How to CREATE JAVA web applications with Android Studio

Ένας ολοκληρωμένος οδηγός για να γίνετε ένας προγραμματιστής Android Studio Java

By Vangelis Kakouras - www.studiowdev.click

Πρόλογος

Ένας προγραμματιστής Java μπαίνει σε ένα μπαρ και παραγγέλνει ένα ποτό. Στη συνέχεια βλέπει έναν προγραμματιστή Android να κάθεται δίπλα του και αποφασίζει να πιάσει κουβέντα.

"Γεια σου, είμαι προγραμματιστής Java. Εσύ τι κάνεις;" ρωτάει.

"Είμαι προγραμματιστής Android. Χρησιμοποιώ και εγώ Java." απαντά ο προγραμματιστής Android.

"Αλήθεια; Αυτό είναι τέλειο. Τι είδους εφαρμογές φτιάχνεις;" ρωτάει ο προγραμματιστής της Java.

"Λοιπόν, αυτή τη στιγμή δουλεύω πάνω σε μια εφαρμογή για τα μέσα κοινωνικής δικτύωσης. Ονομάζεται NullBook. Σας επιτρέπει να μοιράζεστε τις σκέψεις και τα συναισθήματά σας με άλλους χρήστες που δεν έχουν τίποτα κοινό με εσάς" λέει ο προγραμματιστής Android.

"NullBook; Αυτό ακούγεται ενδιαφέρον. Πώς λειτουργεί;" ρωτάει ο προγραμματιστής της Java.

"Είναι απλό. Απλώς δημιουργείτε ένα προφίλ με το όνομά σας, την ηλικία, το φύλο, την τοποθεσία, τα χόμπι, τα ενδιαφέροντά σας κ.λπ. Στη συνέχεια, μπορείτε να δημοσιεύσετε ό,τι θέλετε στο χρονολόγιό σας. Μπορείτε επίσης να ακολουθείτε άλλους χρήστες και να βλέπετε τις αναρτήσεις τους στην τροφοδοσία σας" εξηγεί ο προγραμματιστής Android.

"Ουάου, αυτό ακούγεται φοβερό. Πόσους χρήστες έχετε;" ρωτάει ο προγραμματιστής της Java.

"Κανέναν." λέει ο προγραμματιστής Android.

"Κανένας; Πώς γίνεται αυτό;" ρωτά ο προγραμματιστής της Java.

"Λοιπόν, κάθε φορά που κάποιος προσπαθεί να εγγραφεί ή να συνδεθεί, λαμβάνει μια NullPointerException." λέει ο προγραμματιστής Android.

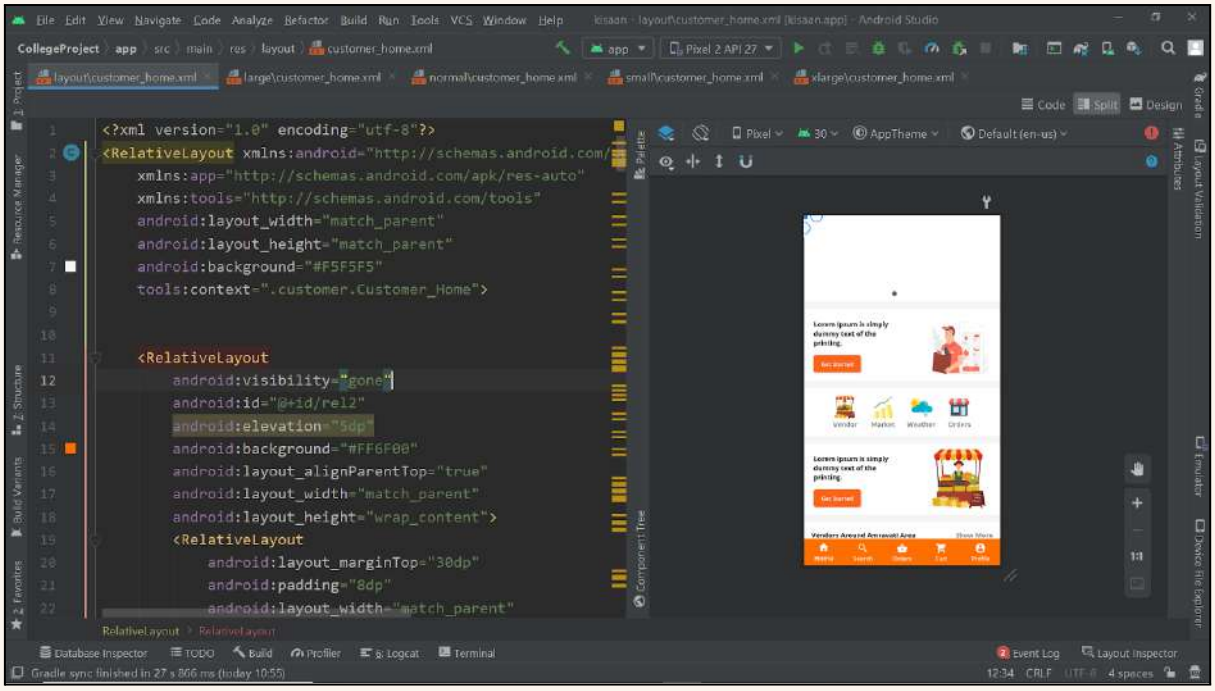
Vangelis Kakouras

ATHENS - March 2023

Πίνακας περιεχομένων

Εισαγωγή

Τι είναι το Android Studio και γιατί είναι ένα ισχυρό εργαλείο για την ανάπτυξη διαδικτυακών εφαρμογών Java



Το Android Studio είναι το επίσημο ολοκληρωμένο περιβάλλον ανάπτυξης (IDE) για την ανάπτυξη εφαρμογών Android. Βασίζεται στο IntelliJ IDEA, ένα ολοκληρωμένο περιβάλλον ανάπτυξης λογισμικού Java, και ενσωματώνει τα εργαλεία επεξεργασίας κώδικα και ανάπτυξης. Το Android Studio προσφέρει πολλά χαρακτηριστικά που βελτιώνουν την παραγωγικότητά σας κατά την ανάπτυξη εφαρμογών Android, όπως:

