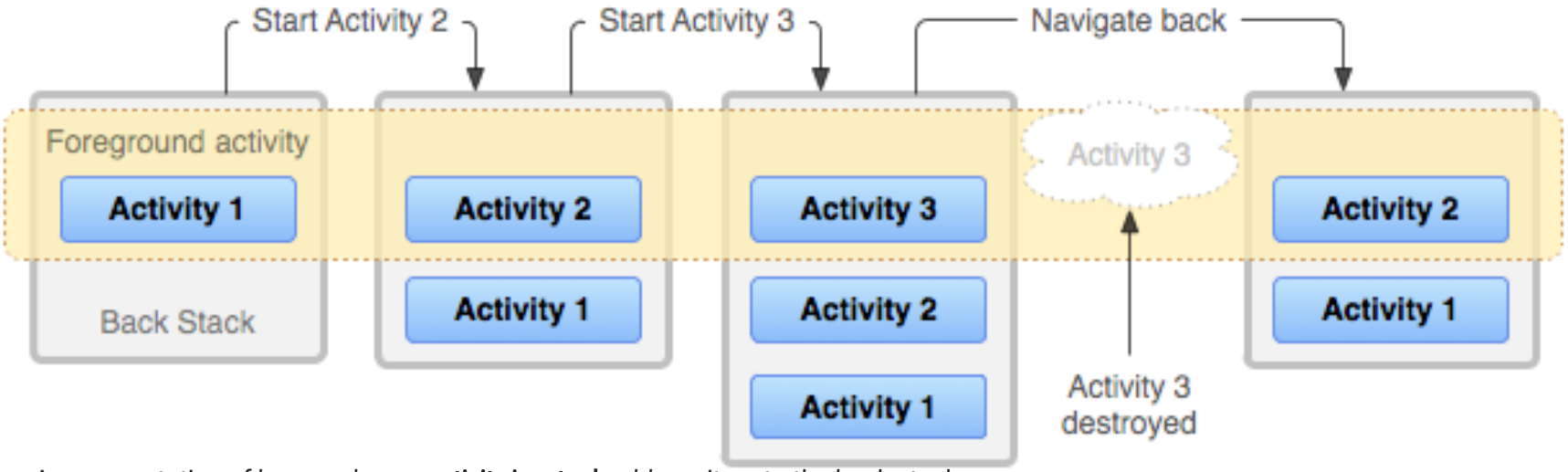
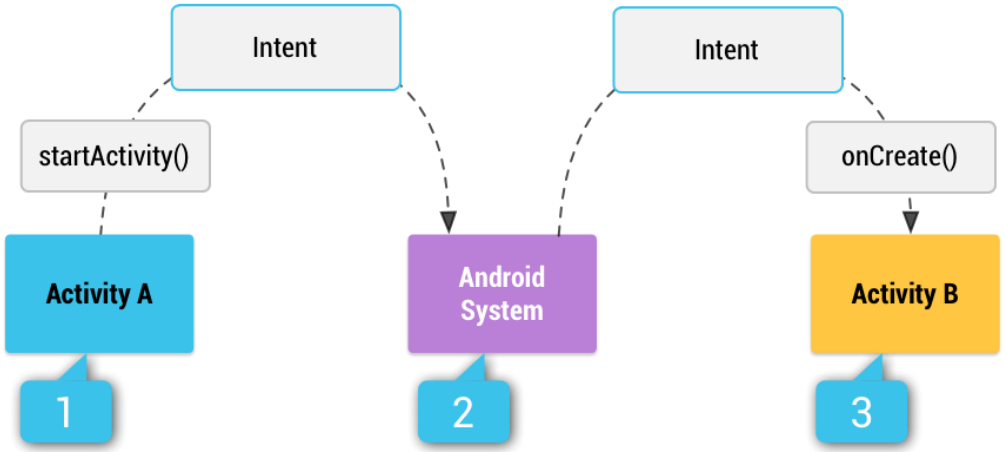



**Figure.** A representation of how each **new activity in a task** adds an item to the back stack.
When the user presses the **Back** button, the current activity is destroyed and the previous activity resumes.

https://developer.android.com/guide/components/activities/tasks-and-back-stack

# Εκτέλεση συστατικών: ο μηχανισμός Intent

- Σε android σχεδόν ΟΛΕΣ οι αλληλοεπιδράσεις γίνονται μέσω intents
- Μηχανισμός που καθορίζει ποια συγκεκριμένη ενέργεια (action) πρέπει να εκτελεστεί στέλνοντας **ασύγχρονα μηνύματα**
- Ο ρόλος ενός intent είναι να αιτηθεί **– στέλνοντας μήνυμα προς τον ActivityManagerService** για μια ενέργεια που ο χρήστης επιθυμεί να γίνει
- Μέσω των intents μπορούμε να ξεκινήσουμε την εκτέλεση των Activities, των Services και των Broadcast Receivers **(όχι Content Providers)**




https://developer.android.com/guide/components/intents-filters?hl=en

# Tasks & Multitasking

1. Έστω ότι η **τρέχουσα εργασία** είναι το **Task A** με 2 activities, μία στο προσκήνιο (foreground) και μία στο παρασκήνιο (background).
2. Ο χρήστης πατά το ***Home* button** και ξεκινά μία **νέα app** από τον app launcher.
3. Όταν εμφανίζεται το **Home screen**, το **Task A** πηγαίνει στο **παρασκήνιο**.
4. Όταν ξεκινά η νέα app, το σύστημα ξεκινά ένα **νέο task** για την app (Task B) με τη δική της **στοίβα** (stack) από **activities**.
5. Αφού διαλεχθεί με αυτή την app, ο χρήστης **επιστρέφει στη Home** πάλι και επιλέγει την προηγούμενη app (Task A).
6. Η Task A έρχεται στο προσκήνιο, όλες οι activities στην στοίβα της είναι άθικτες και η activity στην κορυφή της στοίβας συνεχίζει.
7. Ο χρήστης μπορεί πάλι να επιστρέψει στην Task B (μέσω της Home) ή από τα πρόσφατα screens ή να καλέσει μία άλλη app.

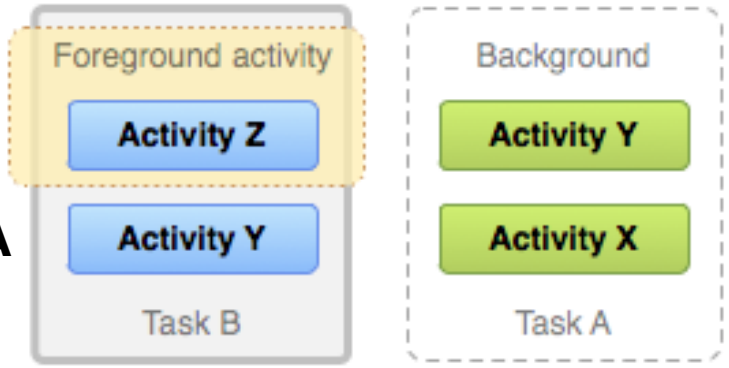


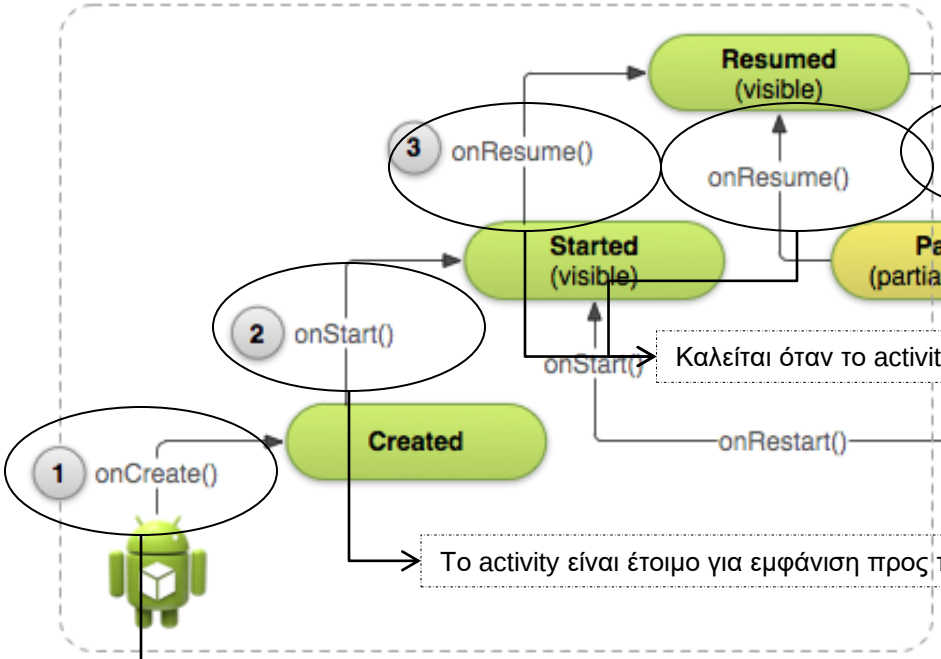

**Figure.** Two tasks: Task B receives user interaction in the foreground, while Task A is in the background, waiting to be resumed.

https://developer.android.com/guide/components/activities/tasks-and-back-stack

# Activity: Καταστάσεις και callbacks

1. Αρχικοποίηση UI
   - `onCreate()` -> CREATED
2. Εμφανίζεται το UI αλλά δεν είναι έτοιμο για αλληλεπίδραση – μετάβαση
   - `onStart()` -> STARTED
3. Αλληλεπίδραση με το χρήστη
   - `onResume()` -> RESUMED
4. Ο χρήστης αφήνει την τρέχουσα activity, η οποία φεύγει από το προσκήνιο φαίνεται ακόμη, μετάβαση
   - `onPause()` -> PAUSED
5. Δεν φαίνεται στην οθόνη, δεν καταστρέφεται παραμένει σε αναμονή στη στοίβα
   - `onStop()` -> STOPPED
6. Ολοκληρώνεται ή καταστρέφεται
   - `onDestroy()` -> DESTROYED

# Activity Lifecycle




Εκτελείται όταν το activity είναι έτοιμο να πάει στο παρασκήνιο (background), συνήθως επειδή ένα άλλο activity ξεκινάει. Εδώ πρέπει να αποθηκεύονται όποιες αλλαγές έχει κάνει το πρόγραμμα.

Καλείται όταν το activity δεν είναι ορατό πλέον στο χρήστη

Καλείται όταν το activity είναι έτοιμο να αλληλεπιδρά με τον χρήστη.`

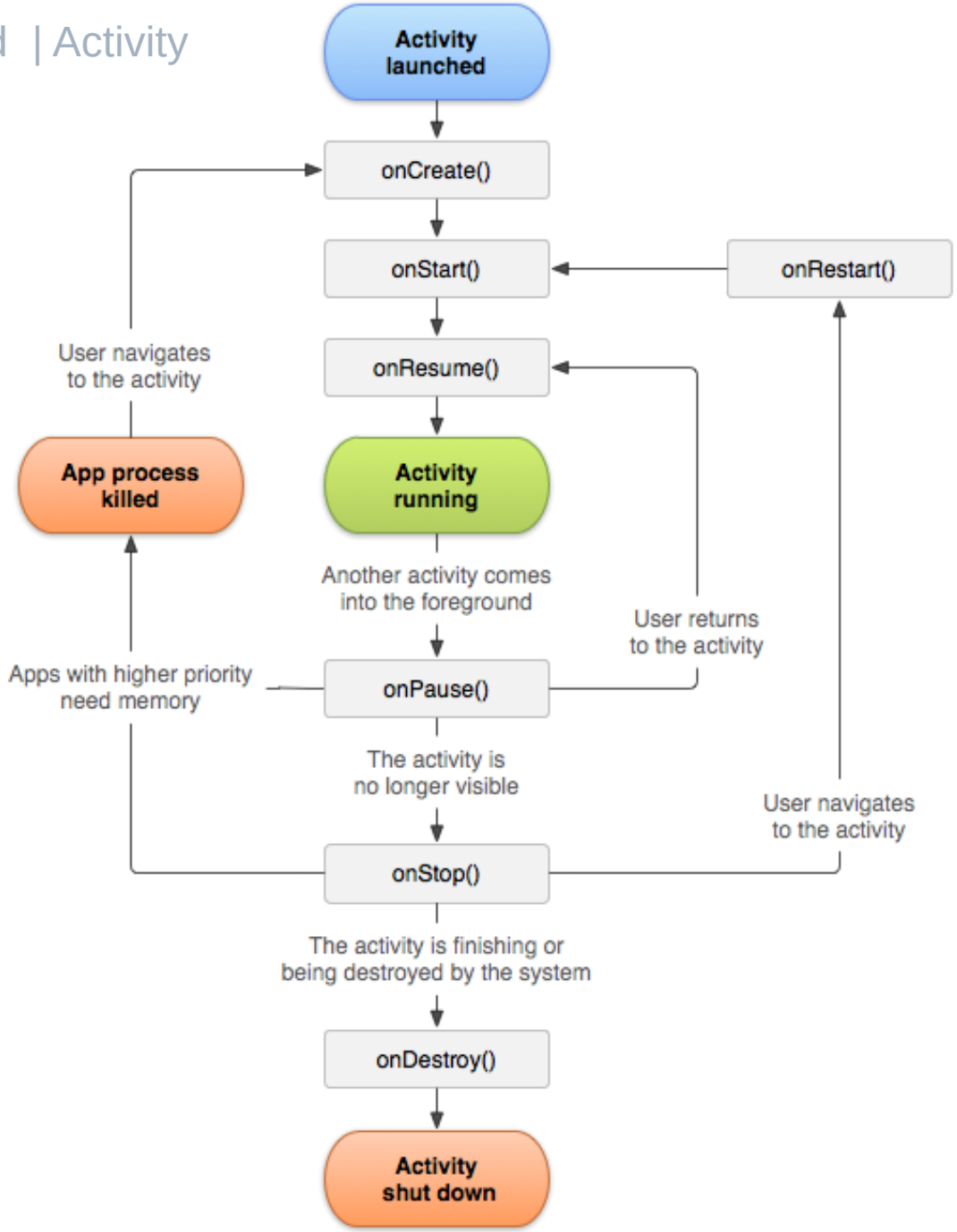
Το activity είναι έτοιμο για εμφάνιση προς το χρήστη

Καλείται όταν το activity καταστρέφεται

Καλείται όταν το activity πρωτοξεκινά. Μπορεί να χρησιμοποιηθεί για αρχικοποίηση της διεπαφής χρήστη. Παίρνει μία παράμετρο null ή πληροφορίες κατάστασης που είχαν αποθηκευτεί από την μέθοδο onSaveInstanceState()

https://developer.android.com/reference/android/app/Activity.html

# Κύκλος ζωής ενός Activity




https://developer.android.com/guide/components/activities/activity-lifecycle

# Κλάση Activity

- Κάθε διεπαφή χρήστη (οθόνη) αναπαρίσταται από μία κλάση Activity
- Μία εφαρμογή έχει ένα ή περισσότερα activities και μία διεργασία Linux που τα περιέχει
- **extends Activity** (π.χ. MainActivity extends Activity)
- **override onCreate() callback**
  - Πυροδοτείται όταν το Android σύστημα δημιουργεί την activity
  - Εδώ αρχικοποιεί κανείς τα βασικά συστατικά της activity π.χ. ορίζει το layout της

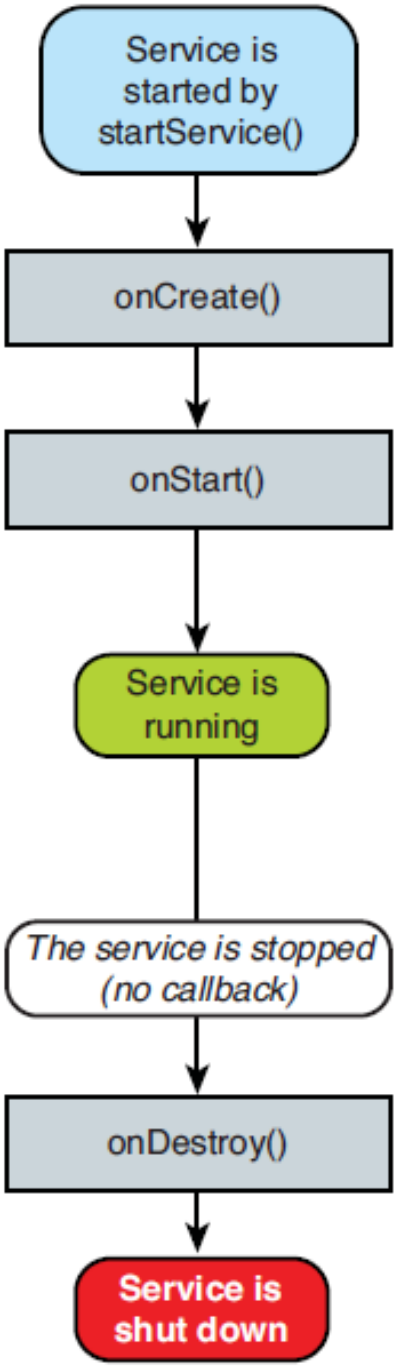
# Service ένα συστατικό εφαρμογής

- Διεργασία που τρέχει στο **παρασκήνιο** χωρίς τη παρέμβαση του χρήστη για να εκτελέσει λειτουργίες
  - λειτουργίες μακράς διάρκειας
  - λειτουργίες για άλλες απομακρυσμένες διεργασίες
- Δεν παρέχει UI στο χρήστη
  - Αναπαραγωγή μουσικής ενώ ο χρήστης είναι σε άλλη app
  - Να κατεβάσει ή συγχρονίσει δεδομένα χωρίς να μπλοκάρει το διάδραση του χρήστη με άλλη app
- Παρόμοιο με Unix daemon
- Υπάρχουν πολλά built-in services σε Android