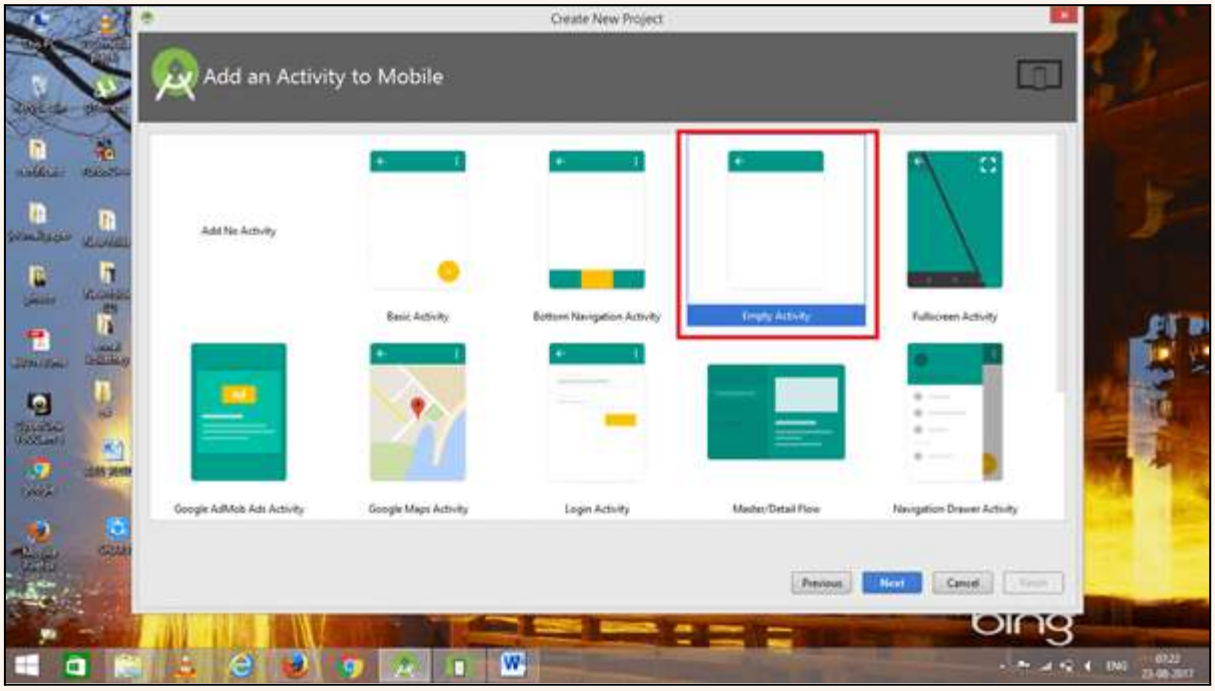
- ➔ Ένα ευέλικτο σύστημα δημιουργίας βασισμένο στο Gradle που σας επιτρέπει να προσαρμόζετε τις εξαρτήσεις, τις διαμορφώσεις και τις εξόδους της εφαρμογής σας.
- ➔ Ένας εξομοιωτής Android που σας επιτρέπει να δοκιμάζετε την εφαρμογή σας σε διαφορετικές συσκευές και διαμορφώσεις χωρίς να χρειάζεστε φυσικές συσκευές.
- ➔ Πρότυπα κώδικα και ενσωμάτωση στο GitHub που σας βοηθούν να δημιουργείτε τυποποιημένα στοιχεία της εφαρμογής και να συνεργάζεστε με άλλους προγραμματιστές.

- ➔ Ένας πλούσιος επεξεργαστής κώδικα που υποστηρίζει επισήμανση σύνταξης, συμπλήρωση κώδικα, αναδιαμόρφωση, αποσφαλμάτωση, δοκιμές, linting και πολλά άλλα.
- ➔ Μια ποικιλία εργαλείων και πρόσθετων που σας βοηθούν να σχεδιάζετε διεπαφές χρήστη, να αναλύετε θέματα επιδόσεων, να βελτιστοποιείτε το μέγεθος και την ταχύτητα της εφαρμογής σας, να προσθέτετε υπηρεσίες Firebase και πολλά άλλα.

Με το Android Studio, μπορείτε να αναπτύξετε εφαρμογές ιστού Java που εκτελούνται σε συσκευές Android και να έχετε πρόσβαση σε υπηρεσίες ιστού χρησιμοποιώντας τυπικές βιβλιοθήκες όπως η HttpURLConnection ή βιβλιοθήκες τρίτων κατασκευαστών όπως η Retrofit ή η Volley. Μπορείτε επίσης να χρησιμοποιήσετε τεχνολογίες ιστού όπως HTML5, CSS, JavaScript, jQuery, Bootstrap, AngularJS κ.λπ. για να δημιουργήσετε υβριδικές εφαρμογές χρησιμοποιώντας πλαίσια όπως το Cordova ή το Ionic.

Το Android Studio είναι ένα ισχυρό εργαλείο για την ανάπτυξη εφαρμογών ιστού Java, επειδή σας παρέχει όλα όσα χρειάζεστε για τη δημιουργία εφαρμογών υψηλής ποιότητας που λειτουργούν ομαλά σε διάφορες συσκευές. Σας βοηθά επίσης να μάθετε τις βέλτιστες πρακτικές ανάπτυξης Android και να παρακολουθείτε τις τελευταίες τάσεις και τεχνολογίες του κλάδου.

Παραδείγματα εφαρμογών web που μπορούν να κατασκευαστούν με το Android Studio



Το Android Studio σάς επιτρέπει να δημιουργείτε εφαρμογές ιστού που εκτελούνται σε συσκευές Android και έχουν πρόσβαση σε υπηρεσίες ιστού χρησιμοποιώντας τυπικές βιβλιοθήκες ή βιβλιοθήκες τρίτων κατασκευαστών. Μπορείτε επίσης να χρησιμοποιήσετε τεχνολογίες ιστού, όπως HTML5, CSS, JavaScript, jQuery, Bootstrap, AngularJS κ.λπ. για να δημιουργήσετε υβριδικές εφαρμογές που χρησιμοποιούν ένα στοιχείο WebView για την εμφάνιση περιεχομένου ιστού μέσα στην εφαρμογή σας. Μερικά παραδείγματα εφαρμογών ιστού που μπορούν να δημιουργηθούν με το Android Studio είναι:

- ➔ Μια εφαρμογή ειδήσεων που αντλεί και εμφανίζει άρθρα από διάφορες πηγές χρησιμοποιώντας ένα API όπως το NewsAPI ή τις ροές RSS.
- ➔ Μια εφαρμογή καιρού που εμφανίζει τις τρέχουσες και προβλεπόμενες καιρικές συνθήκες για διάφορες τοποθεσίες χρησιμοποιώντας ένα API όπως το OpenWeatherMap ή το Dark Sky.
- ➔ Μια εφαρμογή μέσων κοινωνικής δικτύωσης που επιτρέπει στους χρήστες να δημοσιεύουν, να κάνουν like, να σχολιάζουν και να μοιράζονται περιεχόμενο από διάφορες πλατφόρμες όπως το Facebook, το Twitter, το Instagram κ.λπ. χρησιμοποιώντας τα SDK ή τα API τους.
- ➔ Μια εφαρμογή αγορών που επιτρέπει στους χρήστες να περιηγηθούν, να αναζητήσουν, να συγκρίνουν και να αγοράσουν προϊόντα από διάφορα ηλεκτρονικά καταστήματα χρησιμοποιώντας ένα API όπως το Amazon Product Advertising API ή το Google Shopping API.
- ➔ Μια εφαρμογή γυμναστικής που παρακολουθεί και εμφανίζει τη σωματική δραστηριότητα των χρηστών, τις θερμίδες που καίγονται, τους καρδιακούς παλμούς κ.λπ. χρησιμοποιώντας αισθητήρες στις συσκευές τους ή φορητά αξεσουάρ όπως το Fitbit ή το Google Fit.