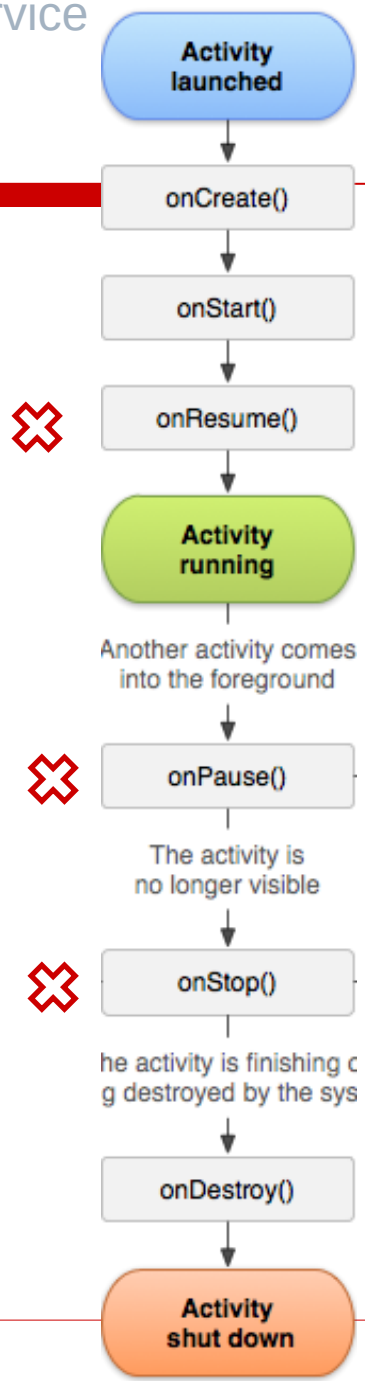
https://developer.android.com/guide/components/fundamentals.html

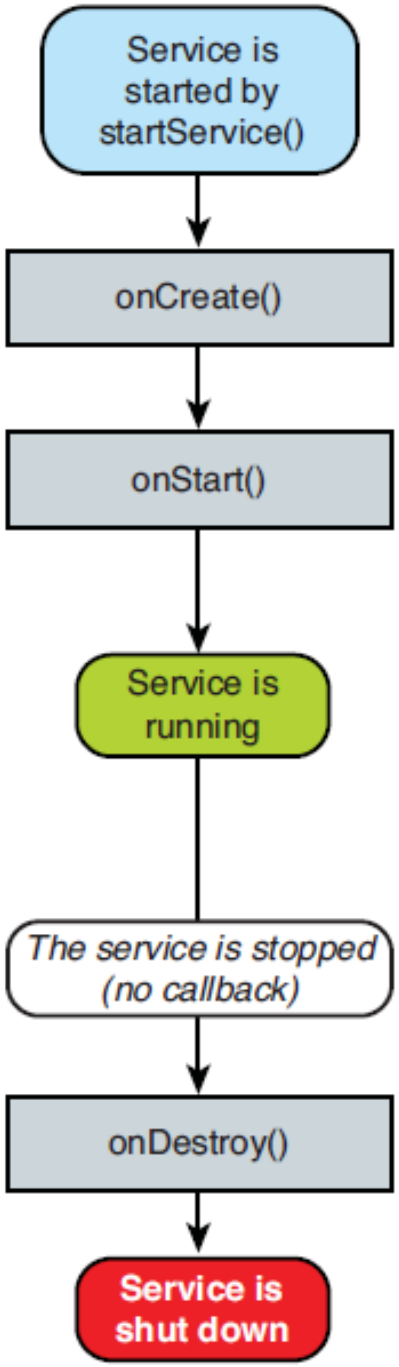
# Η κλάση Service

- Η βασική κλάση για διαδικασίες που εκτελούνται στο παρασκήνιο
  - **extends Service**
  - **override onCreate()**
- Δεν έχει γραφική διεπαφή
- Από default δεν είναι μια ξεχωριστή διεργασία ούτε ένα ξεχωριστό thread. **Αποτελεί τμήμα του main thread**.
  - Αν επιθυμούμε μπορούμε να καθορίσουμε ότι θα τρέχει σε ξεχωριστή διεργασία
- Παραδείγματα χρήσης
  - network downloading
  - Αναπαραγωγή μουσικής στο παρασκήνιο
  - TCP/UDP Server



# Σύγκριση life cycle




https://developer.android.com/guide/components/services.html

# Broadcast Receivers (RB) ένα συστατικό εφαρμογής

- Οι ανακοινώσεις broadcast μπορεί να
  - Παράγονται από το λειτουργικό (**OS-generated**) π.χ. χαμηλή στάθμη μπαταρίας, αποκατάσταση σύνδεσης Wifi, πάτημα του πλήκτρου της φωτογραφικής
    - Το σύστημα μπορεί να στείλει broadcasts ακόμη και σε εφαρμογές που δεν τρέχουν
  - Παράγονται από το χρήστη (**User-generated**) π.χ. έναρξη ή τερματισμός μια διαδικασίας, ενεργοποίηση ενός χαρακτηριστικού
- Η εφαρμογή, μέσω ενός BR, λαμβάνει ανακοινώσεις broadcast και αντιδρά κατάλληλα
  - Προκαλούν την δημιουργία intents που μπορούν προκαλέσουν την εκτέλεση κώδικα εκτός της κανονικής (ροής regular user flow)
- Παραδείγματα χρήσης
  - Ενημέρωση άλλων εφαρμογών ότι δεδομένα έχουν κατεβεί
  - Δεν διαθέτουν διεπαφή χρήστη αλλά μπορούν να δημιουργήσουν ένα status bar notification για να ειδοποιήσουν το χρήστη εάν ένα broadcast event συμβαίνει
  - Ως gateway με άλλα συστατικά με σκοπό να κάνει ελάχιστη δουλειά

# Content Providers ένα συστατικό εφαρμογής

- Επιτρέπουν σε μια εφαρμογή να έχει πρόσβαση σε δεδομένα άλλων εφαρμογών (π.χ. Contacts)
- Επιτρέπουν σε μια εφαρμογή να διαμοιράζει τα δεδομένα που παράγει σε άλλες εφαρμογές

# ΤΑ ΠΡΩΤΑ ΣΑΣ ΒΗΜΑΤΑ ΣΤΟ ANDROID SDK

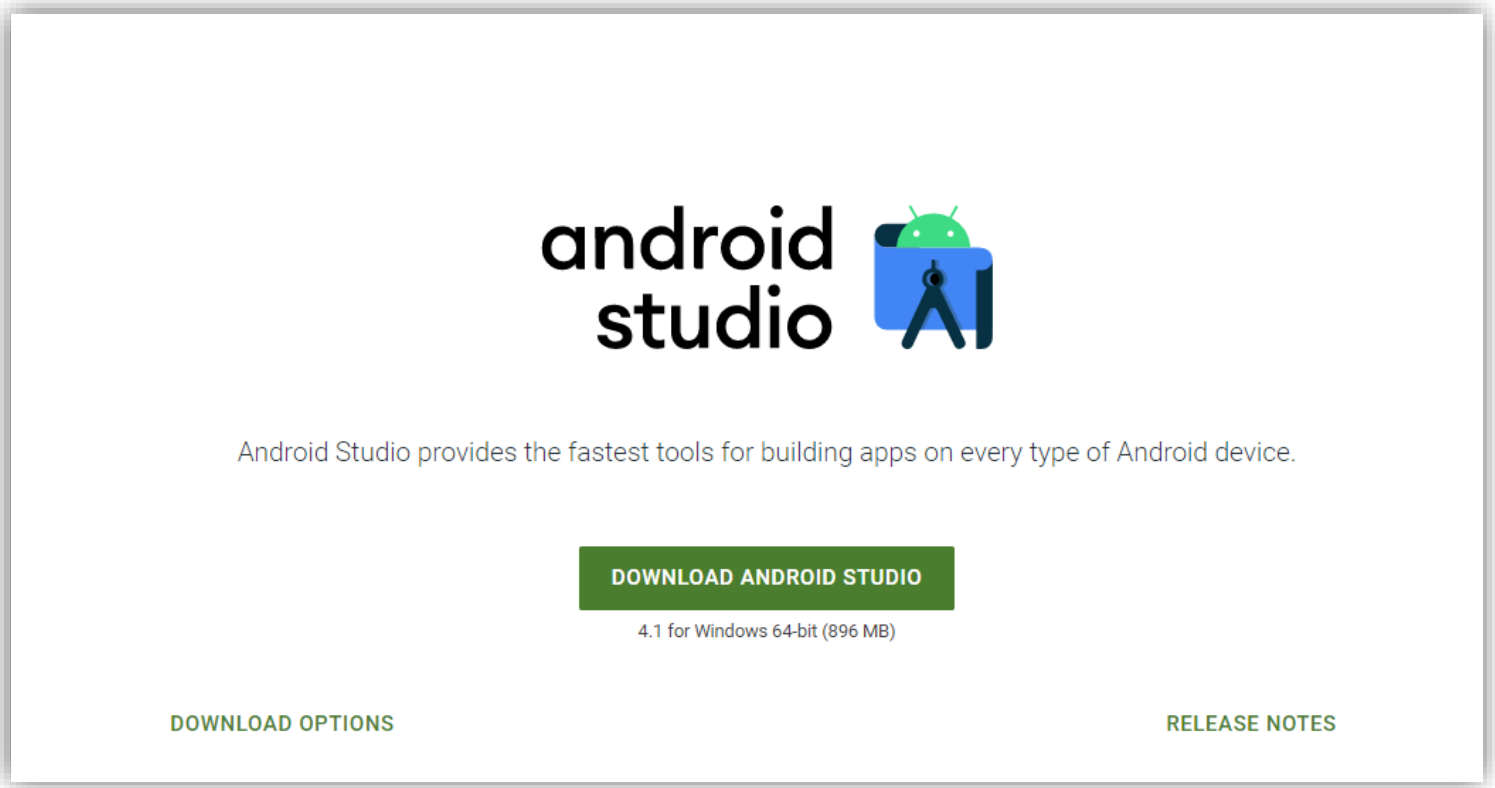
# Εγκατάσταση Android SDK

- Λήψη του Android SDK
- Εγκατάσταση plugin ανάλογα με το περιβάλλον IDE
  - **Android Studio**
  - ADT plugin για Eclipse
  - NBAndroid plugin για Netbeans
- Λήψη SDK tools and platforms μέσω του SDK Manager

**Διάθεση Android SDK**
https://developer.android.com/studio

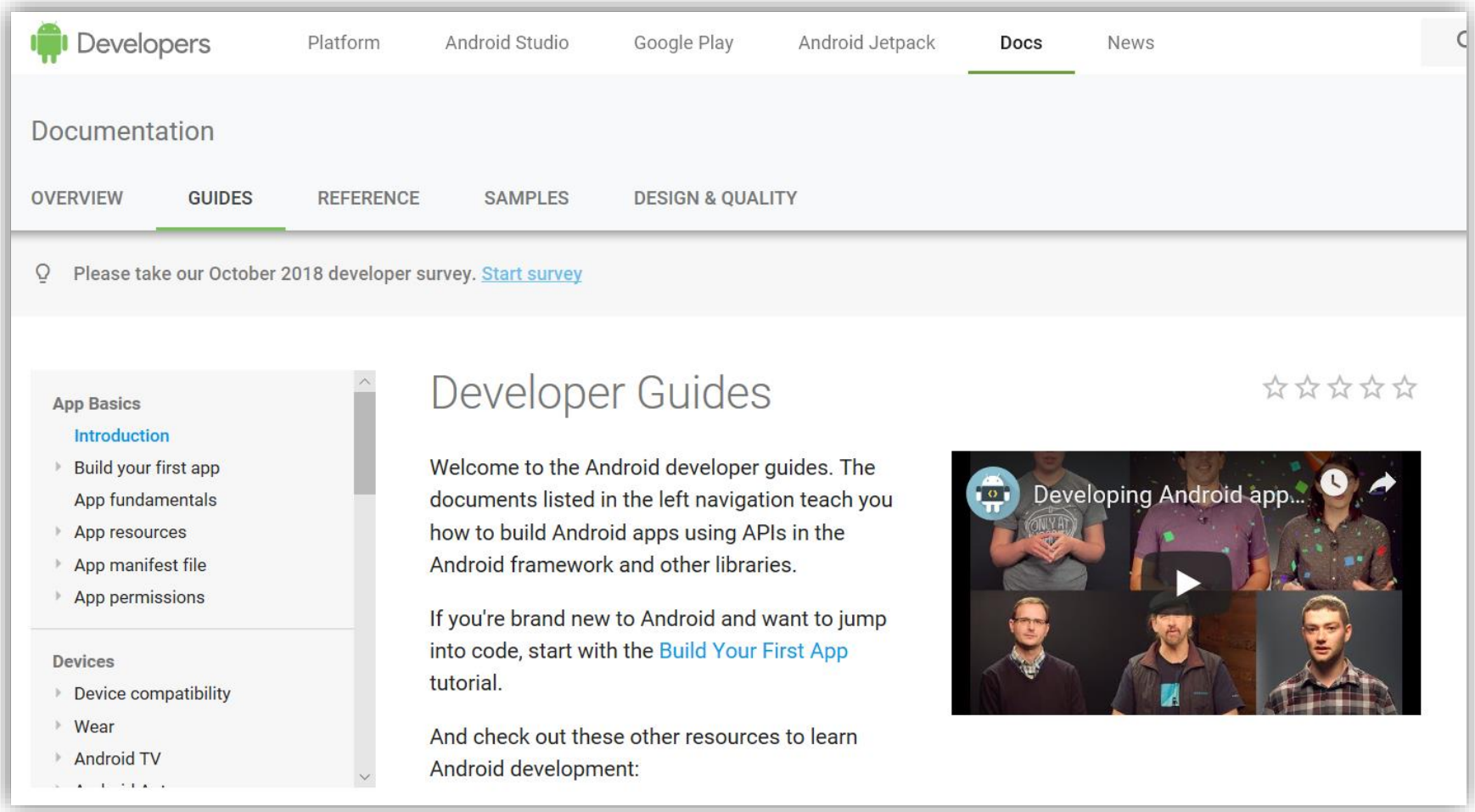
**User Guides Android SDK**
https://developer.android.com/studio/intro
https://developer.android.com/guide/