Αυτά είναι μερικά μόνο παραδείγματα εφαρμογών ιστού που μπορούν να κατασκευαστούν με το Android Studio. Μπορείτε επίσης να δημιουργήσετε τις δικές σας προσαρμοσμένες εφαρμογές ιστού για οποιονδήποτε σκοπό ή τομέα χρησιμοποιώντας τη δημιουργικότητα και τις δεξιότητές σας.

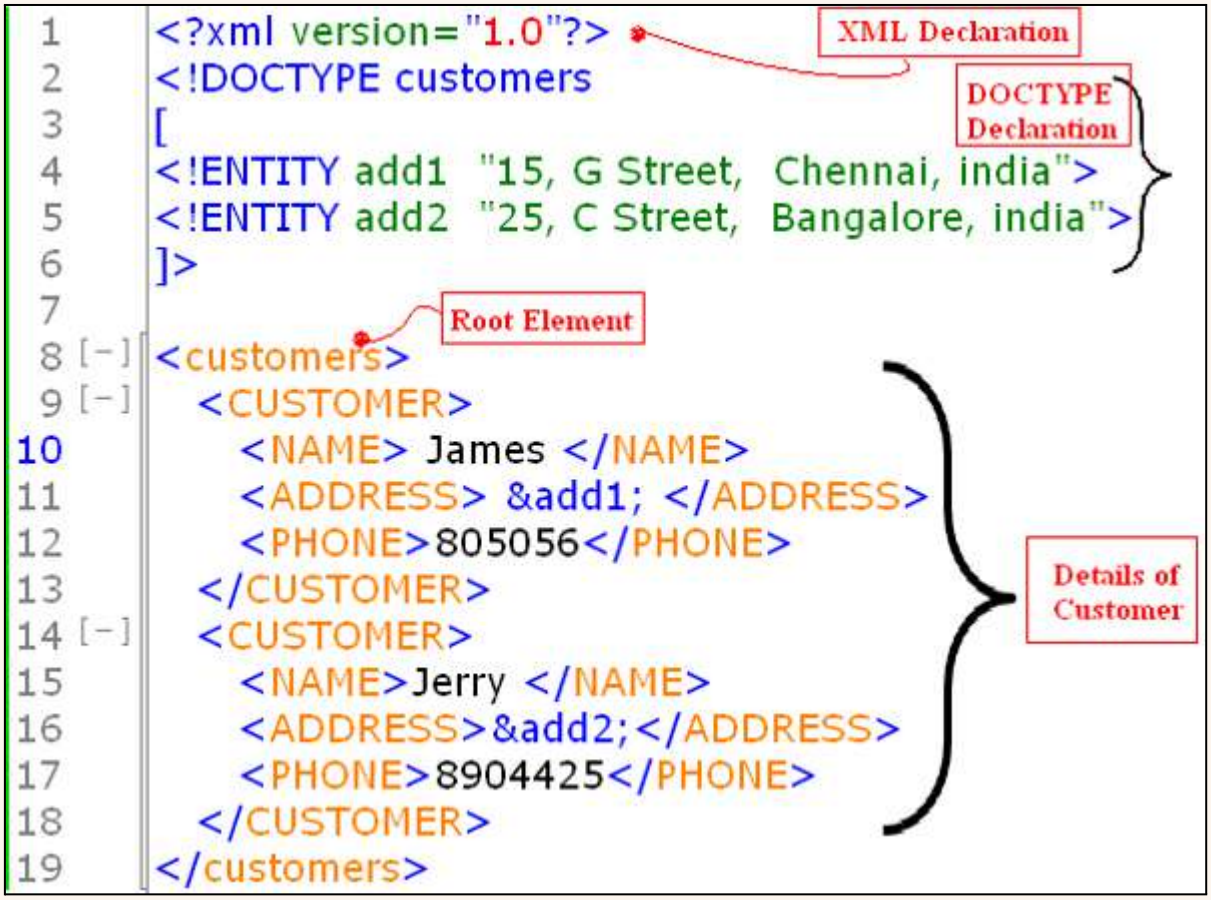
Οι προϋποθέσεις για να ακολουθήσετε αυτό το e-book

Πριν ξεκινήσετε να ακολουθείτε αυτό το ηλεκτρονικό βιβλίο, πρέπει να βεβαιωθείτε ότι έχετε εγκαταστήσει το Android Studio στον υπολογιστή σας και ότι έχετε κάποιες βασικές γνώσεις Java και XML. Το Android Studio είναι το επίσημο ολοκληρωμένο περιβάλλον ανάπτυξης (IDE) για την ανάπτυξη εφαρμογών Android. Μπορείτε να το κατεβάσετε από τον **επίσημο ιστότοπο**.

- ➔ Εάν κατεβάσατε ένα αρχείο .exe (συνιστάται), κάντε διπλό κλικ για να το εκκινήσετε.

- ➔ Αν κατεβάσατε ένα αρχείο .zip, αποσυμπιέστε το ZIP, αντιγράψτε το φάκελο android-studio στο φάκελο Program Files και, στη συνέχεια, ανοίξτε το φάκελο android-studio > bin και εκκινήστε το studio64.exe (για μηχανήματα 64-bit) ή το studio.exe (για μηχανήματα 32-bit).
- ➔ Ακολουθήστε τον οδηγό εγκατάστασης για να εγκαταστήσετε το Android Studio και όλα τα απαραίτητα εργαλεία SDK.

Η Java είναι η κύρια γλώσσα προγραμματισμού για την ανάπτυξη εφαρμογών Android. Θα πρέπει να έχετε κάποιες βασικές γνώσεις για τη σύνταξη της Java, τους τύπους δεδομένων, τις μεταβλητές, τους τελεστές, τις δομές ελέγχου, τους βρόχους, τους πίνακες, τις κλάσεις, τα αντικείμενα, την κληρονομικότητα, τον πολυμορφισμό, τις διεπαφές, τις εξαιρέσεις κ.λπ. Μπορείτε να μάθετε περισσότερα για τη **Java** από διαδικτυακά σεμινάρια ή βιβλία.



Η XML είναι μια γλώσσα σήμανσης (markup) που χρησιμοποιείται για τον καθορισμό διεπαφών χρήστη και πόρων για εφαρμογές Android. Πρέπει να έχετε κάποιες βασικές γνώσεις σύνταξης XML, στοιχείων, χαρακτηριστικών, χώρων ονομάτων, σχημάτων κ.λπ. Πρέπει επίσης να γνωρίζετε πώς να χρησιμοποιείτε ετικέτες XML όπως <LinearLayout>, <TextView>, <Button>, <ImageView> κ.λπ. και πως να δημιουργείτε διατάξεις και γραφικά

στοιχεία για τις οθόνες της εφαρμογής σας. Μπορείτε να μάθετε περισσότερα για την XML από ηλεκτρονικά σεμινάρια **όπως αυτό** ή βιβλία.

Δημιουργία ενός νέου project στο Android Studio

Πώς να ξεκινήσετε ένα νέο Android Studio project

Για να ξεκινήσετε ένα νέο έργο Android Studio και να δημιουργήσετε μια εφαρμογή ιστού, πρέπει να ακολουθήσετε τα εξής βήματα:

- ➔ Εκκινήστε το Android Studio και κάντε κλικ στην επιλογή "Start a new Android Studio project" στην οθόνη καλωσορίσματος.
- ➔ Στην οθόνη "Select a Project Template" (Επιλογή προτύπου έργου), επιλέξτε ως πρότυπο το "Empty Activity" (Κενή δραστηριότητα). Αυτό θα δημιουργήσει μια απλή εφαρμογή με μια δραστηριότητα και ένα αρχείο διάταξης.
- ➔ Στην οθόνη "Configure your project" (Διαμόρφωση του έργου σας), εισάγετε το όνομα της εφαρμογής σας, το όνομα του πακέτου, τη θέση αποθήκευσης, τη γλώσσα (Java ή Kotlin), την ελάχιστη έκδοση του SDK κ.λπ. Μπορείτε επίσης να αλλάξετε αυτές τις ρυθμίσεις αργότερα, αν χρειαστεί.
- ➔ Κάντε κλικ στο "Finish" (Τέλος) και περιμένετε το Android Studio να δημιουργήσει το έργο σας.
- ➔ Μόλις δημιουργηθεί το έργο σας, θα δείτε δύο αρχεία ανοιχτά στον επεξεργαστή: MainActivity.java (ή MainActivity.kt) και activity_main.xml. Αυτά είναι τα αρχεία όπου θα γράψετε τον κώδικά σας και θα σχεδιάσετε τη διεπαφή χρήστη σας αντίστοιχα.

Για να δημιουργήσετε μια διαδικτυακή εφαρμογή, πρέπει να προσθέσετε ένα στοιχείο WebView στο αρχείο διάταξης. Ένα WebView είναι μια προβολή που εμφανίζει ιστοσελίδες μέσα στην εφαρμογή σας. Μπορείτε να χρησιμοποιήσετε τυπικές βιβλιοθήκες ή βιβλιοθήκες τρίτων κατασκευαστών για να αποκτήσετε πρόσβαση σε υπηρεσίες ιστού από το WebView σας.

Για να προσθέσετε ένα WebView, ανοίξτε το αρχείο activity_main.xml και σύρετε ένα WebView από το παράθυρο Palette στο παράθυρο Design. Μπορείτε επίσης να επεξεργαστείτε απευθείας τον κώδικα XML μεταβαίνοντας στην καρτέλα Κείμενο στο κάτω μέρος του επεξεργαστή.

Για να φορτώσετε μια ιστοσελίδα στο WebView σας, πρέπει να προσθέσετε κάποιο κώδικα στο αρχείο MainActivity.java (ή MainActivity.kt). Πρώτον, πρέπει να λάβετε μια αναφορά στο WebView σας χρησιμοποιώντας τη μέθοδο findViewById(). Στη συνέχεια, πρέπει να ενεργοποιήσετε τη JavaScript για το WebView σας χρησιμοποιώντας την κλάση

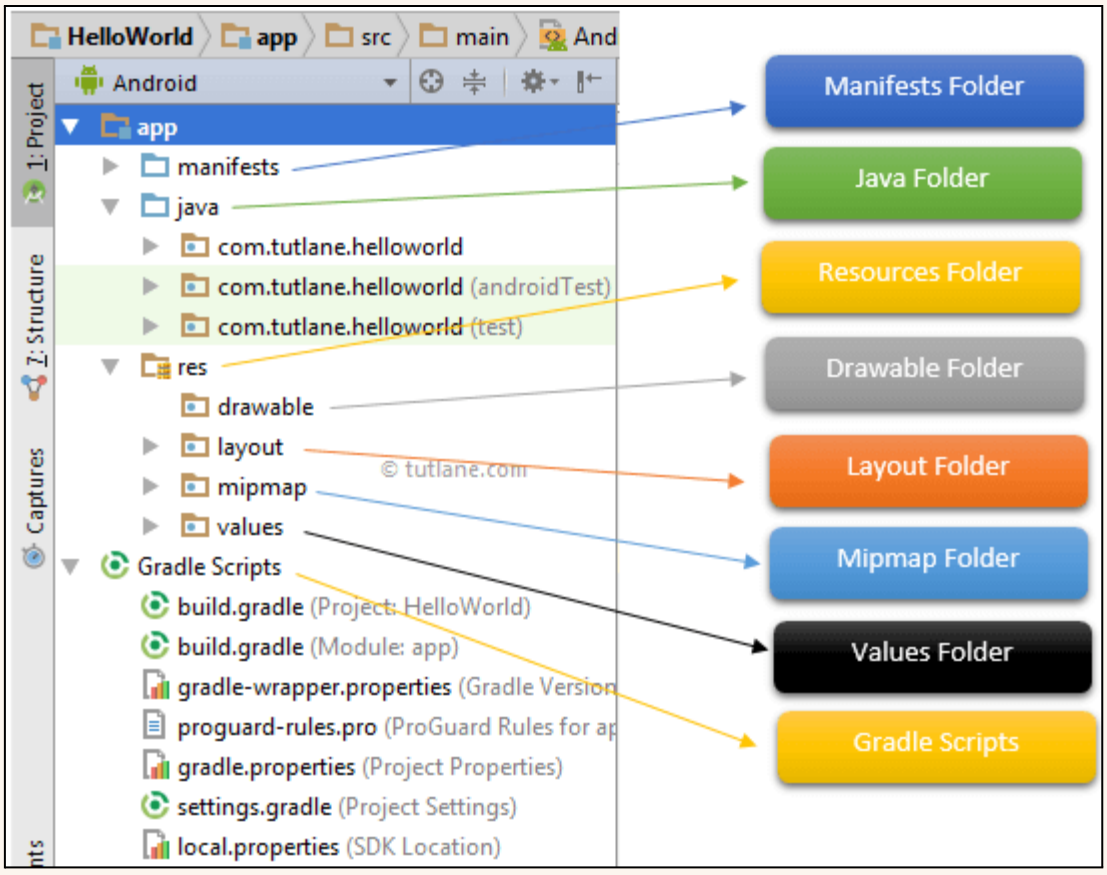
WebSettings. Τέλος, πρέπει να φορτώσετε μια διεύθυνση URL χρησιμοποιώντας τη μέθοδο loadUrl().