# Διάθεση Android SDK




**Σύνδεσμοι:**
https://developer.android.com/studio
https://developer.android.com/studio/intro/

# Οδηγίες για την πρώτη σας εφαρμογή



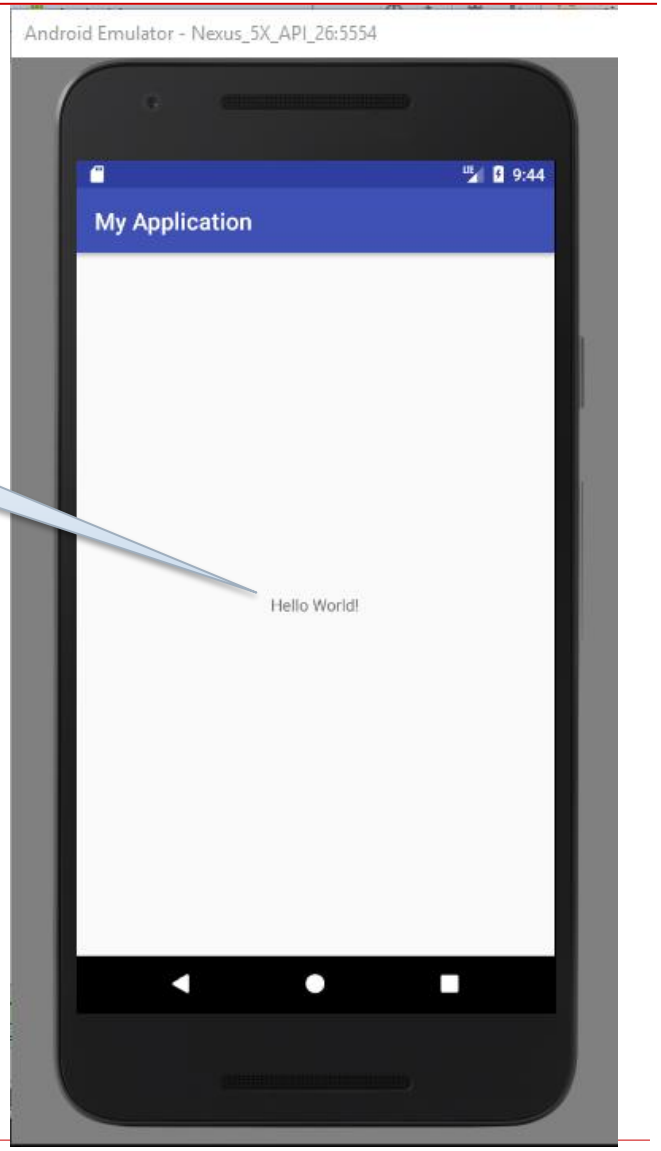

**Σύνδεσμοι:** https://developer.android.com/guide/
https://developer.android.com/training/basics/firstapp/index.html

# Εκτέλεση εφαρμογής

- Emulator
- Στην Android συσκευή σας

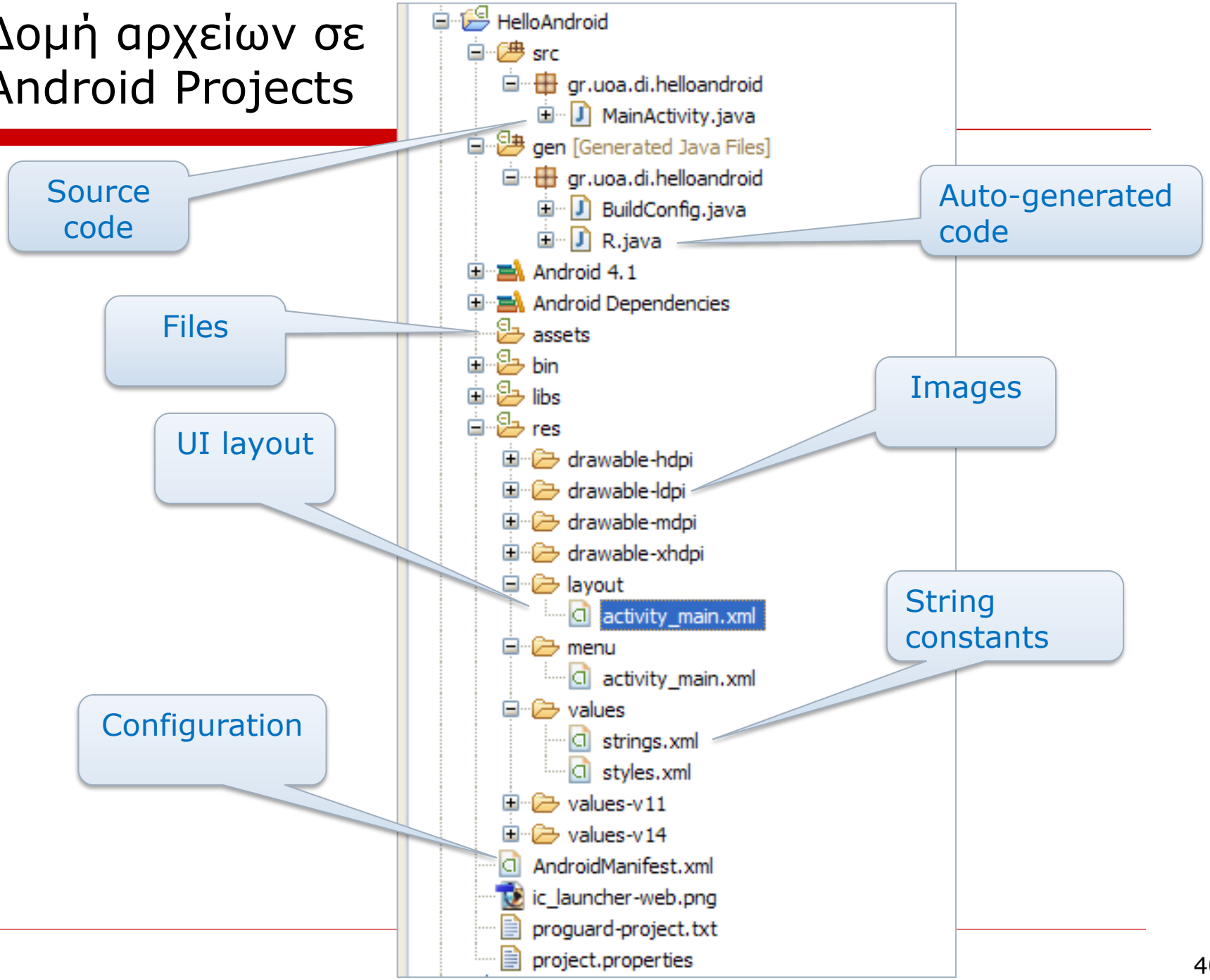
# Παράδειγμα εφαρμογής με ένα Activity

- Hello World !




TextView Component

# Δομή αρχείων σε Android Projects



# Android manifest file

Κάθε εφαρμογή έχει ένα αρχείο **AndroidManifest.xml** που παρέχει σημαντικές πληροφορίες για την εφαρμογή στο OS

- Δίνει **όνομα στο Java πακέτο** για την εφαρμογή το οποίο αποτελεί μοναδικό αναγνωριστικό για την εφαρμογή.
- Καθορίζει το **ελάχιστο επίπεδο Android API** που απαιτείται από την εφαρμογή.
- Περιέχει λίστα με τις βιβλιοθήκες με τις οποίες πρέπει να συνδεθεί η εφαρμογή.
- Καθορίζει **τα δικαιώματα** που πρέπει να έχει η εφαρμογή
  - για να έχει πρόσβαση σε προστατευόμενα κομμάτια του API
  - για να αλληλεπιδρά με άλλες εφαρμογές
  - τα δικαιώματα που πρέπει να έχουν άλλες εφαρμογές για να αλληλεπιδρούν με αυτή
- Δηλώνει τα χαρακτηριστικά (h/w & s/w), π.χ. κάμερα, Bluetooth κ.α., που χρησιμοποιεί η εφαρμογή.
- **Περιγράφει τα components της εφαρμογής (activities, services, content providers κλπ.).**

http://developer.android.com/guide/topics/manifest/manifest-intro.html

```
<?xml version="1.0" encoding="utf-8"?>

<manifest>

    <uses-permission />
    <permission />
    <permission-tree />
    <permission-group />
    <instrumentation />
    <uses-sdk />
    <uses-configuration />
    <uses-feature />
    <supports-screens />
    <compatible-screens />
    <supports-gl-texture />

    <application>

        <activity>
            <intent-filter>
                <action />
                <category />
                <data />
            </intent-filter>
            <meta-data />
        </activity>

        <activity-alias>
            <intent-filter> . . . </intent-filter>
            <meta-data />
        </activity-alias>

        <service>
            <intent-filter> . . . </intent-filter>
            <meta-data/>
        </service>

        <receiver>
            <intent-filter> . . . </intent-filter>
            <meta-data />
        </receiver>

        <provider>
            <grant-uri-permission />
            <meta-data />
            <path-permission />
        </provider>

        <uses-library />

    </application>

</manifest>
```

# Android manifest file (2/2)

- Παράδειγμα από την “HelloWorld” app

```
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="gr.uoa.di.helloandroid"
    android:versionCode="1"
    android:versionName="1.0" >

    <uses-sdk
        android:minSdkVersion="8"
        android:targetSdkVersion="15" />
    <uses-permission android:name="android.permission.INTERNET" />
    <application
        android:icon="@drawable/ic_launcher"
        android:label="@string/app_name"
        android:theme="@style/AppTheme" >
        <activity
            android:name=".MainActivity"
            android:label="@string/title_activity_main" >
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />

                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
    </application>

</manifest>
```

SDK versions

Application permissions

Activities

Intents

# Απλό αρχείο string resources

/res/values/strings.xml

String Hello word!

```
<resources>
    <string name="app_name">HelloAndroid2</string>
    <string name="hello_world">Hello world!</string>
    <string name="menu_settings">Settings</string>
    <string name="title_activity_main">MainActivity</string>
</resources>
```

# Κλάση Activity

- extends Activity (π.χ. MainActivity extends Activity)
- override onCreate() callback
  - Πυροδοτείται όταν το Android σύστημα δημιουργεί την activity
  - Εδώ αρχικοποιεί κανείς τα βασικά συστατικά της activity π.χ. ορίζει το layout της

# Απλή Κλάση Activity

/src/gr/uoa/di/helloandroid/MainActivity.java

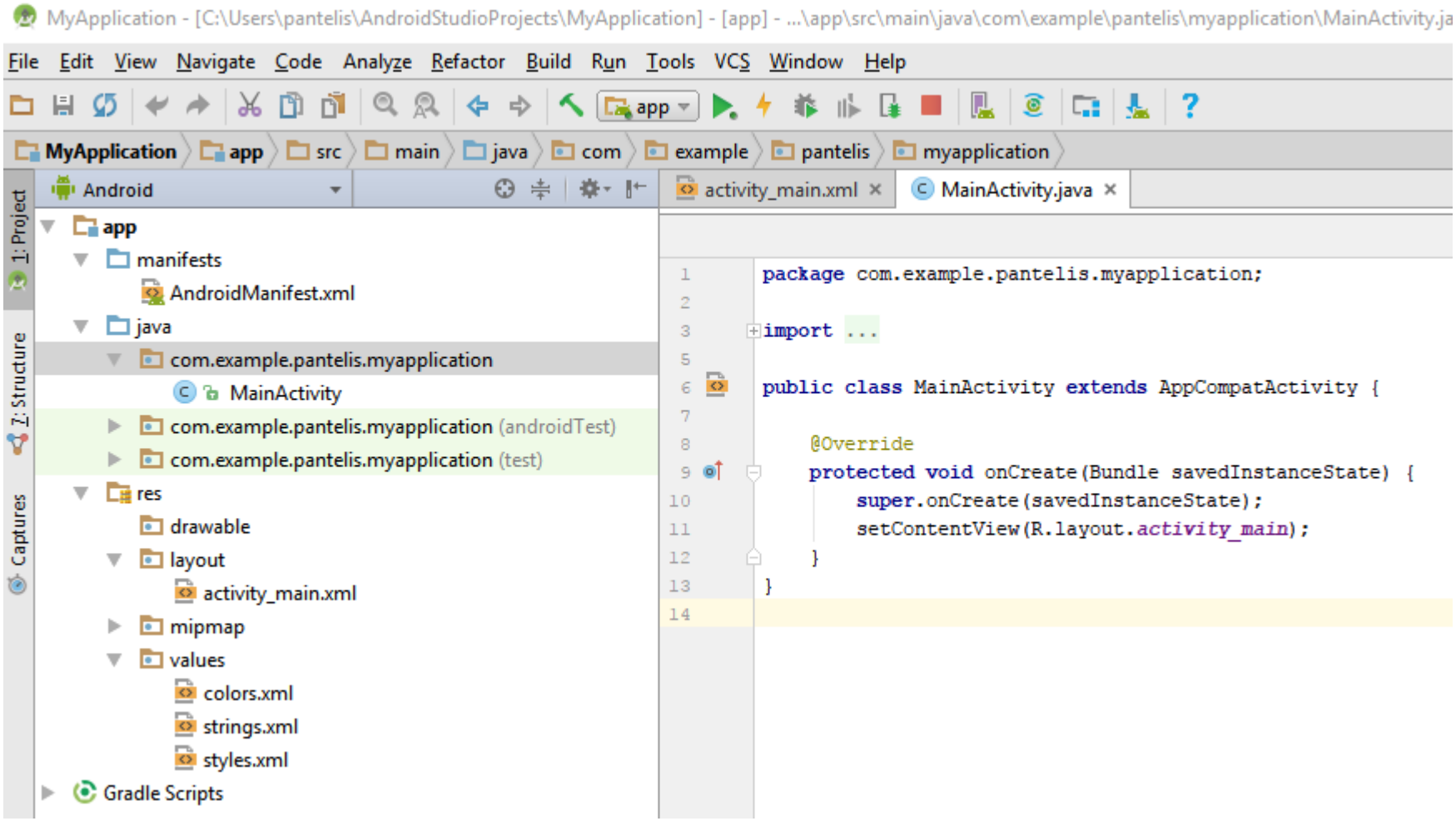
```
package gr.uoa.di.helloandroid;

import android.os.Bundle;
import android.app.Activity;

public class MainActivity extends Activity {

    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
    }

}
```

Τρέχει ότι ορίζεται στην onCreate() της γονικής Activity

Κώδικας για επιπλέον λειτουργικότητα από της γονικής onCreate()

UI Layout activity_main.xml

# MainActivity



# Απλό UI Layout XML

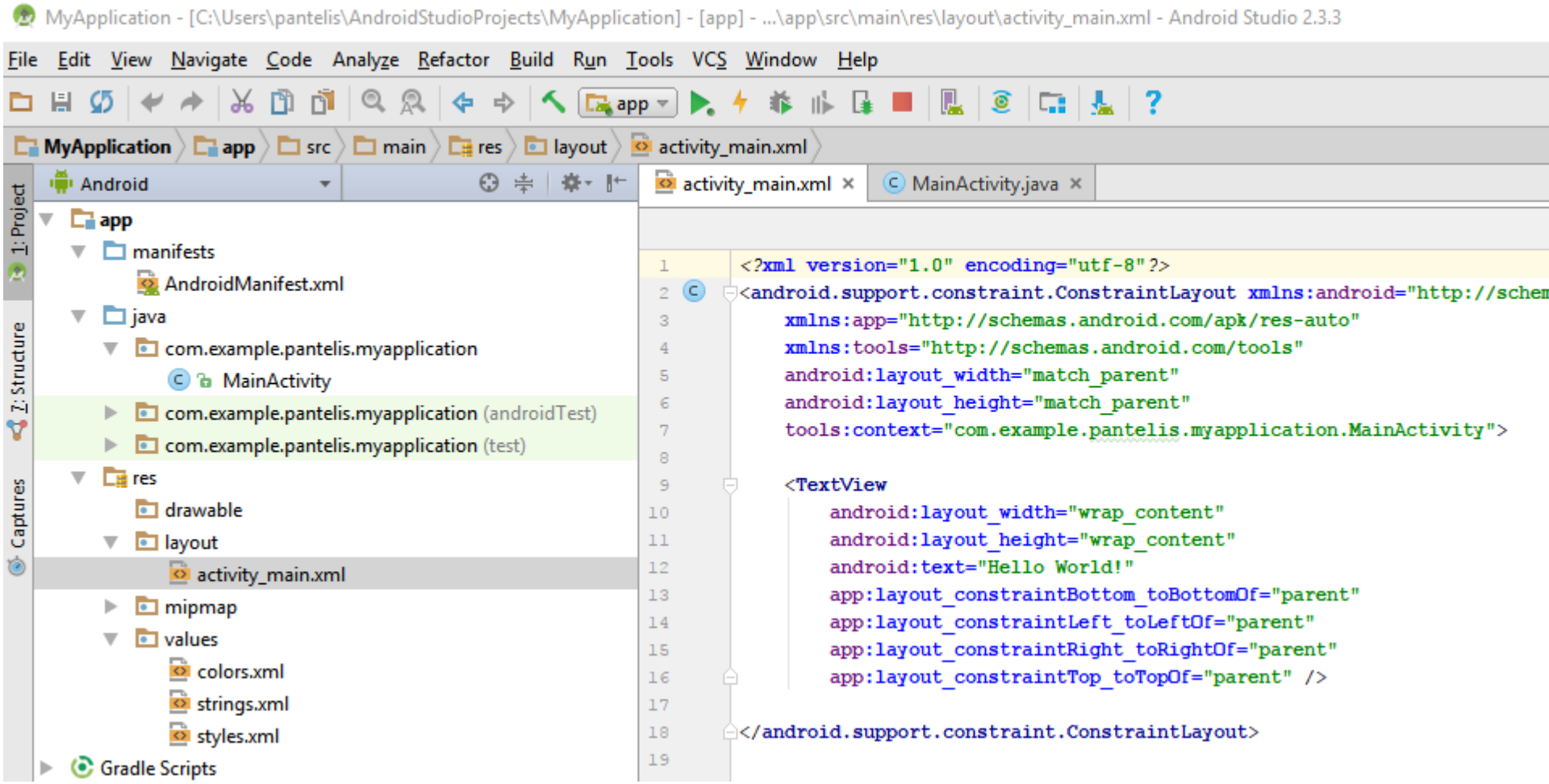
/res/layout/activity_main.xml

```
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent" >

    <TextView
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_centerHorizontal="true"
        android:layout_centerVertical="true"
        android:text="@string/hello_world"
        tools:context=".MainActivity" />

</RelativeLayout>
```

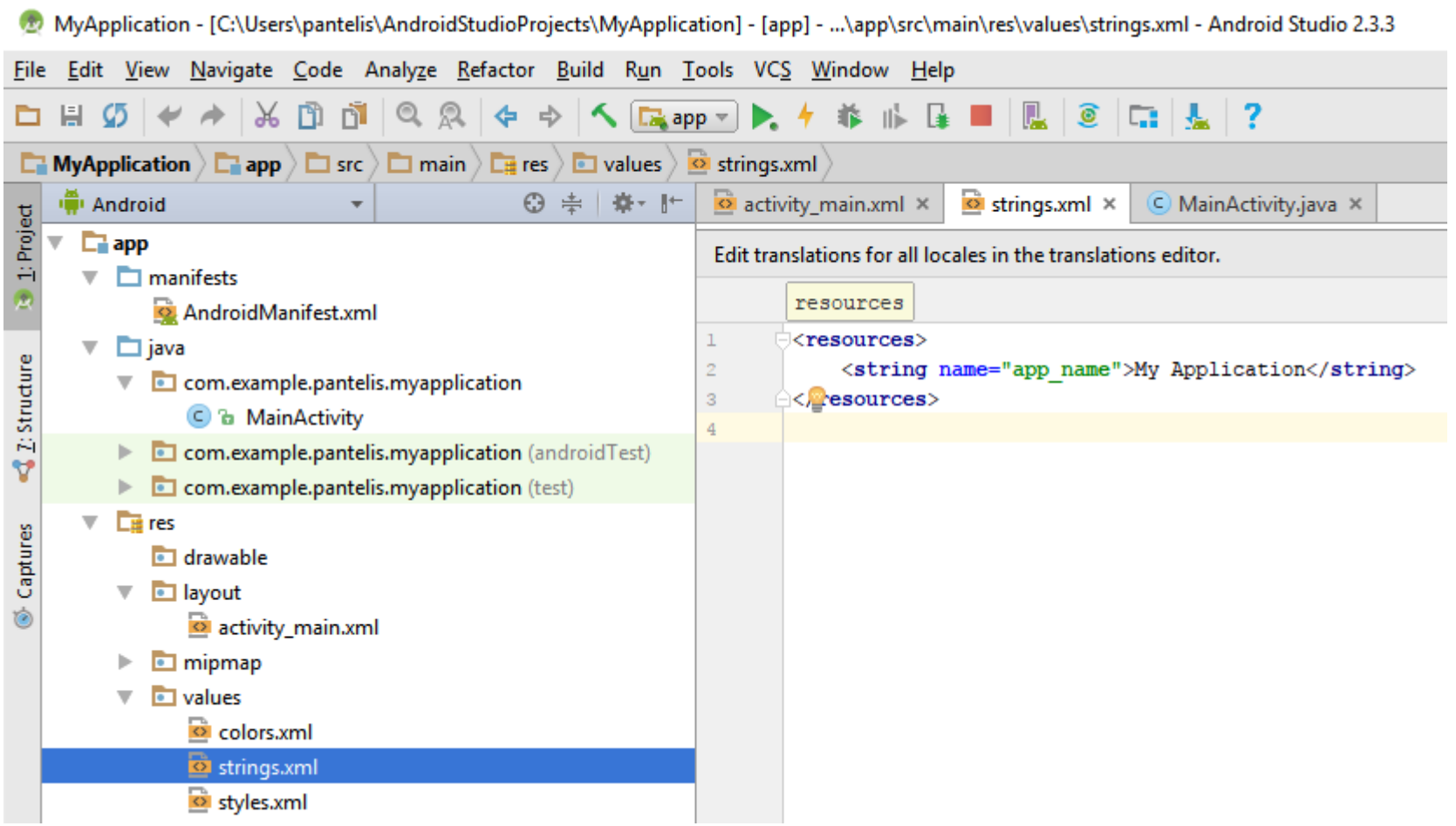
TextView Component

String hello_word

# Layout xml file

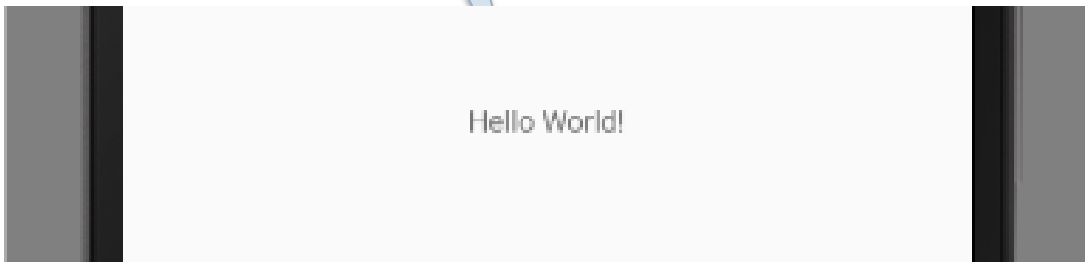


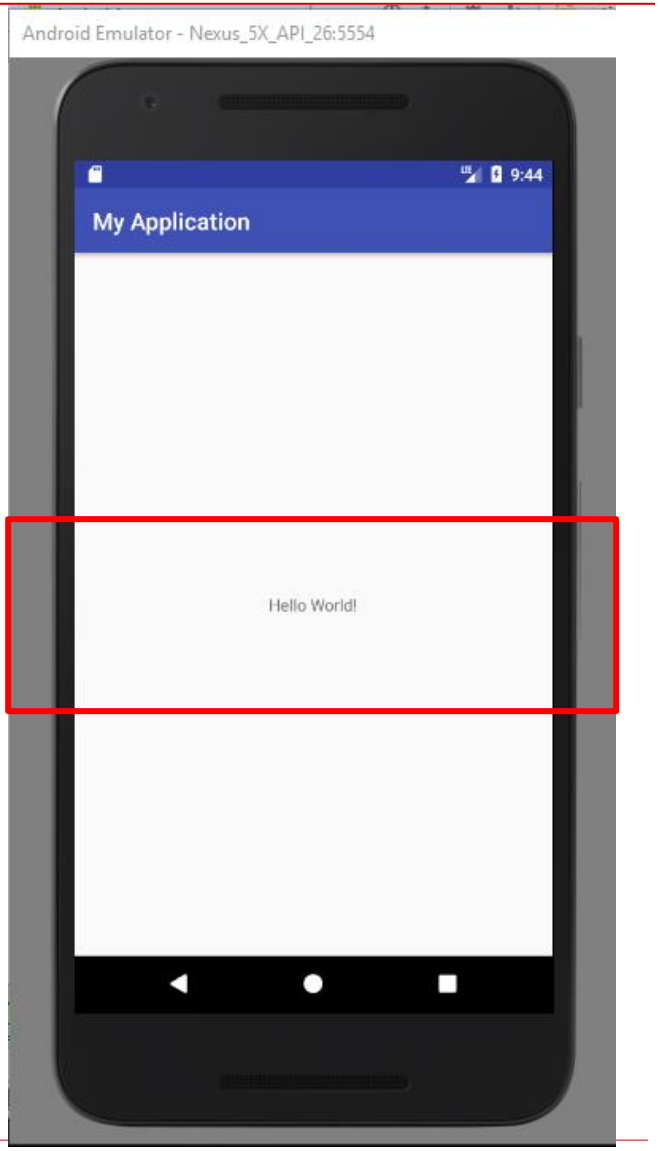
# strings.xml


MyApplication - [C:\Users\pantelis\AndroidStudioProjects\MyApplication] - [app] - ...\app\src\main\res\values\strings.xml - Android Studio 2.3.3
File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help
app
MyApplication app src main res values strings.xml
Android
app
manifests
AndroidManifest.xml
java
com.example.pantelis.myapplication
MainActivity
com.example.pantelis.myapplication (androidTest)
com.example.pantelis.myapplication (test)
res
drawable
layout
activity_main.xml
mipmap
values
colors.xml
strings.xml
styles.xml
activity_main.xml
strings.xml
MainActivity.java
Edit translations for all locales in the translations editor.
resources
<resources>
<string name="app_name">My Application</string>
</resources>
1: Project
7: Structure
Captures

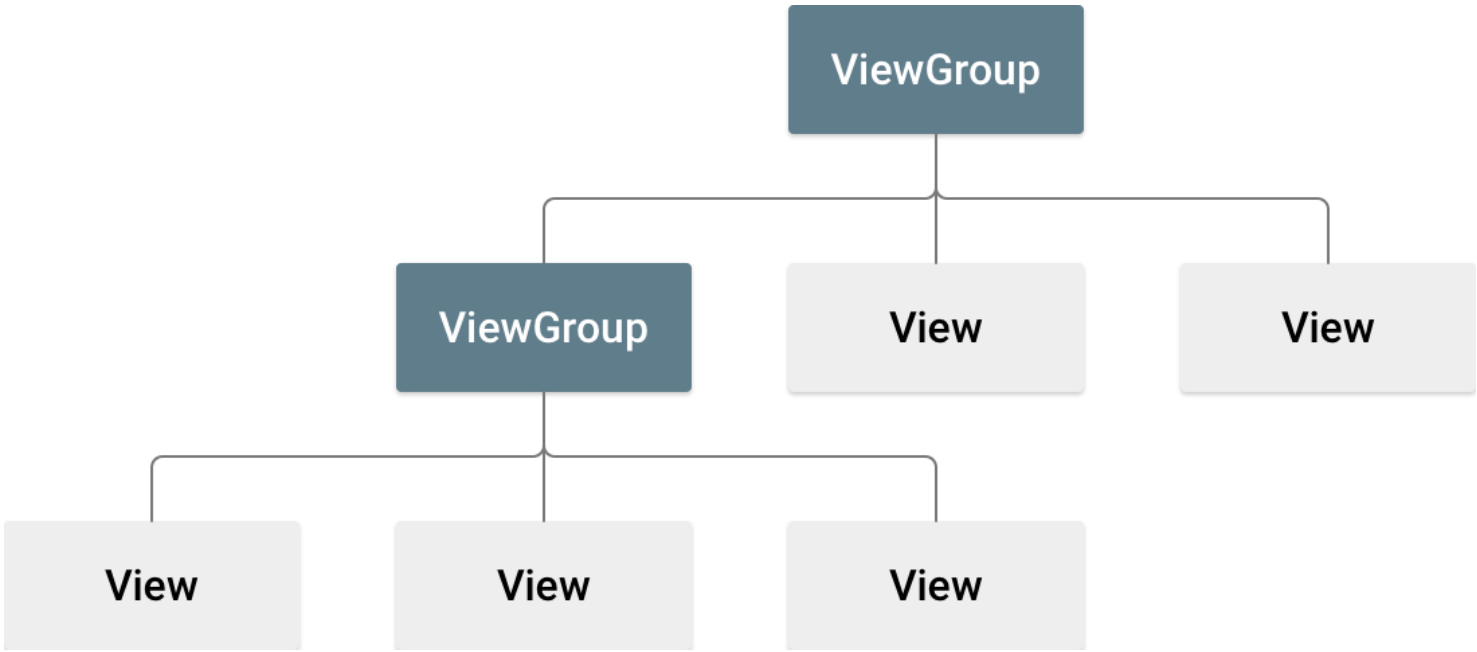
# Emulator

TextView Component



# User Interface Layout

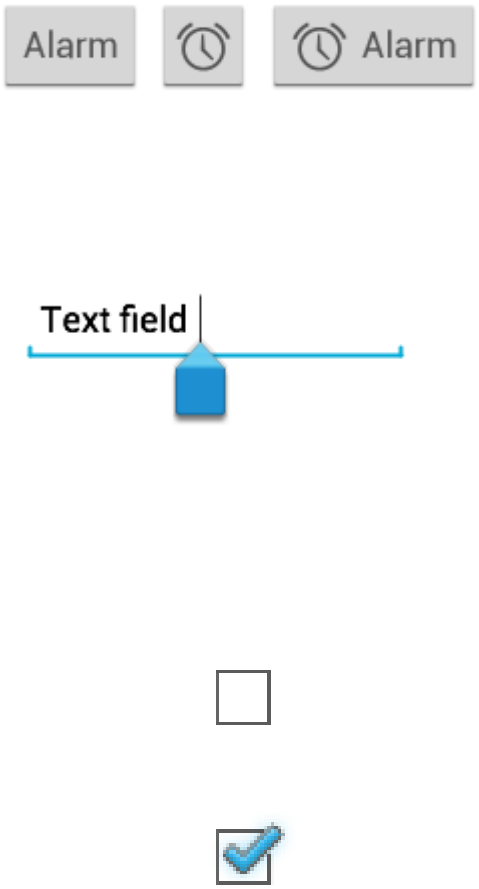



- Ότι εμφανίζετε σε μία οθόνη είναι view
- Ένα view μπορεί να είναι ένα **input control** ή **widgets** που συνθέτει κάποιο τμήμα του UI.
- Τα views **ομαδοποιούνται σε ViewGroup**. Κάθε view group είναι ένα αόρατο container που οργανώνει views.
- Το UI Layout ορίζεται ως μία ιεραρχία από ViewGroups και View αντικείμενα

https://developer.android.com/guide/topics/ui/overview.html

# Input controls (1)

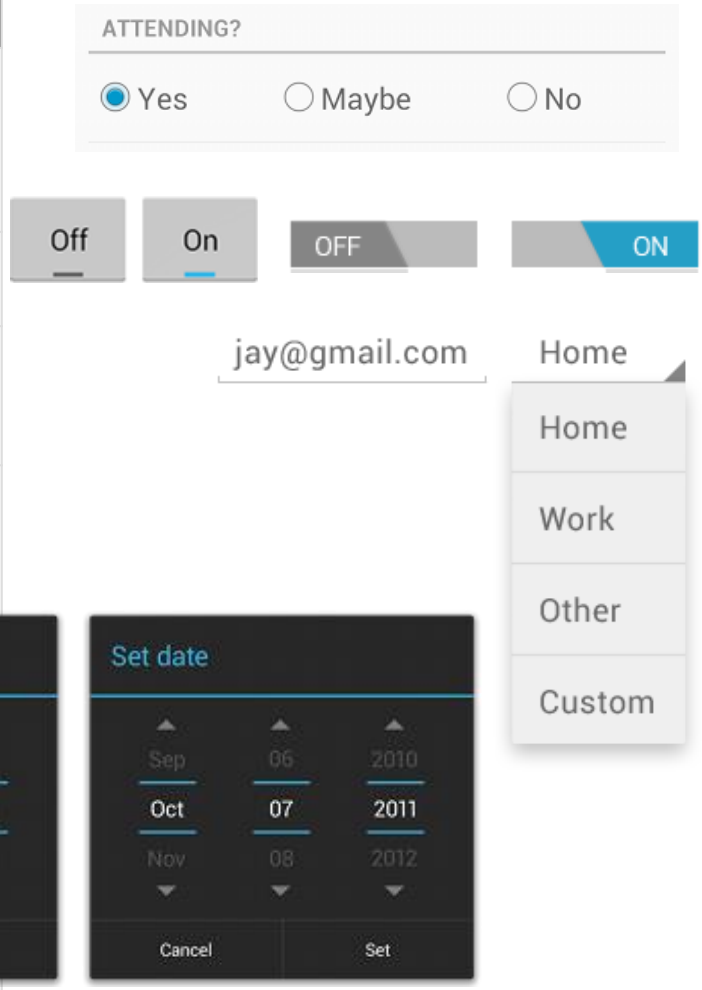
| Control Type | Description | Related Classes |
|---|---|---|
| Button | A **push-button** that can be **pressed**, or **clicked**, by the user to perform an action. | Button |
| Text field | An **editable text field**. You can use the AutoCompleteTextView widget to create a text entry widget that provides auto-complete suggestions | EditText, AutoCompleteTextView |
| Checkbox | An **on/off switch** that can be toggled by the user. You should use checkboxes when presenting users with a group of selectable options that are **not mutually exclusive.** | CheckBox |




https://developer.android.com/guide/topics/ui/controls.html

# Input controls (2)

| Control Type | Description | Related Classes |
|---|---|---|
| Radio button | Similar to checkboxes, except that only **one option can be selected in the group**. | RadioGroup<br>RadioButton |
| Toggle button | An on/off button with a light indicator. | ToggleButton |
| Spinner | A drop-down list that allows users to select one value from a set. | Spinner |
| Pickers | A dialog for users to select a single value for a set by using up/down buttons or via a swipe gesture. Use a DatePickercode> widget to enter the values for the date (month, day, year) or a TimePicker widget to enter the values for a time (hour, minute, AM/PM), which will be formatted automatically for the user's locale. | DatePicker, TimePicker |




https://developer.android.com/guide/topics/ui/controls.html

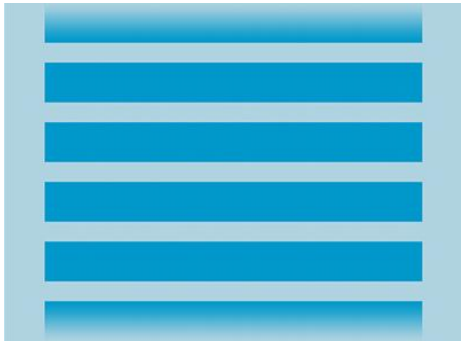
# Layouts: View Groups (1)

**Linear Layout:** Οργανώνει τα views σε μονές
- οριζόντιες ή
- κάθετες γραμμές

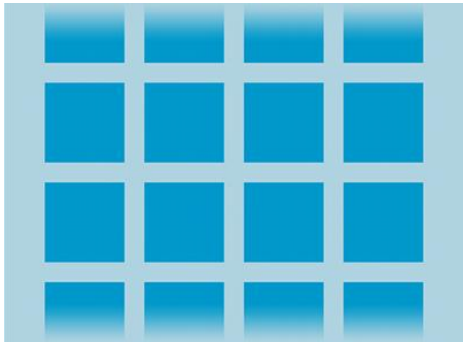
**Relative Layout:** Οργανώνει τα child-views σε σημεία που σχετίζονται με άλλα στοιχεία και σχετικά με άλλα child-views (child A αριστερά από το child B) ή στη πάνω περιοχή του γονέα (parent)