Ακολουθεί ένα παράδειγμα για το πώς μπορεί να μοιάζει ο κώδικάς σας:

```
import androidx.appcompat.app.AppCompatActivity;
import android.os.Bundle;
import android.webkit.WebSettings;
import android.webkit.WebView;

public class MainActivity extends AppCompatActivity {

    private WebView webView;

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

        // Get reference to WebView
        webView = findViewById(R.id.webView);

        // Enable JavaScript
        WebSettings webSettings = webView.getSettings();
        webSettings.setJavaScriptEnabled(true);

        // Load URL
        webView.loadUrl("https://www.example.com");
    }
}


// Kotlin code
import androidx.appcompat.app.AppCompatActivity
import android.os.Bundle
import android.webkit.WebSettings
import android.webkit.WebView

class MainActivity : AppCompatActivity() {

    private lateinit var webView: WebView

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_main)
```

```
        // Get reference to WebView
        webView = findViewById(R.id.webView)

        // Enable JavaScript
        val webSettings: WebSettings = webView.settings
        webSettings.javaScriptEnabled = true

        // Load URL
        webView.loadUrl("https://www.example.com")
    }
}
```

Η δομή ενός project Android



Ένα έργο Android είναι μια συλλογή αρχείων και φακέλων που καθορίζουν μια εφαρμογή και τα χαρακτηριστικά της. Ένα έργο Android έχει τα ακόλουθα κύρια στοιχεία:

Φάκελος Manifests: Ο φάκελος αυτός περιέχει το αρχείο AndroidManifest.xml, το οποίο δηλώνει βασικές πληροφορίες σχετικά με την εφαρμογή σας, όπως το όνομα του πακέτου, τα δικαιώματα, τις δραστηριότητες, τις υπηρεσίες, τους δέκτες εκπομπής, τους παρόχους περιεχομένου κ.λπ. Το αρχείο manifest καθορίζει επίσης τον τρόπο με τον οποίο η εφαρμογή σας αλληλεπιδρά με άλλες εφαρμογές και το σύστημα.

Φάκελος Java: Αυτός ο φάκελος περιέχει τα αρχεία πηγαίου κώδικα για τη λογική της εφαρμογής σας. Αυτά τα αρχεία είναι οργανωμένα σε πακέτα που αντιστοιχούν στο όνομα του πακέτου της εφαρμογής σας. Ο φάκελος Java περιέχει επίσης φακέλους δοκιμών για δοκιμές μονάδας και δοκιμές με όργανα.

Φάκελος res (Resources): Αυτός ο φάκελος περιέχει διάφορους τύπους αρχείων πόρων που καθορίζουν την εμφάνιση και τη συμπεριφορά της εφαρμογής σας. Αυτά περιλαμβάνουν:

Φάκελος Drawable: Αυτός ο φάκελος περιέχει εικόνες και άλλους σχεδιάσιμους πόρους που μπορούν να χρησιμοποιηθούν ως φόντο, εικονίδια, κουμπιά κ.λπ.

Φάκελος διάταξης (Layout): Αυτός ο φάκελος περιέχει αρχεία XML που καθορίζουν τις διατάξεις διεπαφής χρήστη για τις οθόνες της εφαρμογής σας. Κάθε αρχείο διάταξης αντιστοιχεί σε μια δραστηριότητα ή ένα τμήμα.

Φάκελος Mipmap: Αυτός ο φάκελος περιέχει εικονίδια εκκίνησης για διαφορετικές πυκνότητες οθόνης. Αυτά τα εικονίδια χρησιμοποιούνται από το σύστημα για την εμφάνιση της εφαρμογής σας στην αρχική οθόνη ή στο συρτάρι εφαρμογών.

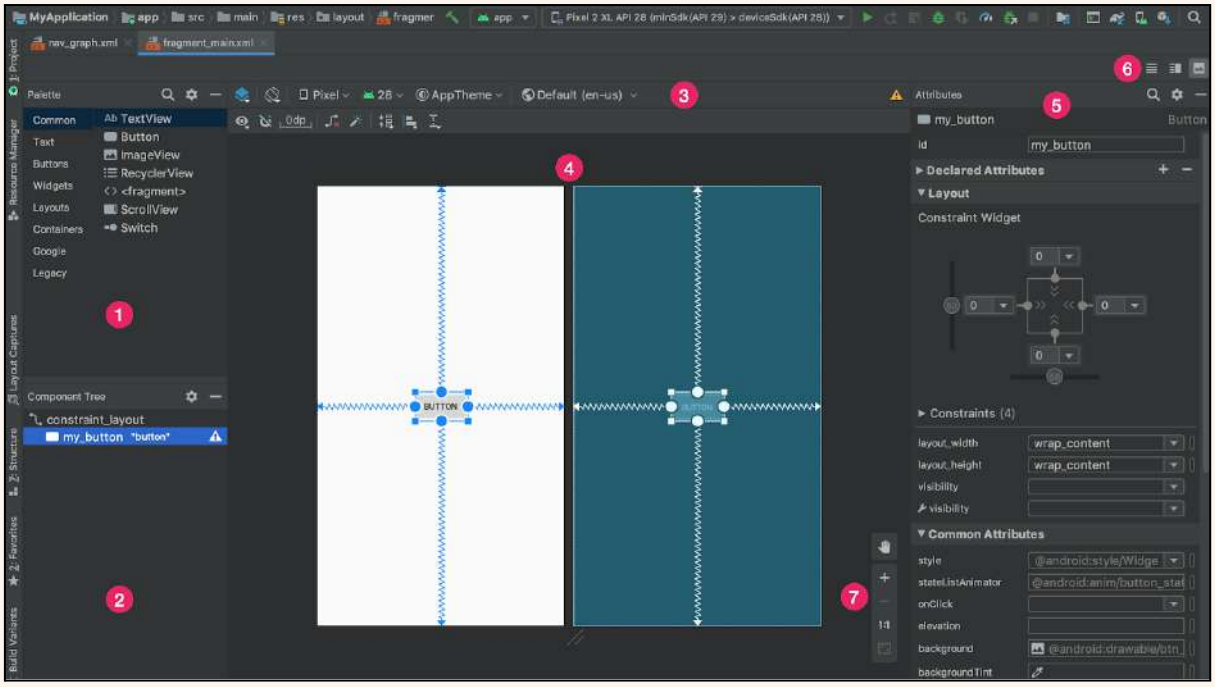
Φάκελος Values: Αυτός ο φάκελος περιέχει αρχεία XML που ορίζουν απλές τιμές όπως συμβολοσειρές, χρώματα, διαστάσεις, στυλ κ.λπ. Αυτές οι τιμές μπορούν να αναφέρονται από άλλα αρχεία πόρων ή αρχεία κώδικα.

Σενάρια Gradle (scripts): Πρόκειται για αρχεία που ρυθμίζουν τον τρόπο με τον οποίο το έργο σας κατασκευάζεται, δοκιμάζεται και αναπτύσσεται χρησιμοποιώντας το Gradle. Το Gradle είναι ένα ισχυρό εργαλείο αυτοματοποίησης κατασκευής που διαχειρίζεται εξαρτήσεις, παράγει APKs, εκτελεί δοκιμές κ.λπ.

Αυτά είναι τα κύρια συστατικά ενός έργου Android. Μπορείτε να τα εξερευνήσετε λεπτομερέστερα χρησιμοποιώντας το παράθυρο Project του Android Studio. Μπορείτε επίσης να προσθέσετε περισσότερα συστατικά, όπως βιβλιοθήκες, περιουσιακά στοιχεία (assets), εγγενή κώδικα κ.λπ. ανάλογα με τις απαιτήσεις της εφαρμογής σας.

Σχεδιάζοντας μια διεπαφή χρήστη (UI) με XML

Πώς να χρησιμοποιείτε τον επεξεργαστή διάταξης (layout editor) και τα drag-and-drop widgets



Ο επεξεργαστής διάταξης είναι ένα εργαλείο που σας βοηθά να σχεδιάσετε και να κάνετε προεπισκόπηση της διεπαφής χρήστη (UI) της εφαρμογής σας, σύροντας και αφήνοντας στοιχεία UI, όπως κουμπιά, πεδία κειμένου, εικόνες κ.λπ. σε μια εικονική οθόνη της συσκευής. Μπορείτε επίσης να επεξεργαστείτε τις ιδιότητες κάθε στοιχείου και να δείτε πώς φαίνονται σε διαφορετικές διαμορφώσεις συσκευών.

Για να χρησιμοποιήσετε τον επεξεργαστή διάταξης, πρέπει να δημιουργήσετε ένα αρχείο XML διάταξης που ορίζει τη δομή και την εμφάνιση του UI σας. Μπορείτε να το κάνετε αυτό επιλέγοντας File > New > XML > Layout XML File από το κύριο μενού του Android Studio. Μπορείτε επίσης να ανοίξετε ένα υπάρχον αρχείο διάταξης από το παράθυρο Project (Εργο).

Μόλις ανοίξετε ένα αρχείο διάταξης, μπορείτε να αλλάξετε μεταξύ δύο τρόπων λειτουργίας: Σχεδίαση και Κώδικας. Η λειτουργία Σχεδίασης σας εμφανίζει μια γραφική αναπαράσταση του UI σας, όπου μπορείτε να σύρετε και να αποθέσετε στοιχεία από το παράθυρο Παλέτα στην επιφάνεια σχεδίασης. Η λειτουργία Code (Κώδικας) σας εμφανίζει τον κώδικα XML που αντιστοιχεί στο σχεδιασμό του UI σας. Μπορείτε να

εναλλάσσεστε μεταξύ αυτών των λειτουργιών κάνοντας κλικ στις καρτέλες στο κάτω μέρος του παραθύρου του επεξεργαστή.

Για να προσθέσετε ένα στοιχείο στο UI σας, πρέπει να το επιλέξετε από το παράθυρο Palette και να το σύρετε στην επιφάνεια σχεδίασης. Μπορείτε επίσης να χρησιμοποιήσετε συντομεύσεις πληκτρολογίου ή μενού δεξιού κλικ για να προσθέσετε στοιχεία. Στη συνέχεια, μπορείτε να προσαρμόσετε τη θέση και το μέγεθός του σύροντας τις λαβές του ή εισάγοντας τιμές στο παράθυρο Attributes.

Για να ενεργοποιήσετε τη λειτουργικότητα drag-and-drop στην εφαρμογή σας, πρέπει να υλοποιήσετε ορισμένες μεθόδους που χειρίζονται συμβάντα drag, όπως εκκίνηση, είσοδος, έξοδος, ρίψη κ.λπ. Πρέπει επίσης να παρέχετε κάποια δεδομένα που αναπαριστούν αυτό που σύρεται και κάποια μεταδεδομένα που το περιγράφουν. Μπορείτε να ξεκινήσετε μια λειτουργία drag-and-drop καλώντας την startDragAndDrop() σε οποιαδήποτε προβολή της τρέχουσας διάταξής σας.

Πώς να χρησιμοποιείτε χαρακτηριστικά, στυλ, θέματα και πόρους για να προσαρμόσετε την εμφάνιση και τη συμπεριφορά των widgets

Τα χαρακτηριστικά (Attributes) είναι ιδιότητες που καθορίζουν την εμφάνιση και τη συμπεριφορά μιας προβολής (όπως ένα κουμπί, ένα πεδίο κειμένου, μια προβολή εικόνας κ.λπ.). Για παράδειγμα, μπορείτε να χρησιμοποιήσετε τα χαρακτηριστικά για να ορίσετε το χρώμα γραμματοσειράς, το μέγεθος γραμματοσειράς, το χρώμα φόντου, τη συμπλήρωση, το περιθώριο κ.λπ. μιας Προβολής. Μπορείτε να ορίσετε χαρακτηριστικά σε XML χρησιμοποιώντας το πρόθεμα android: ακολουθούμενο από το όνομα και την τιμή του χαρακτηριστικού. *Για παράδειγμα:*

```
<Button
    android:id="@+id/button"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="Click me"
    android:textColor="#FFFFFF"
    android:background="#FF0000"/>
```

Τα στυλ (Styles) είναι συλλογές χαρακτηριστικών που καθορίζουν την εμφάνιση για πολλαπλές προβολές. Για παράδειγμα, μπορείτε να δημιουργήσετε ένα στυλ που ορίζει μια κοινή εμφάνιση για όλα τα κουμπιά της εφαρμογής σας. Μπορείτε να ορίσετε στυλ σε XML χρησιμοποιώντας την ετικέτα <style> μέσα σε ένα αρχείο πόρων (συνήθως res/values/styles.xml). *Για παράδειγμα:*

```
<style name="MyButtonStyle">
    <item name="android:textColor">#FFFFFF</item>
    <item name="android:background">#FF0000</item>
</style>
```

Στη συνέχεια, μπορείτε να εφαρμόσετε ένα στυλ σε μια Προβολή χρησιμοποιώντας το χαρακτηριστικό στυλ με το όνομα του στυλ. *Για παράδειγμα:*

```
<Button
    style="@style/MyButtonStyle"
    android:id="@+id/button"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="Click me"/>
```

Τα θέματα (Themes) είναι συλλογές στυλ που καθορίζουν την εμφάνιση μιας ολόκληρης εφαρμογής ή δραστηριότητας. Για παράδειγμα, μπορείτε να δημιουργήσετε ένα θέμα που ορίζει ένα κοινό συνδυασμό χρωμάτων και τυπογραφίας για την εφαρμογή σας. Μπορείτε να ορίσετε θέματα σε XML χρησιμοποιώντας την ετικέτα <style> με ένα χαρακτηριστικό parent μέσα σε ένα αρχείο πόρων (συνήθως res/values/themes.xml). *Για παράδειγμα:*

```
<style name="MyTheme"
parent="Theme.AppCompat.Light.DarkActionBar">
    <item name="colorPrimary">#FF0000</item>
    <item name="colorPrimaryDark">#AA0000</item>
    <item name="colorAccent">#FFFF00</item>
</style>
```

Στη συνέχεια, μπορείτε να εφαρμόσετε ένα θέμα σε μια εφαρμογή ή δραστηριότητα χρησιμοποιώντας το χαρακτηριστικό android:theme με το όνομα του θέματος. *Για παράδειγμα:*

```
<application
    ...
    android:theme="@style/MyTheme">

<activity
   ...
   android:theme="@style/MyTheme">
```