**Web View:** Απεικονίζει web σελίδες

# Layouts: View Groups (2)



**List View :** Απεικονίζει μία **κυλιόμενη** λίστα μονών γραμμών

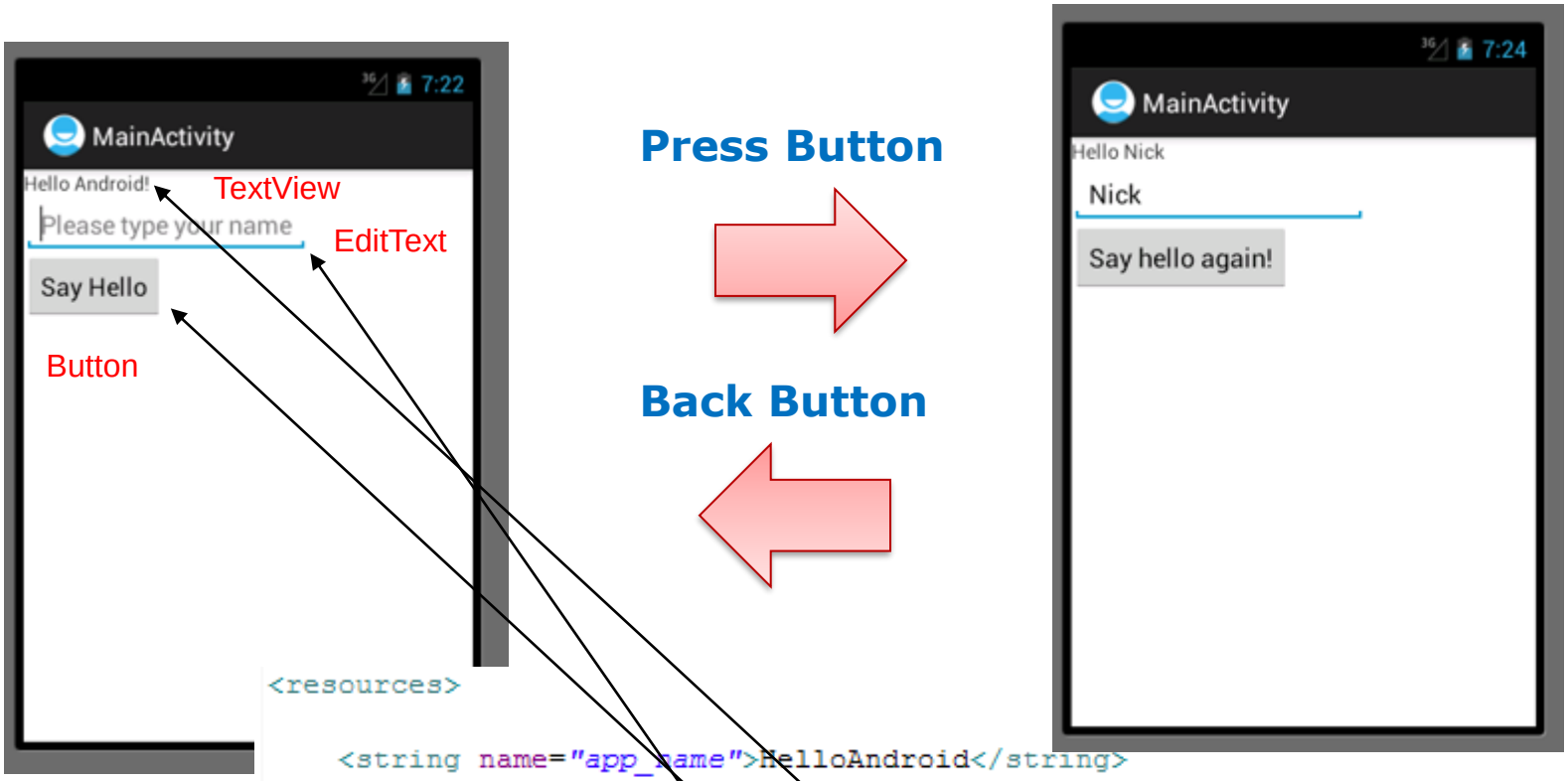


**Grid View:** Απεικονίζει ένα δισδιάστατο **κυλιόμενο** πλέγμα

# Inputs control @ layout xml file

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:orientation="horizontal">
    <EditText android:id="@+id/edit_message"
        android:layout_weight="1"
        android:layout_width="0dp"
        android:layout_height="wrap_content"
        android:hint="@string/edit_message" />
    <Button android:id="@+id/button_send"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="@string/button_send"
        android:onClick="sendMessage" />
</LinearLayout>
```

# Παράδειγμα με 3 Control Inputs




```
<resources>

    <string name="app_name">HelloAndroid</string>
    <string name="hello_android">Hello Android!</string>
    <string name="menu_settings">Settings</string>
    <string name="title_activity_main">MainActivity</string>
    <string name="button_hello">Say Hello</string>
    <string name="edit_name">Please type your name</string>

</resources>
```

# UI Layout XML (Παράδειγμα)

```
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical">
    <TextView android:id="@+id/hello_message"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="@string/hello_android"
        tools:context=".MainActivity" />
    <EditText android:id="@+id/edit_name"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:hint="@string/edit_name" />
    <Button
            android:id="@+id/button_hello"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:text="@string/button_hello"
            android:onClick="sayHello" />
</LinearLayout>
```

TextView Component

EditText Component

Κουμπί button_hello

String button_hello

Όνομα μεθόδου που θα κληθεί όταν πατηθεί το κουμπί

# Κώδικας

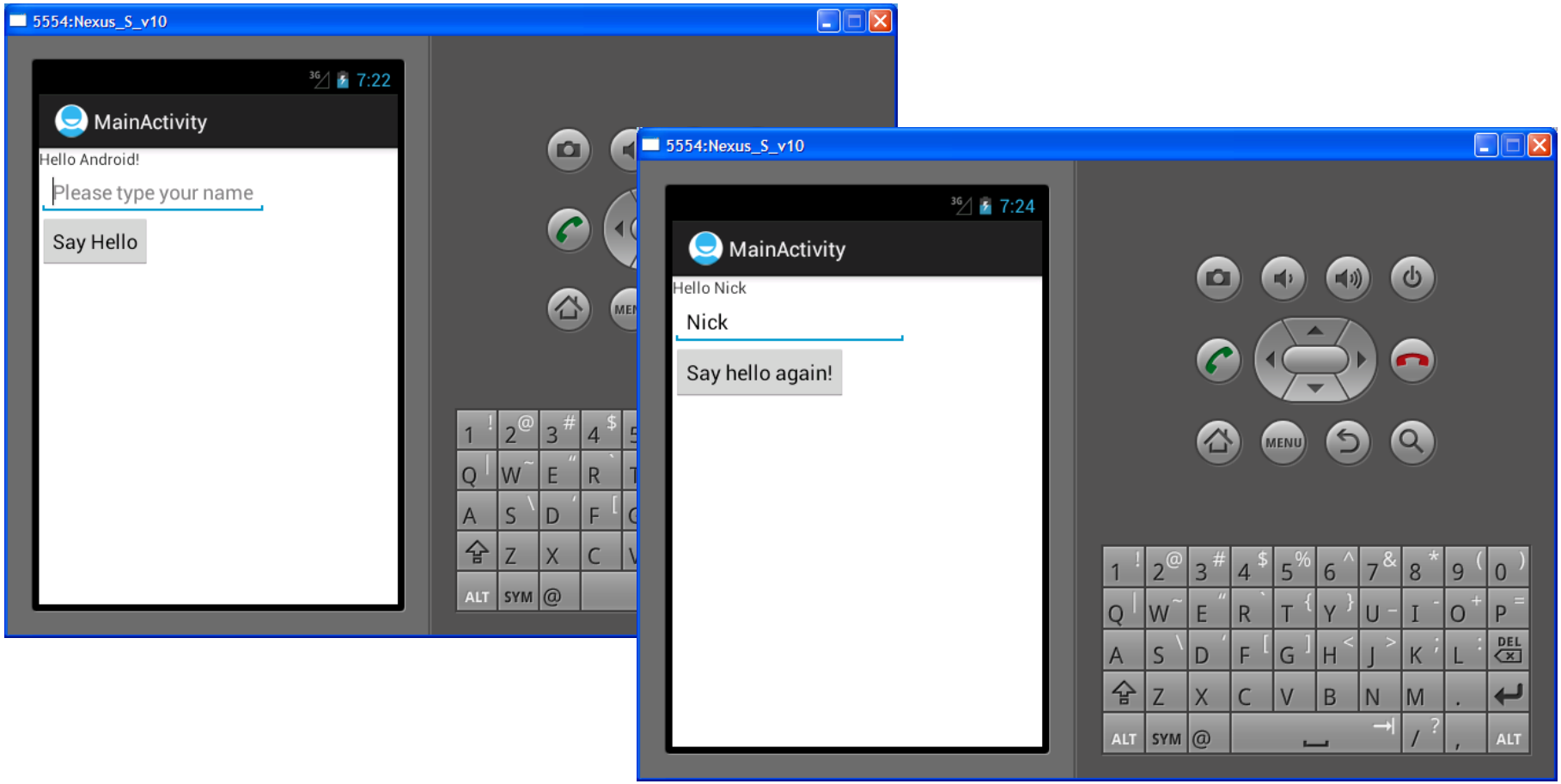
```
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.TextView;
import android.widget.EditText;
import android.app.Activity;

public class MainActivity extends Activity {

    private Button helloButton;
    private EditText editTextName;
    private TextView textViewHello;

    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        helloButton = (Button) findViewById(R.id.button_hello);
        editTextName = (EditText) findViewById(R.id.edit_name);
        textViewHello = (TextView) findViewById(R.id.hello_message);
    }

    public void sayHello(View view) {
        textViewHello.setText("Hello " + editTextName.getText());
        helloButton.setText("Say hello again!");
    }
}
```

UI Layout activity_main.xml

References to components of activity_main.xml

Μέθοδος που καλείται όταν πατηθεί το κουμπί

# Εκτέλεση στον Emulator (Παράδειγμα)



# Άλλες αναφορές για το σχεδιασμό UI

- https://developer.android.com/design/index.html
- https://developer.android.com/training/basics/firstapp/building-ui.html
- https://developer.android.com/guide/topics/ui
  - https://developer.android.com/guide/topics/ui/menus

Πως χρησιμοποιούμε το μηχανισμό Intent

# ΠΑΡΑΔΕΙΓΜΑ ΜΕ 2 ACTIVITIES

# Η κλάση Intent (1)

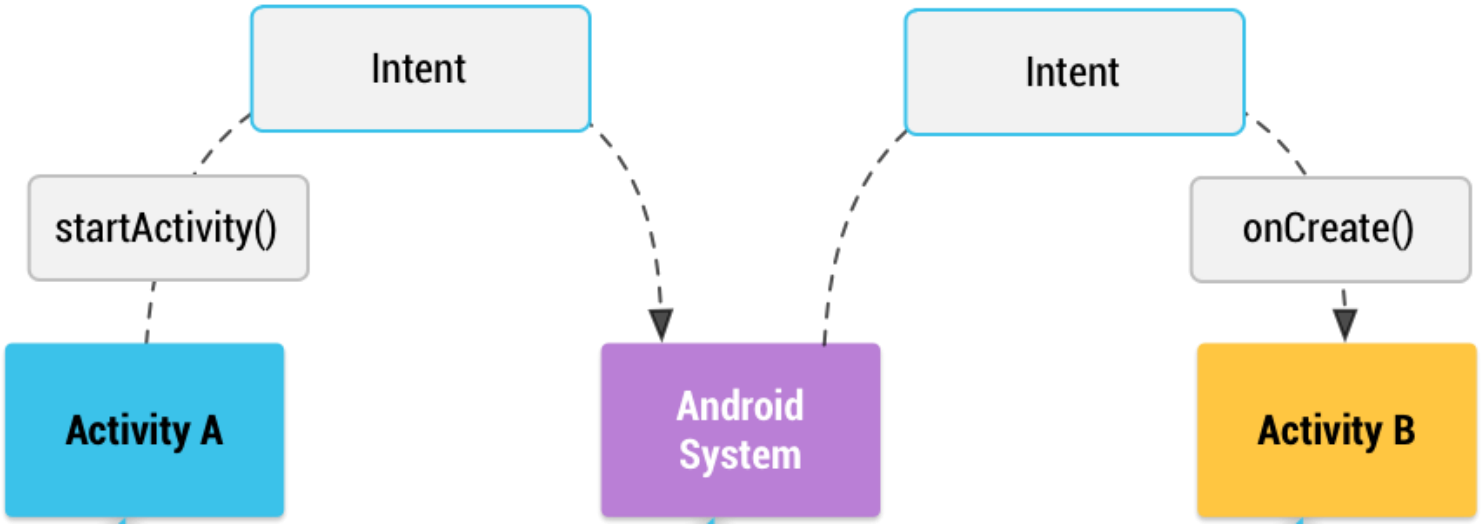
- Ο ρόλος ενός intent είναι **να ειδοποιηθεί ο Activity Manager** ότι ο χρήστης επιθυμεί να γίνει κάποια ενέργεια.
- Ένα intent είναι ένα **αντικείμενο της κλάσης Intent** το οποίο περιέχει κάποιο περιεχόμενο (το μήνυμα του intent)
- Μέσω των intents μπορούμε να ξεκινήσουμε την εκτέλεση
  - των Activities,
  - των Services και
  - των Broadcast Receivers

# Η κλάση Intent (2)

- Ένα **service** μπορεί να κληθεί με χρήση της
  - `Context.startService(Intent service)`
- Ένα **activity** μπορεί να κληθεί με χρήση των
  - `Context.startActivity(Intent intent)`
  - `Activity.startActivityForResult(Intent intent, int RequestCode)`
- Μια εφαρμογή μπορεί να δημιουργήσει ένα **broadcast μήνυμα** με χρήση του Intent
  - `Context.sendBroadcast(Intent intent),`
  - `Context.sendOrderedBroadcast(Intent intent, String receiverPermission)`
  - `Context.sendStickyBroadcast(Intent intent)`

https://developer.android.com/guide/components/intents-filters

# Intent activities




1. Η activity A "MainActivity" δημιουργεί ένα Intent μήνυμα με την περιγραφή της **ενέργειας – κλήση της Activity B** - και την στέλνει στο Android σύστημα με τη startActivity()
2. Το Android σύστημα καλεί την onCreate() της activity's B "HelloActivity" και περνά το Intent μήνυμα
3. Η "HelloActivity" έρχεται στο προσκήνιο κ.τ.λ.

# Κώδικας

```
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.TextView;
import android.widget.EditText;
import android.app.Activity;

public class MainActivity extends Activity {

+       private Button helloButton;
+       private EditText editTextName;
+       private TextView textViewHello;

        @Override
        public void onCreate(Bundle savedInstanceState) {
            super.onCreate(savedInstanceState);
            setContentView(R.layout.activity_main);

+           helloButton = (Button) findViewById(R.id.button_hello);
+           editTextName = (EditText) findViewById(R.id.edit_name);
+           textViewHello = (TextView) findViewById(R.id.hello_message);
        }

+       public void sayHello(View view) {
+           textViewHello.setText("Hello " + editTextName.getText());
+           helloButton.setText("Say hello again!");
        }
}
```

UI Layout activity_main.xml

References to components of activity_main.xml

Μέθοδος που καλείται όταν πατηθεί το κουμπί

# Παράδειγμα με 2 Activities

Χρήση Intent για έναρξη νέας activity

## 1η Activity: MainActivity

```
public void sayHello(View view) {

    Intent intent = new Intent(this, HelloActivity.class);
    String name = editTextName.getText().toString();
    intent.putExtra("name", name);
    startActivity(intent);
}
```

Δημιουργία μηνύματος Intent με την νέα activity

Προσθήκη δεδομένων στο Intent

Activity Manager

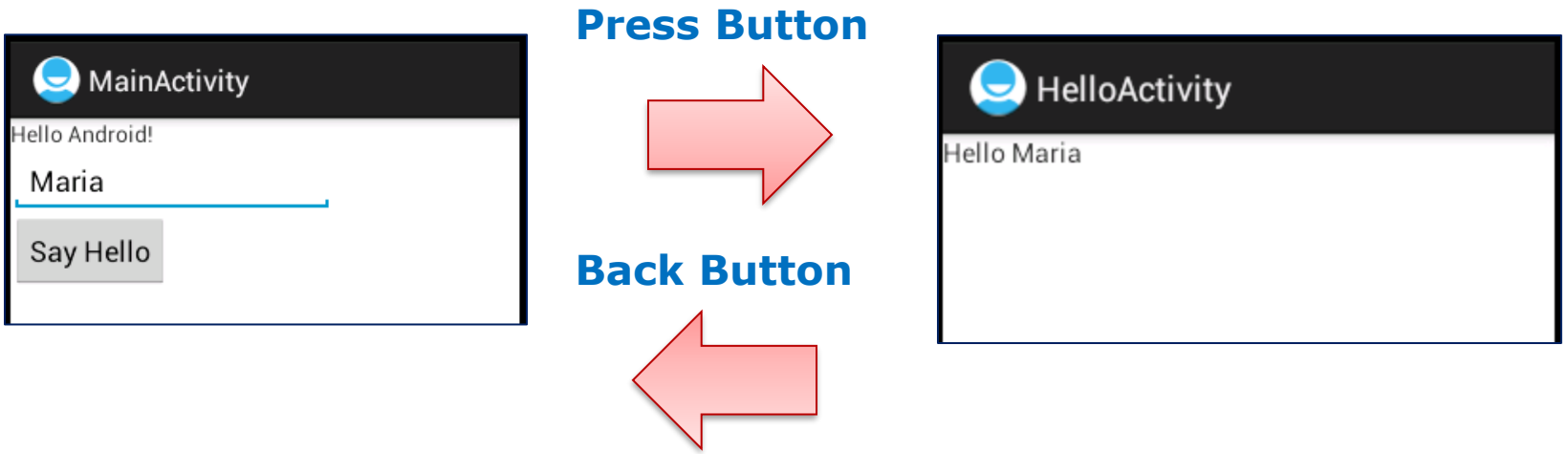
## 2η Activity: HelloActivity

```
@Override
public void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    Intent intent = getIntent();
    String name = intent.getStringExtra("name");
    TextView helloTextView = new TextView(this);
    helloTextView.setText("Hello " + name);
    setContentView(helloTextView);
}
```

Προσπέλαση μηνύματος Intent

Προσπέλαση δεδομένων από το Intent

# Εκτέλεση στον Emulator (Intent)



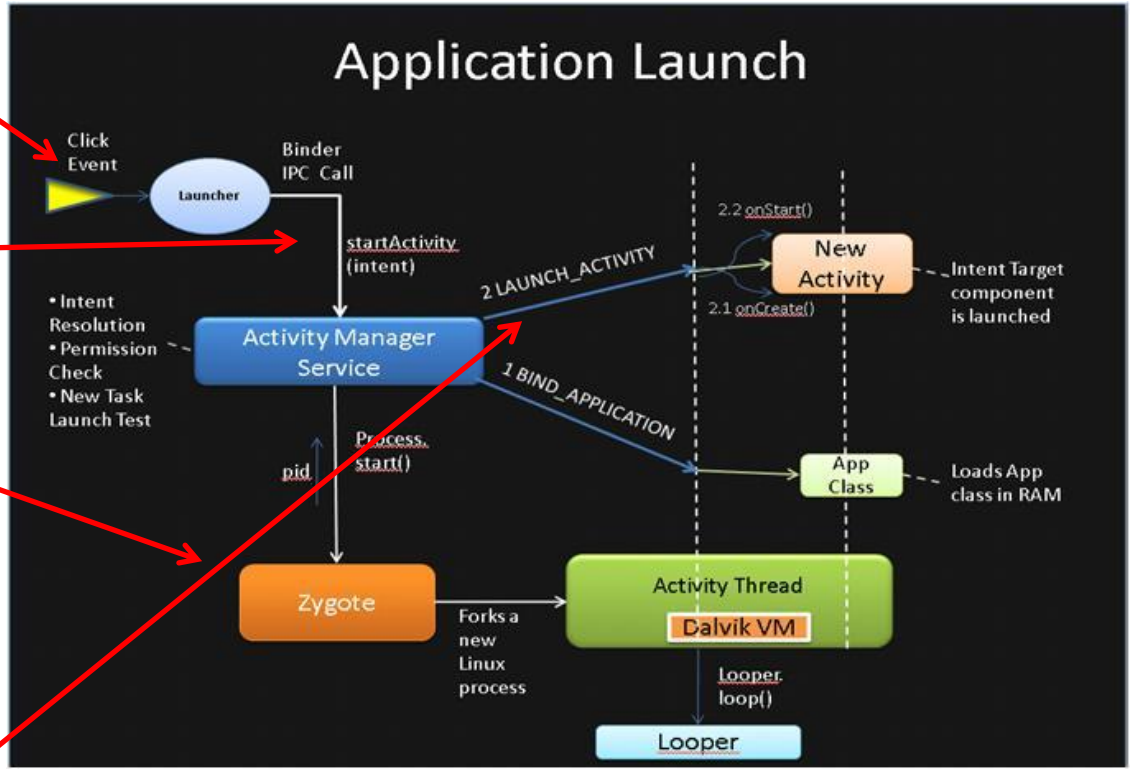
# Παράδειγμα Intent – Εκκίνηση Εφαρμογής[1]

Πάτημα κουμπιού εφαρμογής

Ειδοποίηση του ActivityManager σχετικά με την ενέργεια που επιθυμεί ο χρήστης

Έλεγχος και δημιουργία Διεργασίας για την εφαρμογή που εκκίνησε ο χρήστης

Δημιουργία του Activity που έχει δηλωθεί ως αρχικό στο Android Manifest xml αρχείο της εφαρμογής




1 http://coltf.blogspot.gr/p/android-os-processes-and-zygote.html

# ΆΛΛΕΣ ΣΗΜΑΝΤΙΚΕΣ ΚΛΑΣΕΙΣ

# Η κλάση Pending Intent

- Η κλάση **Pending Intent** δίνει τη δυνατότητα εκτέλεσης (επιτρέπει, παρέχει τη δικαιοδοσία) μίας λειτουργίας από άλλη εφαρμογή όπως
  - NotificationManager
  - AlarmManager
  - HomeScreen AppWidgetManager
- Παρέχοντας ένα PendingIntent σε μια άλλη εφαρμογή Β, της παρέχεις το δικαίωμα να **εκτελέσει τη λειτουργία** που προδιαγράφεις με τα **δικαιώματα και ταυτότητα της δικής σου εφαρμογής Α**
- Αντιθέτα, στο Intent όπου η λειτουργία θα **εκτελεστεί με τα δικαιώματα της άλλης εφαρμογής Β**

https://developer.android.com/reference/android/app/PendingIntent

# Η κλάση BroadcastReceiver

- Λαμβάνει ανακοινώσεις broadcast και αντιδρά κατάλληλα
  - **extends BroadcastReceiver**
  - implements **onReceive():** καλείται όταν ο BroadcastReceiver λαμβάνει ένα **Intent broadcast**
- Οι ανακοινώσεις broadcast μπορεί να παράγονται:
  - από το λειτουργικό (OS-generated)
    - π.χ. χαμηλή στάθμη μπαταρίας, αποκατάσταση σύνδεσης Wifi, πάτημα του πλήκτρου της φωτογραφικής
  - από το χρήστη (User-generated)
    - π.χ. έναρξη ή τερματισμός μια διαδικασίας, ενεργοποίηση ενός χαρακτηριστικού

https://developer.android.com/guide/components/broadcasts.html

# System Events

| Event | Description |
| --- | --- |
| Intent.ACTION_BOOT_COMPLETED | Το boot έγινε με επιτυχία. Απαιτείται η άδεια, android.permission.RECEIVE_BOOT_COMPLETED. |
| Intent.ACTION_POWER_CONNECTED | Έγινε σύνδεση του φορτιστή στην συσκευή. |
| Intent.ACTION_POWER_DISCONNECTED | Έγινε αποσύνδεση του φορτιστή από την συσκευή. |
| Intent.ACTION_BATTERY_LOW | Η στάθμη της μπαταρίας είναι χαμηλή. Χρησιμοποιείται για την εξοικονόμηση ενέργειας από εφαρμογές που καταναλώνουν πολύ. |
| Intent.ACTION_BATTERY_OKAY | Η στάθμη της μπαταρίας είναι σε καλό επίπεδο. |

# BroadcastReceiver – Παράδειγμα

Παράδειγμα χρήσης BroadcastReceiver για ανάκτηση πληροφοριών σχετικά με το Wifi

## Manifest File

```
<uses-permission android:name="android.permission.ACCESS_WIFI_STATE"></uses-permission>
<uses-permission android:name="android.permission.CHANGE_WIFI_STATE"></uses-permission>
```

# BroadcastReceiver – Παράδειγμα

## WifiReceiver

```
package gr.uoa.di.android.monitoring.application;

import android.content.BroadcastReceiver;
import android.content.Context;
import android.content.Intent;
import android.net.wifi.WifiInfo;
import android.net.wifi.WifiManager;
import android.os.Bundle;

public class WifiReceiver extends BroadcastReceiver{

    @Override
    public void onReceive(Context context, Intent intent) {

        Bundle b = intent.getExtras();
        WifiInfo info = (WifiInfo) b.get(WifiManager.EXTRA_WIFI_INFO);

        String ssid = info.getSSID();
        String bssid = info.getBSSID();
    }

}
```

# BroadcastReceiver – Παράδειγμα

## Receiver

```
WifiManager wifiManager = (WifiManager) getSystemService(Context.WIFI_SERVICE);

WifiReceiver wifiReceiver = new WifiReceiver();
registerReceiver(wifiReceiver, new IntentFilter(
        WifiManager.SCAN_RESULTS_AVAILABLE_ACTION));

wifiManager.startScan();
```

WifiReceiver extends BroadcastReceiver

Register a **BroadcastReceiver** to listen to SCAN_RESULTS_AVAILABLE_ACTION.
You can either dynamically register an instance of this class with `Context.registerReceiver()` or statically declare an implementation with the `<receiver>` tag in your `AndroidManifest.xml`.

https://developer.android.com/reference/android/net/wifi/WifiManager.html#startScan()

# Η κλάση Log

- Η εκτύπωση στη standard έξοδο (System.out.println) **ΔΕΝ δουλεύει στο Android**
- Χρήση της κλάσης **Log για debugging**
  - Ενεργοποίηση του LogCat (στο Android Studio): Window-> Show View -> Other -> Android-> Logcat
  - Η κλάση Log παρέχει αρκετές static μεθόδους για εκτύπωση μηνυμάτων διαφορετικής σημασίας για το λειτουργικό σύστημα:

    **Log.e( ):** Errors **Log.w( ):** Warnings

    **Log.i( ):** Information **Log.d( ):** Debugging

    **Log.v( ):** Verbose

http://developer.android.com/reference/android/util/Log.html

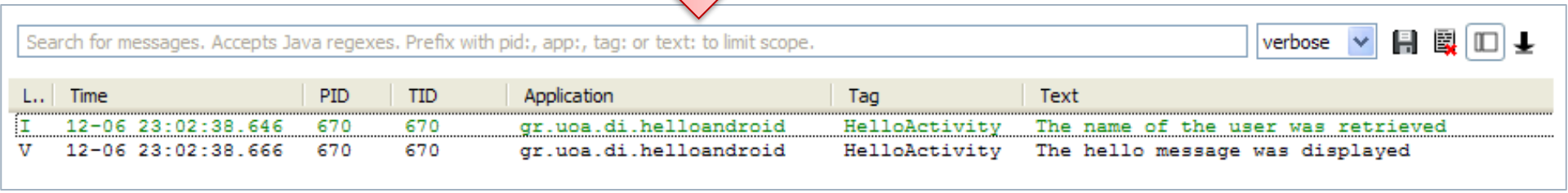
https://developer.android.com/studio/debug/index.html https://developer.android.com/studio/debug/am-logcat.html

# Η κλάση Log

## HelloActivity

```
private static final String TAG = "HelloActivity";

@Override
public void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    Intent intent = getIntent();
    String name = intent.getStringExtra("name");

+   Log.i(TAG, "The name of the user was retrieved");

    TextView helloTextView = new TextView(this);
    helloTextView.setText("Hello " + name);

+   Log.v(TAG, "The hello message was displayed");

    setContentView(helloTextView);
}
```

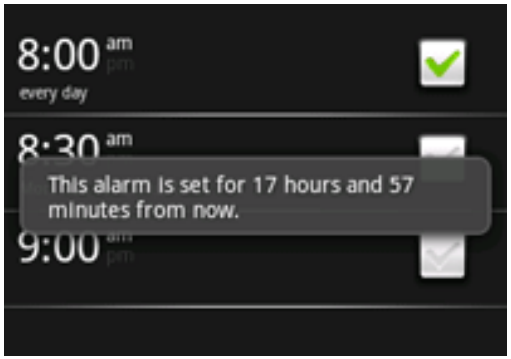
## LogCat Monitor

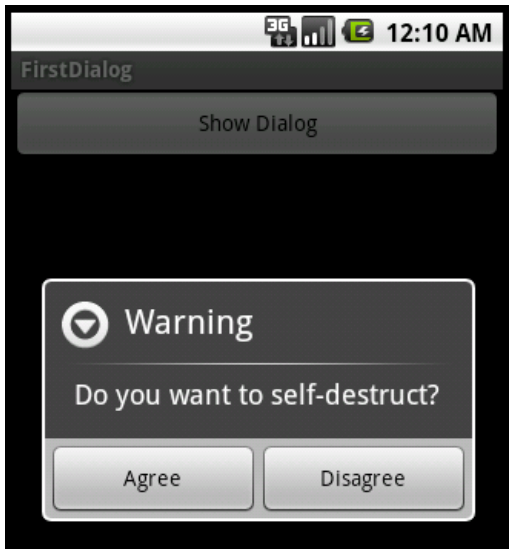


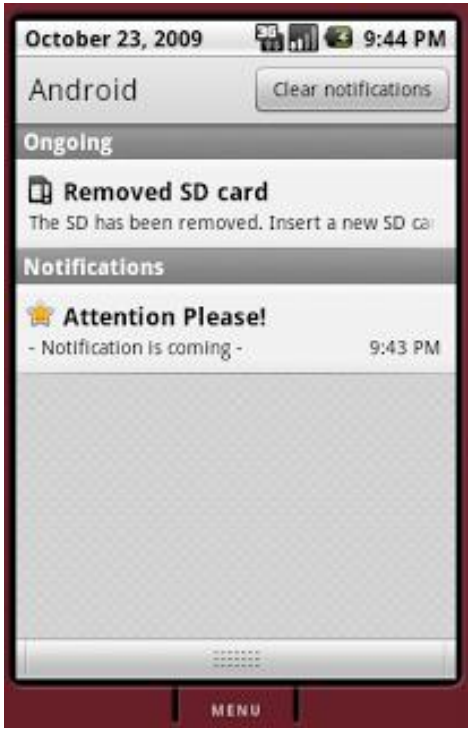
# Ειδοποιήσεις (Notifications) (1)

**Δημιουργία ειδοποιήσεων στο Android**

- Τριών ειδών ειδοποιήσεις
  1. Toast
  2. AlertDialog
  3. Notification



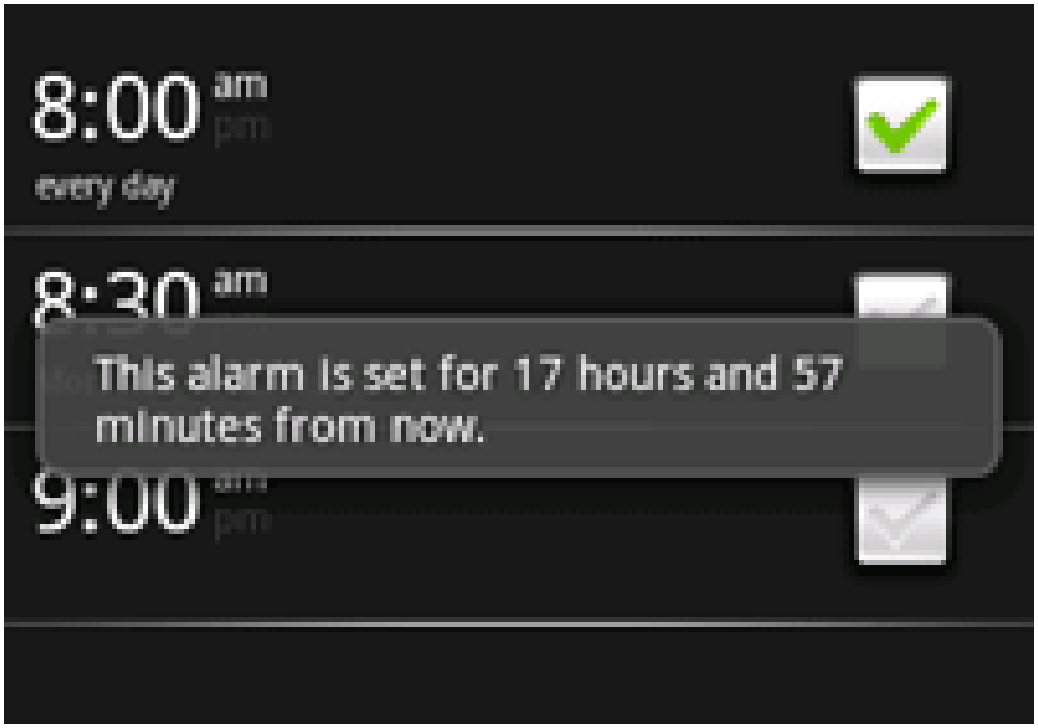



# Ειδοποιήσεις (Notifications) (2)

- **Toast**

```
Toast toast = Toast.makeText(this,”Notification",Toast.LENGTH_SHORT);
toast.show();
```

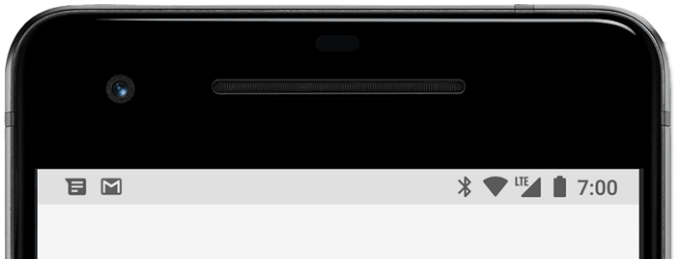


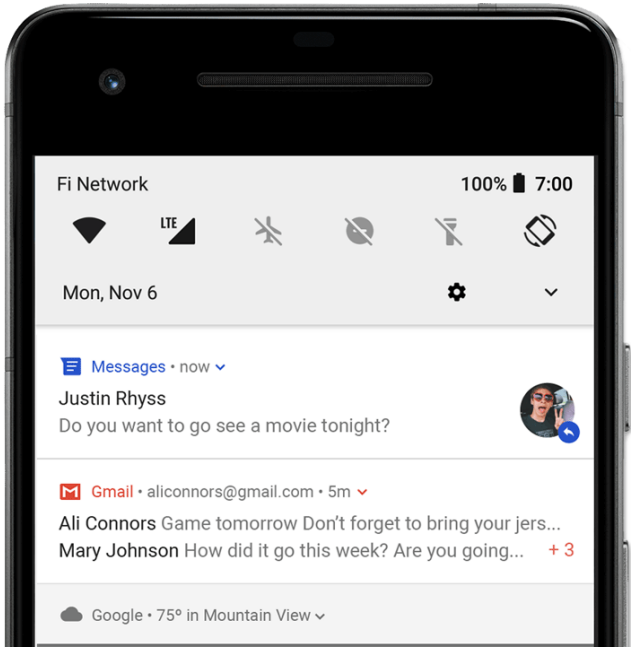

https://developer.android.com/guide/topics/ui/notifiers/toasts.html

# Ειδοποιήσεις (Notifications) (3)

- **Notification**