Οι πόροι (Resources) είναι αρχεία που περιέχουν δεδομένα ή ορισμούς που δεν αποτελούν μέρος του κώδικα της εφαρμογής σας, αλλά χρησιμοποιούνται από την εφαρμογή σας κατά την εκτέλεση. Για παράδειγμα, μπορείτε να χρησιμοποιήσετε πόρους για να αποθηκεύσετε συμβολοσειρές, χρώματα, διαστάσεις, drawables (εικόνες), layouts (σχέδια UI) κ.λπ. Μπορείτε να ορίσετε πόρους σε XML χρησιμοποιώντας διαφορετικές ετικέτες μέσα σε διαφορετικά αρχεία πόρων (συνήθως στο φάκελο res/values). *Για παράδειγμα:*

```
<resources>
  <!-- Strings -->
  <string name="app_name">My App</string>
  <string name="button_text">Click me</string>

  <!-- Colors -->
  <color name="white">#FFFFFF</color>
  <color name="red">#FF0000</color>

  <!-- Dimensions -->
  <dimen name="button_width">100dp</dimen>
  <dimen name="button_height">50dp</dimen>

  <!-- Drawables -->
  <!-- See res/drawable folder -->

  <!-- Layouts -->
  <!-- See res/layout folder -->
</resources>
```

Στη συνέχεια, μπορείτε να αναφέρεστε σε πόρους στον κώδικα ή την XML σας χρησιμοποιώντας το @ ακολουθούμενο από τον τύπο και το όνομα του πόρου. *Για παράδειγμα:*

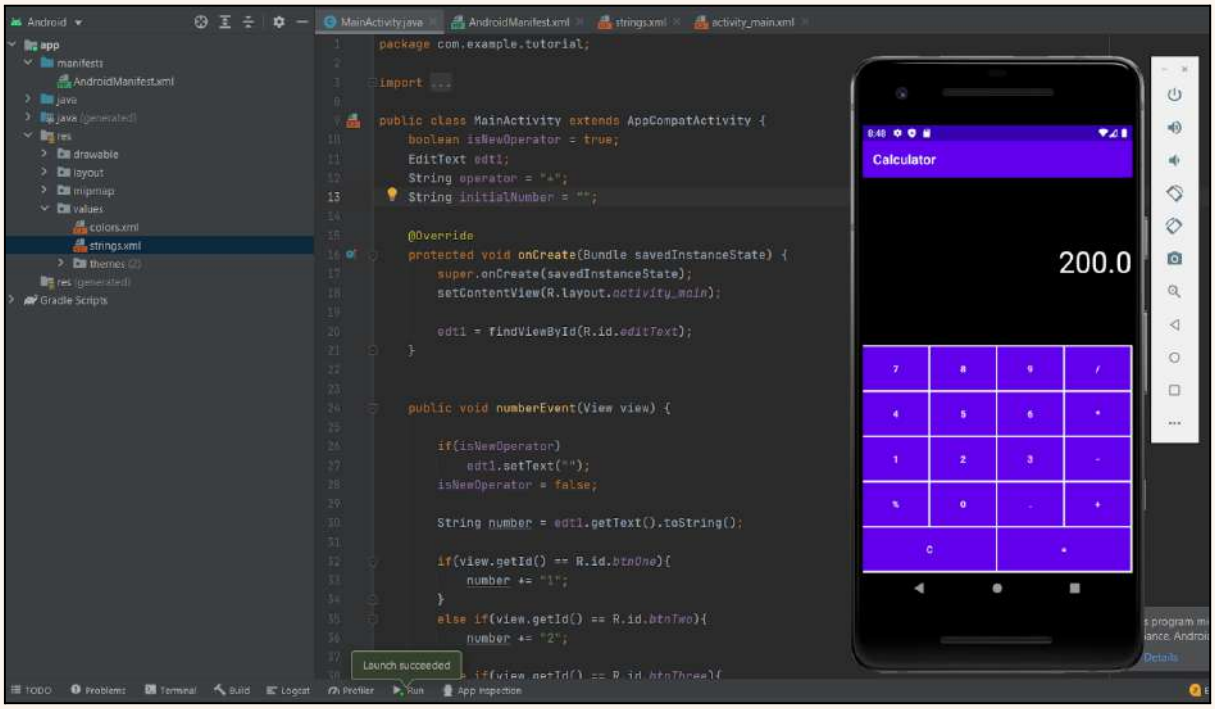
```
String appName = getResources().getString(R.string.app_name);
int buttonWidth =
getResources().getDimensionPixelSize(R.dimen.button_width);
Drawable buttonBackground =
getResources().getDrawable(R.drawable.button_background);
```

```
<TextView
   ...
   android:text="@string/app_name"/>
```

```
<Button
    ...
    android:layout_width="@dimen/button_width"
    android:layout_height="@dimen/button_height"
    android:background="@drawable/button_background"/>
```

Κωδικοποίηση με τη Java

Πώς να χρησιμοποιείτε τον επεξεργαστή κώδικα και να γράφετε κώδικα Java



Ο επεξεργαστής κώδικα (code editor) είναι ένα εργαλείο που σας βοηθά να γράψετε, να επεξεργαστείτε, να αποσφαλματώσετε και να εκτελέσετε κώδικα Java για την εφαρμογή Android. Μπορείτε να αποκτήσετε πρόσβαση στον επεξεργαστή κώδικα ανοίγοντας οποιοδήποτε αρχείο Java (συνήθως στο φάκελο src/main/java) από το παράθυρο Project.

Ο επεξεργαστής κώδικα παρέχει πολλά χαρακτηριστικά που κάνουν τον προγραμματισμό ευκολότερο και ταχύτερο, όπως επισήμανση σύνταξης, συμπλήρωση κώδικα, αναδιαμόρφωση, μορφοποίηση, χώνευση, δοκιμές κ.λπ. Μπορείτε επίσης να χρησιμοποιήσετε συντομεύσεις πληκτρολογίου ή μενού για να εκτελέσετε διάφορες ενέργειες στον κώδικά σας.

Για να γράψετε κώδικα Java για την εφαρμογή σας Android, πρέπει να ακολουθήσετε ορισμένα βασικά βήματα:

- ➔ Δημιουργήστε ένα έργο Android στο Android Studio επιλέγοντας File > New > New Project από το κύριο μενού.
- ➔ Επιλέξτε ένα πρότυπο για την κύρια δραστηριότητα της εφαρμογής σας (όπως το Empty Activity) και δώστε κάποιες πληροφορίες για την εφαρμογή σας (όπως όνομα, όνομα πακέτου, γλώσσα κ.λπ.).
- ➔ Γράψτε τη λογική και τη λειτουργικότητα της εφαρμογής σας σε κλάσεις Java που επεκτείνονται από την AppCompatActivity ή άλλες βασικές κλάσεις. Μπορείτε επίσης να δημιουργήσετε προσαρμοσμένες προβολές ή στοιχεία επεκτείνοντας από την View ή άλλες υποκλάσεις.
- ➔ Χρησιμοποιήστε τη μέθοδο findViewById() για να αποκτήσετε πρόσβαση στα στοιχεία UI (όπως κουμπιά, πεδία κειμένου, εικόνες κ.λπ.) που έχετε ορίσει στα αρχεία XML της διάταξης (συνήθως στο φάκελο res/layout). Μπορείτε επίσης να χρησιμοποιήσετε το view binding ή το data binding για να απλοποιήσετε αυτή τη διαδικασία.
- ➔ Προσθέστε ακροατές συμβάντων ή callbacks για να χειρίζεστε τις αλληλεπιδράσεις του χρήστη (όπως κλικ, άγγιγμα, σάρωση κ.λπ.) με τα στοιχεία UI σας. Μπορείτε επίσης να χρησιμοποιήσετε intents ή θραύσματα για την πλοήγηση μεταξύ διαφορετικών οθονών ή δραστηριοτήτων.
- ➔ Χρησιμοποιήστε API του Android (όπως αισθητήρες, κάμερα, τοποθεσία, αποθήκευση, δικτύωση κ.λπ.) για να αποκτήσετε πρόσβαση σε λειτουργίες και υπηρεσίες της συσκευής. Μπορείτε επίσης να χρησιμοποιήσετε βιβλιοθήκες ή πλαίσια τρίτων για να προσθέσετε περισσότερες λειτουργίες στην εφαρμογή σας.
- ➔ Να δοκιμάζετε και να αποσφαλματώνετε την εφαρμογή σας χρησιμοποιώντας εξομοιωτές ή πραγματικές συσκευές. Μπορείτε επίσης να χρησιμοποιήσετε το logcat,
- ➔ breakpoints ή δοκιμές μονάδας για να βρείτε και να διορθώσετε σφάλματα στον κώδικά σας.