```
NotificationCompat.Builder mBuilder = new NotificationCompat.Builder(this,
CHANNEL_ID)
        .setSmallIcon(R.drawable.notification_icon)
        .setContentTitle(textTitle)
        .setContentText(textContent)
        .setPriority(NotificationCompat.PRIORITY_DEFAULT);
```

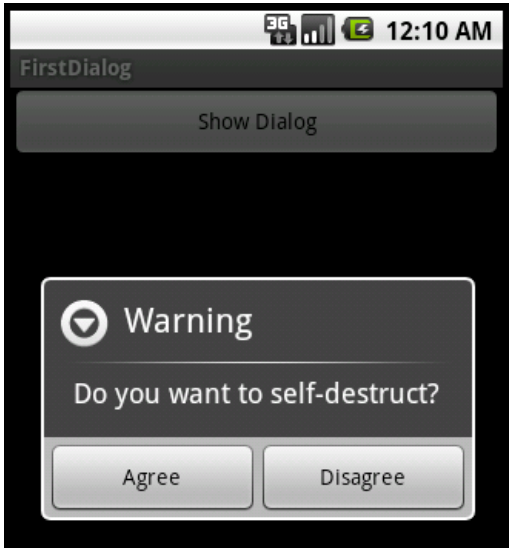



https://developer.android.com/guide/topics/ui/notifiers/notifications.html

# Ειδοποιήσεις (Notifications) (4)

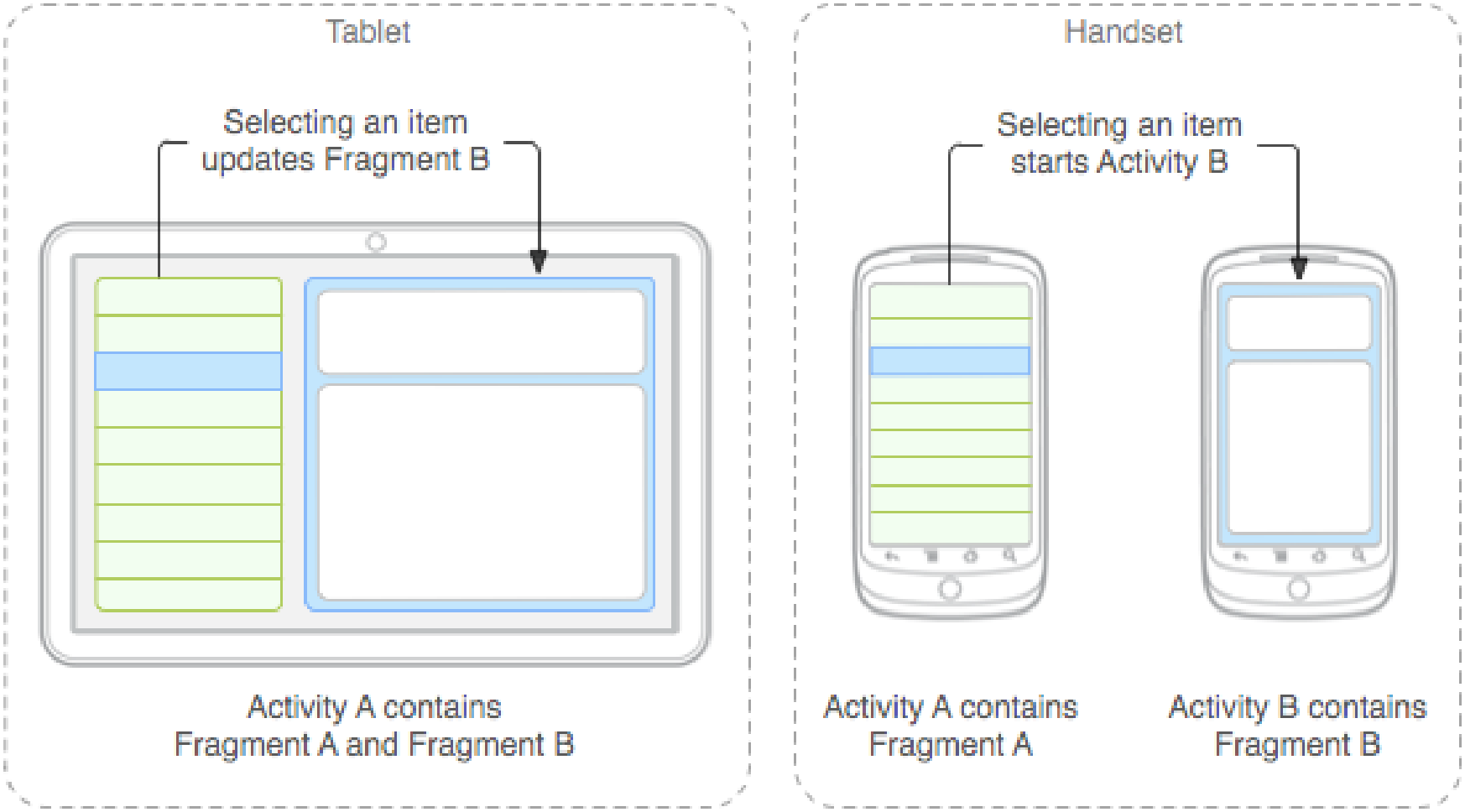
- **AlertDialog**




```
AlertDialog.Builder builder = new AlertDialog.Builder(getActivity());

    builder.setMessage("Do you want to self-destruct?")
           .setPositiveButton("Agree", new DialogInterface.OnClickListener() {
               public void onClick(DialogInterface dialog, int id) {
                   // Exit code
               }
           })
           .setNegativeButton("Disagree", new DialogInterface.OnClickListener() {
               public void onClick(DialogInterface dialog, int id) {
                   // Return code
               }
           });

    AlertDialog dialog = builder.create();
```

https://developer.android.com/guide/topics/ui/dialogs.html

# Fragments

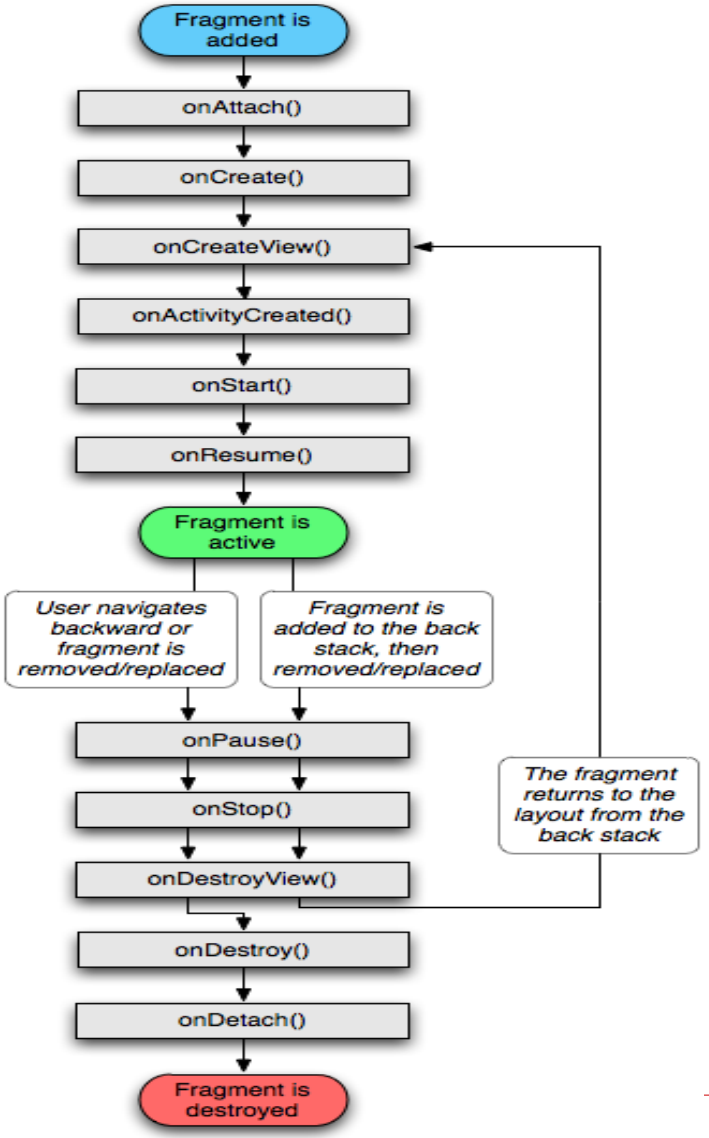

Tablet
Selecting an item
updates Fragment B
Activity A contains
Fragment A and Fragment B
Handset
Selecting an item
starts Activity B
Activity A contains
Fragment A
Activity B contains
Fragment B

# Fragments

- Ανεξάρτητο component το οποίο χρησιμοποιείται από activities.
  - Θεωρείται ως ένα είδος sub-activity
- Για τα fragments ορίζεται κύκλος ζωής αντίστοιχος του κύκλου ζωής ενός activity
- Δυναμικός και στατικός ορισμός (on application running)
- Μια κλάση ορίζεται ως Fragment
  - **extend Fragment, ListFragment, DialogFragment, PreferenceFragment**
  - **Override onCreateView()**
- Γενικά μας δίνει την δυνατότητα να επαναχρησιμοποιούμε components σε διαφορετικά layouts

http://developer.android.com/guide/components/fragments.html

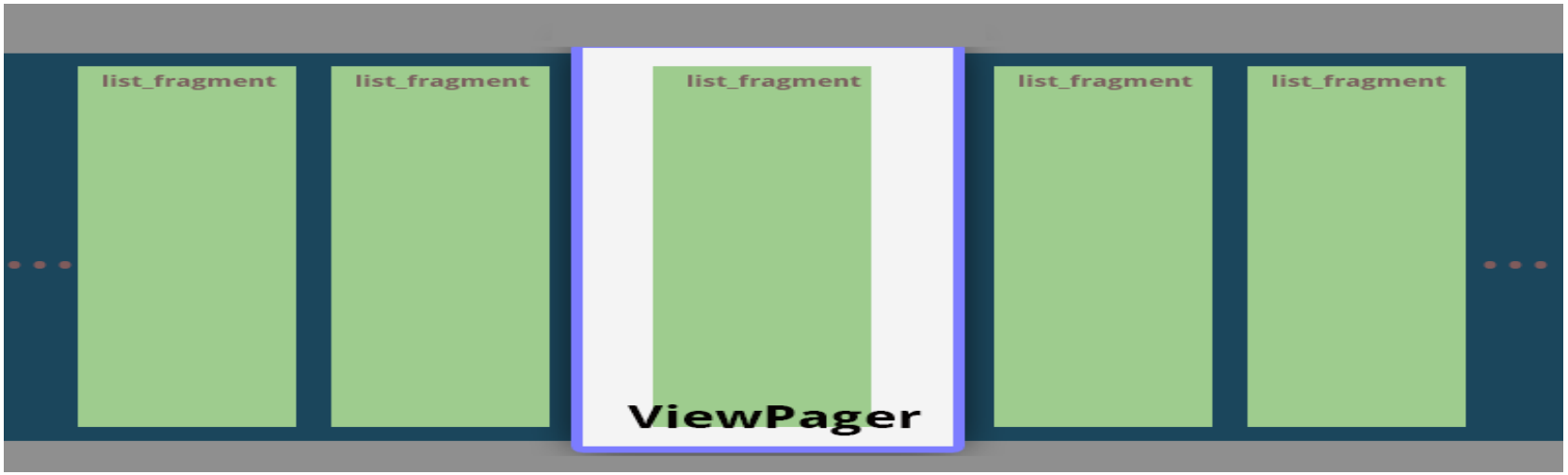
# Fragments




| Method | Description |
|---|---|
| onAttach() | Το instance του fragment συσχετίζεται με το instance του activity. |
| onCreate() | Δημιουργία του fragment |
| **onCreateView()** | Το instance του fragment δημιουργεί την ιεραρχία παρουσίασης (view hierarchy) του. Η ιεραρχία αυτή γίνεται μέλος της ιεραρχίας του activity στο όποιο έχει οριστεί το fragment. |
| onActivityCreated() | Το activity και το fragment έχουν δημιουργήσει τα instances τους με βάση τα view hierarchy τους. Πλέον υπάρχει πρόσβαση σε αυτά με την μέθοδο **findViewbyId().** |
| onResume() | Το fragment είναι πλέον ενεργό και ορατό στον χρήστη. |
| onPause() | Το fragment είναι ορατό αλλά όχι ενεργό, πχ κάποιο άλλο activity χρησιμοποιεί το συγκεκριμένο fragment. |
| onStop() | Το fragment πλέον δεν είναι ορατό στον χρήστη. |

# Κλάση ViewPager

- **Layout διαχειριστής**, επιτρέπει στον χρήστη να πλοηγείται «δεξιά - αριστερά» σε διαφορετικές οθόνες δεδομένων.
- Η χρήση του **ViewPager** προϋποθέτει τον ορισμό του **PageAdapter**.
  - Ο PageAdapter είναι υπεύθυνος για την δημιουργία των οθονών του ViewPager.
- Συνδυαστική χρήση με Fragment με σκοπό την βέλτιστη διαχείριση του κύκλου ζωής της εκάστοτε σελίδας.




https://developer.android.com/reference/android/support/v4/view/ViewPager

- Εισαγωγή σε Android
- Δομή αρχείων σε Android Projects
- Activity
- Intent
- Άλλες σημαντικές κλάσεις
- Sensors

# Αισθητήρες - Sensors

- **Τι είναι αισθητήρας;**
  - Συσκευή που ανιχνεύει και αποκρίνεται σε κάποιου είδους εξωτερικό ερέθισμα
- **Τύποι αισθητήρων**
  - Κίνησης (π.χ. επιταχυνσιόμετρο, γυροσκόπιο)
  - Περιβάλλοντος (π.χ. θερμοκρασίας, φωτισμού)
  - Τοποθεσίας (π.χ. προσανατολισμού, μαγνητόμετρο)

https://developer.android.com/guide/topics/sensors/sensors_overview https://source.android.com/devices/sensors

# Τύποι αισθητήρων

| Sensor | Type | Description | Common Uses |
|---|---|---|---|
| **TYPE_ACCELEROMETER** | Hardware | Measures the acceleration force in $m/s^2$ that is applied to a device on all three physical axes (x, y, and z), including the force of gravity. | **Motion detection (shake, tilt, etc.).** |
| TYPE_AMBIENT_TEMPERATURE | Hardware | Measures the ambient room temperature in degrees Celsius (°C). See note below. | **Monitoring air temperatures.** |
| TYPE_GRAVITY | Software or Hardware | Measures the force of gravity in $m/s^2$ that is applied to a device on all three physical axes (x, y, z). | **Motion detection (shake, tilt, etc.).** |
| TYPE_GYROSCOPE | Hardware | Measures a device's rate of rotation in rad/s around each of the three physical axes (x, y, and z). | **Rotation detection (spin, turn, etc.).** |
| TYPE_LIGHT | Hardware | Measures the ambient light level (illumination) in lx. | **Controlling screen brightness.** |
| TYPE_LINEAR_ACCELERATION | Software or Hardware | Measures the acceleration force in $m/s^2$ that is applied to a device on all three physical axes (x, y, and z), excluding the force of gravity. | **Monitoring acceleration along a single axis.** |
| TYPE_MAGNETIC_FIELD | Hardware | Measures the ambient geomagnetic field for all three physical axes (x, y, z) in μT. | **Creating a compass.** |
| TYPE_ORIENTATION | Software | Measures degrees of rotation that a device makes around all three physical axes (x, y, z). As of API level 3 you can obtain the inclination matrix and rotation matrix for a device by using the gravity sensor and the geomagnetic field sensor in conjunction with the getRotationMatrix() method. | **Determining device position.** |

| Sensor | Type | Description | Common Uses |
| --- | --- | --- | --- |
| TYPE_PRESSURE | Hardware | Measures the ambient air pressure in hPa or mbar. | **Monitoring air pressure changes.** |
| TYPE_PROXIMITY | Hardware | Measures the proximity of an object in cm relative to the view screen of a device. This sensor is typically used to determine whether a handset is being held up to a person's ear. | **Phone position during a call.** |
| TYPE_RELATIVE_HUMIDITY | Hardware | Measures the relative ambient humidity in percent (%). | **Monitoring dewpoint, absolute, and relative humidity.** |
| TYPE_ROTATION_VECTOR | Software or Hardware | Measures the orientation of a device by providing the three elements of the device's rotation vector. | **Motion detection and rotation detection.** |
| TYPE_TEMPERATURE | Hardware | Measures the temperature of the device in degrees Celsius (°C). This sensor implementation varies across devices and this sensor was replaced with the TYPE_AMBIENT_TEMPERATURE sensor in API Level 14 | **Monitoring temperatures.** |

# Android sensor framework (ASF)

- **Android sensor framework (ASF):** Μπορεί κανείς να έχει πρόσβαση στους διαθέσιμους αισθητήρες και να αποκτήσει raw δεδομένα χρησιμοποιώντας
- Το ASF παρέχει **κλάσεις και διεπαφές** που βοηθούν στην εκτέλεση εργασιών σχετικών με τους αισθητήρες
  - Προσδιορίζουν **ποιοι αισθητήρες είναι διαθέσιμοι** σε μια συσκευή.
  - Προσδιορίζουν τις **δυνατότητες μεμονωμένων αισθητήρων**, όπως μέγιστη εμβέλεια, κατασκευαστή, απαιτήσεις ισχύος και ανάλυση.
  - **Αποκτούν δεδομένα αισθητήρων** και καθορίζουν τον ελάχιστο ρυθμό απόκτησης των δεδομένων αισθητήρων.
  - **Καταχωρούν και καταργούν sensor event listeners** που παρακολουθούν τις αλλαγές αισθητήρα

# Android Sensors API

- **Sensor Manager**
  - System service; gives access to hardware sensors
- **Sensor**
  - Representation of a sensor in a device
- **SensorEventListener**
  - Interface providing callbacks
- **SensorEvent**
  - Data structure with event information

# Android Sensors API

## SensorManager class

- Μπορείτε να χρησιμοποιήσετε αυτήν την κλάση για να δημιουργήσετε ένα instance ενός **sensor service**. Αυτή η κλάση παρέχει διάφορες **μεθόδους** για
  - δημιουργία και πρόσβαση στη λίστα των διαθέσιμων αισθητήρων
  - την καταχώριση και την κατάργηση **sensor event listeners** και
  - την απόκτηση πληροφοριών προσανατολισμού
- Αυτή η κλάση παρέχει επίσης αρκετές **σταθερές αισθητήρων** οι οποίες χρησιμοποιούνται
  - για να αναφέρουν την ακρίβεια του αισθητήρα
  - για τον ορισμό των ρυθμών λήψης δεδομένων
  - για την βαθμονόμηση των αισθητήρων

```
private SensorManager sensorManager;
...
sensorManager = (SensorManager) getSystemService(Context.SENSOR_SERVICE);
```

```
List<Sensor> deviceSensors = sensorManager.getSensorList(Sensor.TYPE_ALL);
```

https://developer.android.com/reference/android/hardware/SensorManager

# Sensor Rates

- SensorManager.SENSOR_DELAY_NORMAL (delay 200000 microseconds) (default value)
- SensorManager.SENSOR_DELAY_GAME (delay 20000 microseconds)
- SensorManager.SENSOR_DELAY_UI (delay 60000 microseconds)
- SensorManager.SENSOR_DELAY_FASTEST (delay 0 microseconds)

https://source.android.com/devices/sensors/sensor-types

# Android Sensors API

**Sensor**

- Μπορείτε να χρησιμοποιήσετε αυτήν την κλάση για να δημιουργήσετε το **instance ενός συγκεκριμένου αισθητήρα**.
  - Αυτή η κλάση παρέχει διάφορες μεθόδους που σας επιτρέπουν να προσδιορίσετε τις δυνατότητες ενός αισθητήρα.

```
if (sensorManager.getDefaultSensor(Sensor.TYPE_MAGNETIC_FIELD) != null){
    // Success! There's a magnetometer.
} else {
    // Failure! No magnetometer.
}
```

https://developer.android.com/reference/android/hardware/Sensor.html

# SensorEventListener

- Μπορείτε να χρησιμοποιήσετε αυτήν διεπαφή για να δημιουργήσετε **δύο callback μεθόδους** που λαμβάνουν ειδοποιήσεις (συμβάντα αισθητήρων) όταν αλλάζουν
  1. οι τιμές του αισθητήρα ή
  2. όταν αλλάζει η ακρίβεια του αισθητήρα.

```
public class SensorActivity extends Activity implements SensorEventListener {
    private SensorManager sensorManager;
    private Sensor mLight;
    @Override
    public final void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.main);

        sensorManager = (SensorManager) getSystemService(Context.SENSOR_SERVICE);
        mLight = sensorManager.getDefaultSensor(Sensor.TYPE_LIGHT);
    }
    @Override
    public final void onAccuracyChanged(Sensor sensor, int accuracy) {
        // Do something here if sensor accuracy changes.
    }
    @Override
    public final void onSensorChanged(SensorEvent event) {
        // The light sensor returns a single value.
        // Many sensors return 3 values, one for each axis.
        float lux = event.values[0];
        // Do something with this sensor value.
    }
```

https://developer.android.com/reference/android/hardware/SensorEventListener.html

## SensorEvent

- Το σύστημα χρησιμοποιεί αυτή την κλάση για να δημιουργήσει ένα **αντικείμενο sensor event**, το οποίο παρέχει πληροφορίες σχετικά με ένα event αισθητήρα.
  - τα ακατέργαστα δεδομένα αισθητήρα
  - τον τύπο του αισθητήρα που δημιούργησε το συμβάν
  - την ακρίβεια των δεδομένων και τη χρονική σφραγίδα για το συμβάν

https://developer.android.com/reference/android/hardware/SensorEvent.html

# Πως χρησιμοποιείτε τα API (1)

Σε μια τυπική εφαρμογή χρησιμοποιείτε αυτά τα API που σχετίζονται με αισθητήρες για να εκτελέσετε δύο βασικές εργασίες:

**1. Προσδιορισμός αισθητήρων και δυνατοτήτων τους**.

- Η **αναγνώριση των αισθητήρων και των δυνατοτήτων τους** είναι χρήσιμη εάν η εφαρμογή σας διαθέτει χαρακτηριστικά που βασίζονται σε συγκεκριμένους τύπους ή δυνατότητες αισθητήρων.
  - Για παράδειγμα, μπορεί να θέλετε να αναγνωρίσετε όλους τους αισθητήρες που υπάρχουν σε μια συσκευή και να απενεργοποιήσετε τις λειτουργίες εφαρμογής που βασίζονται σε αισθητήρες που δεν υπάρχουν.
  - Ομοίως, μπορεί να θέλετε να εντοπίσετε όλους τους αισθητήρες ενός δεδομένου τύπου, π.χ. κίνησης, ώστε να μπορείτε να επιλέξετε την εφαρμογή αισθητήρων που έχει τη βέλτιστη απόδοση για την εφαρμογή σας.

# Πως χρησιμοποιείτε τα API (2)

**2. Παρακολούθηση συμβάντων αισθητήρων.**

- Είναι ο τρόπος με τον οποίο αποκτάτε ακατέργαστα δεδομένα αισθητήρα.
- **Ένα συμβάν αισθητήρα συμβαίνει κάθε φορά που ένας αισθητήρας ανιχνεύει μια αλλαγή στις παραμέτρους που μετράει**.
- Ένα συμβάν αισθητήρα σας παρέχει τέσσερα τεμάχια πληροφοριών:
  - το όνομα του αισθητήρα που ενεργοποίησε το συμβάν
  - τη χρονική σφραγίδα για το συμβάν
  - την ακρίβεια του συμβάντος
  - τα δεδομένα ακατέργαστου αισθητήρα που ενεργοποίησαν το συμβάν

```
private SensorManager mSensorManager;
private Sensor mSensor;
...
mSensorManager = (SensorManager) getSystemService(Context.SENSOR_SERVICE);
mSensor = mSensorManager.getDefaultSensor(Sensor.TYPE_LINEAR_ACCELERATION);


public void onSensorChanged(SensorEvent event){
    // In this example, alpha is calculated as t / (t + dT),
    // where t is the low-pass filter's time-constant and
    // dT is the event delivery rate.

    final float alpha = 0.8;

    // Isolate the force of gravity with the low-pass filter.
    gravity[0] = alpha * gravity[0] + (1 - alpha) * event.values[0];
    gravity[1] = alpha * gravity[1] + (1 - alpha) * event.values[1];
    gravity[2] = alpha * gravity[2] + (1 - alpha) * event.values[2];

    // Remove the gravity contribution with the high-pass filter.
    linear_acceleration[0] = event.values[0] - gravity[0];
    linear_acceleration[1] = event.values[1] - gravity[1];
    linear_acceleration[2] = event.values[2] - gravity[2];
}
```

# Καλές πρακτικές

- Unregister sensor listeners
- Don't block the onSensorChanged() method
- Avoid using deprecated methods or sensor types
- Verify sensors before you use them
- Choose sensor delays carefully

# LocationManager class

- public class LocationManager

- All Location API methods require the Manifest.permission.ACCESS_COARSE_LOCATION or Manifest.permission.ACCESS_FINE_LOCATION permissions.

https://developer.android.com/reference/android/location/LocationManager
https://developer.android.com/training/location
https://github.com/android/location-samples/tree/main/LocationUpdates

# LocationManager class

**Manifest file:**

```
<uses-permission android:name="android.permission.ACCESS_FINE_LOCATION" />
```

**LocationManager instance:**

```
LocationManager locationManager = (LocationManager)
getSystemService(Context.LOCATION_SERVICE);
```

**LocationListener instance:**

```
LocationListener locationListener = new MyLocationListener();
locationManager.requestLocationUpdates(
LocationManager.GPS_PROVIDER, 5000, 10, locationListener);
```

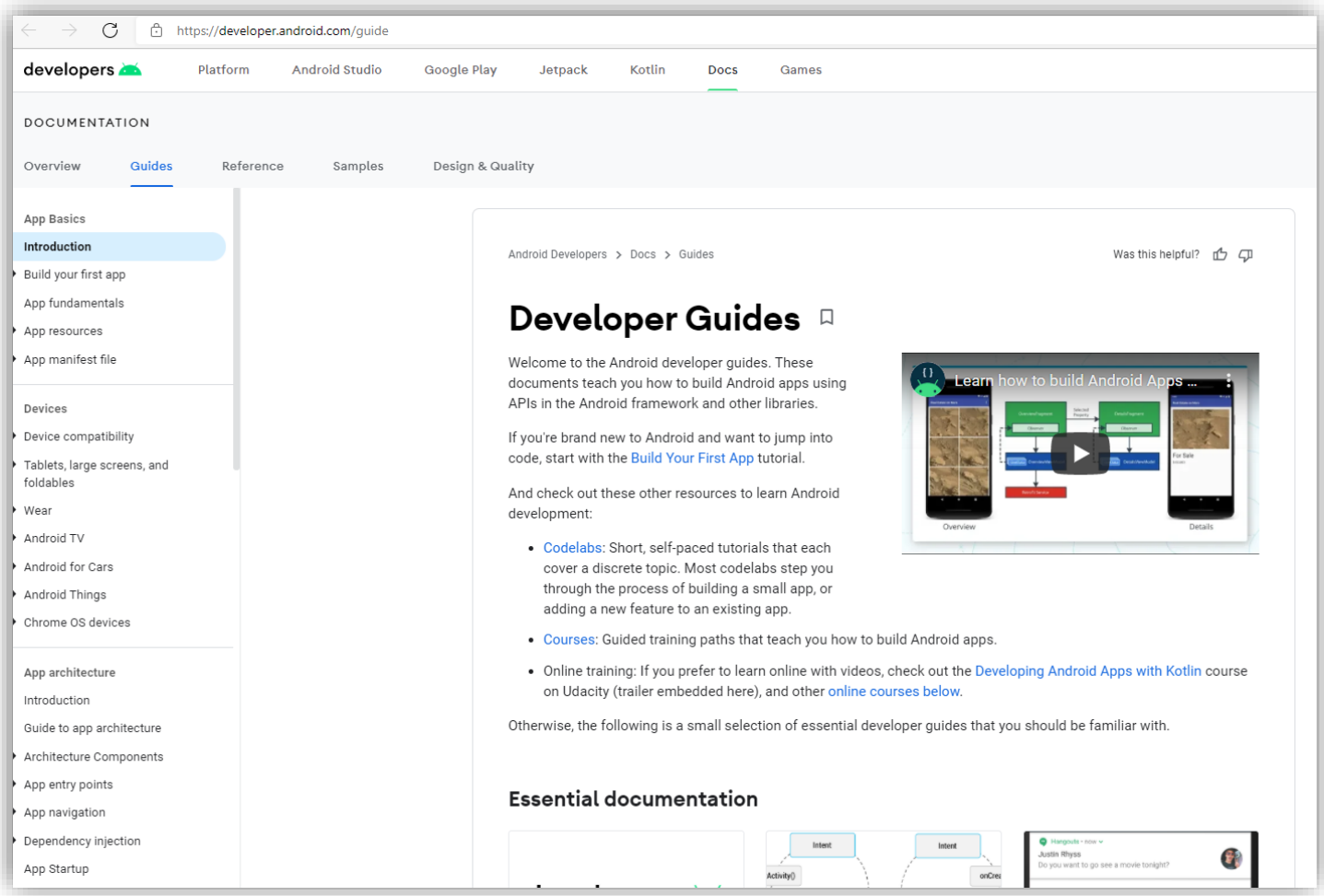
https://developer.android.com/training/location
https://stackoverflow.com/questions/1513485/how-do-i-get-the-current-gps-location-programmatically-in-android

```
private class MyLocationListener implements LocationListener {
    @Override
     public void onLocationChanged(Location loc) {
         editLocation.setText("");
         pb.setVisibility(View.INVISIBLE);
         Toast.makeText(getBaseContext(), "Location changed: Lat: " +
loc.getLatitude() + " Lng: "
                        + loc.getLongitude(), Toast.LENGTH_SHORT).show();
         String longitude = "Longitude: " + loc.getLongitude();
         Log.v(TAG, longitude);
         String latitude = "Latitude: " + loc.getLatitude();
         Log.v(TAG, latitude);
    @Override
    public void onProviderDisabled(String provider) {}
    @Override
    public void onProviderEnabled(String provider) {}
    @Override
    public void onStatusChanged(String provider, int status, Bundle extras) {}
}
```

- https://developer.android.com/training/location
- https://developer.android.com/training/location/retrieve-current
- https://developer.android.com/training/location/request-updates
- http://rdcworld-android.blogspot.com/2012/01/get-current-location-coordinates-city.html
- https://stackoverflow.com/questions/1513485/how-do-i-get-the-current-gps-location-programmatically-in-android

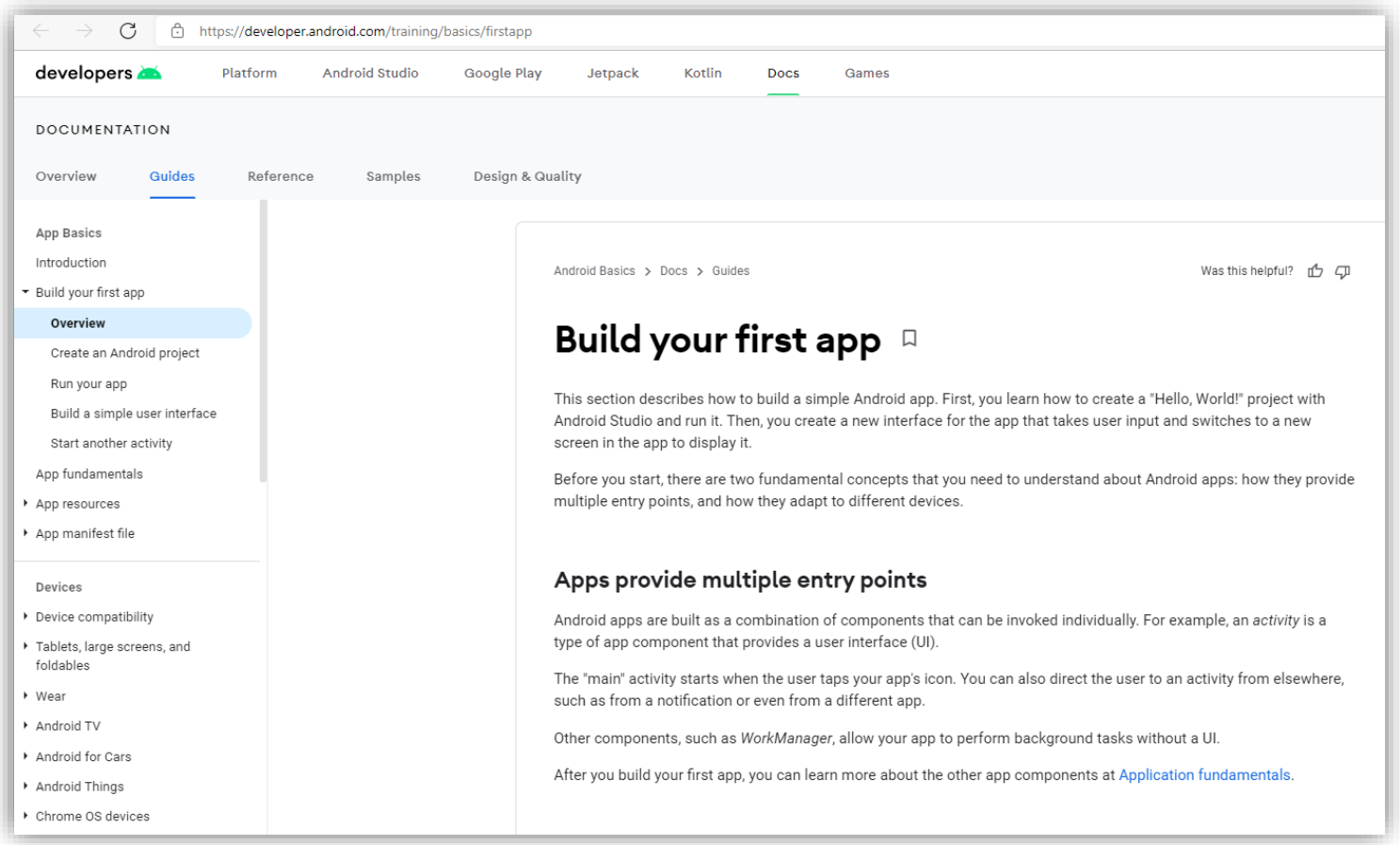
# ΠΕΡΑΙΤΕΡΩ ΜΕΛΕΤΗ

# Android Developer Guides




https://developer.android.com/guide/index.html

# Οδηγίες για την πρώτη σας εφαρμογή




**Σύνδεσμος:** https://developer.android.com/training/basics/firstapp/index.html

Ερωτήσεις ;

# Αναφορές

- Android Official Training:

  http://developer.android.com/training/

- Android Documentation:

  http://developer.android.com/reference/packages.html

- Android Permissions

  http://developer.android.com/guide/topics/security/permissions.html

- Android Sensor Overview

  https://developer.android.com/guide/topics/sensors/sensors_overview.html

# Αναφορές

- Log documentation:

  http://developer.android.com/reference/android/util/Log.html

- Android Web Services: http://mobileorchard.com/android-appdevelopment-calling-web-services/

- Android Tab Layout with Swipeable Views :

  http://www.androidhive.info/2013/10/android-tab-layout-withswipeable-views-1/

- Fragments:

  http://developer.android.com/guide/components/fragments.html

- ViewPager :

  http://developer.android.com/reference/android/support/v4/view/ViewPager.html