Πώς να χρησιμοποιείτε μεταβλητές, τύπους δεδομένων, τελεστές, δομές ελέγχου, μεθόδους, κλάσεις, αντικείμενα, κληρονομικότητα, διεπαφές, εξαιρέσεις, συλλογές, γενιές και επισημειώσεις στη Java

Οι μεταβλητές (variables) είναι δοχεία που αποθηκεύουν τιμές διαφορετικών τύπων. Μπορείτε να δηλώσετε μια μεταβλητή καθορίζοντας το όνομα και τον τύπο δεδομένων της. *Για παράδειγμα:*

```
int x; // declare an integer variable named x
x = 10; // assign 10 to x
String name; // declare a string variable named name
name = "Alice"; // assign "Alice" to name
```

Οι τύποι δεδομένων (Data types) είναι κατηγορίες που καθορίζουν το μέγεθος και τον τύπο των τιμών των μεταβλητών. Υπάρχουν δύο είδη τύπων δεδομένων στη Java: primitive και reference.

Οι πρωτογενείς τύποι δεδομένων είναι προκαθορισμένοι από τη γλώσσα και δεν διαθέτουν πρόσθετες μεθόδους. Υπάρχουν οκτώ πρωτόγονοι τύποι δεδομένων στη Java:

byte, short, int, long, float, double, char και boolean. *Για παράδειγμα:*

```
byte b = 127; // a byte can store values from -128 to 127
short s = 32767; // a short can store values from -32768 to 32767
int i = 2147483647; // an int can store values from -2147483648 to
2147483647
long l = 92233720368547758O7L; // a long can store values from
-92233720368547758O8L to 92233720368547758O7L
float f = 3.14f; // a float can store decimal numbers with up to
6-7 digits of precision
double d = 3.141592653589793d; // a double can store decimal
numbers with up to 15-16 digits of precision
char c = 'A'; // a char can store a single character using Unicode
encoding
boolean bool = true; // a boolean can store either true or false
values
```

Οι τύποι δεδομένων αναφοράς(Reference Data types) ορίζονται από κλάσεις και διαθέτουν πρόσθετες μεθόδους. Ένας τύπος δεδομένων αναφοράς αποθηκεύει τη διεύθυνση ή την αναφορά ενός αντικειμένου (μια περίπτωση μιας κλάσης) και όχι την πραγματική τιμή. *Για παράδειγμα:*

```
String str = "Hello"; // a string is an object that represents a
sequence of characters
Integer num = new Integer(10); // an integer is an object that
wraps an int value as an object
Scanner sc = new Scanner(System.in); // a scanner is an object
that allows reading input from various sources (such as keyboard)
```

Οι τελεστές(Operators) είναι σύμβολα που εκτελούν πράξεις σε ένα ή περισσότερα τελεστέα (μεταβλητές ή τιμές). Υπάρχουν διάφορα είδη τελεστών στη Java: αριθμητικοί, ανάθεσης, σχετικοί, λογικοί, δυαδικοί, μοναδιαίοι, τριμερείς και άλλοι. *Για παράδειγμα:*

```
// arithmetic operators perform basic mathematical operations such
as addition (+), subtraction (-), multiplication (*), division
(/), modulus (%), and exponentiation (**)
int sum = x + y; // add x and y and assign the result to sum
int diff = x - y; // subtract y from x and assign the result to
diff
int prod = x * y; // multiply x and y and assign the result to
prod
int quot = x / y; // divide x by y and assign the result to quot
int rem = x % y; // get the remainder of dividing x by y and
assign it to rem
double pow = Math.pow(x,y); // raise x to the power of y and
assign it to pow

// assignment operators assign values to variables using symbols
such as (=), (+=), (-=), (*=), (/=), (%=), etc.
x += 5; // equivalent to x = x + 5;
y -= 2; // equivalent to y = y - 2;
z *= 3; // equivalent to z = z * 3;
w /= 4; / equivalent t w w / ;
v %= ; / equivalent t v v % ;

// relational operators compare two operands and return a boolean
value based on their relationship such as equal (==),
not equal (!=),
greater than (>),
less than (<),
greater than or equal (>=),
less than or equal (<=)
bool b1 b == ; / b true if b equals ;
bool b2 b != ; / b true if b does not equal ;
bool b3 b > ; / b true if b greater than ;
bool b4 < ; /
```

Οι δομές ελέγχου (Control structures) είναι εντολές που ελέγχουν τη ροή της εκτέλεσης σε ένα πρόγραμμα. Υπάρχουν διάφορα είδη δομών ελέγχου στη Java: υπό όρους (if/else),

επαναληπτικές for/while/do-while), και διακλαδώσεις (break/continue/return). *Για παράδειγμα:*

```
// conditional statements execute a block of code based on a condition
if (x > 10) { // if x is greater than 10
  System.out.println("x is large"); // print "x is large"
} else { // otherwise
  System.out.println("x is small"); // print "x is small"
}

// iterative statements execute a block of code repeatedly until a condition is met
for (int i = 0; i < 10; i++) { // for i from 0 to 9
  System.out.println(i); // print i
}

while (x < 100) { // while x is less than 100
  x = x * 2; // double x
}

do {
  y = y + 1; // increment y
} while (y < 50); // do this while y is less than 50

// branching statements alter the normal flow of execution by jumping to another point in the program
break; // exit the current loop or switch statement
continue; // skip the rest of the current iteration and start a new one
return z; // return z as the value of the current method and exit it
```

Οι μέθοδοι (Methods) είναι τμήματα κώδικα που εκτελούν μια συγκεκριμένη εργασία και μπορούν να επαναχρησιμοποιηθούν. Μπορείτε να ορίσετε μια μέθοδο καθορίζοντας το όνομα, τις παραμέτρους, τον τύπο επιστροφής και το σώμα της. Μπορείτε επίσης να καλέσετε μια μέθοδο χρησιμοποιώντας το όνομά της και περνώντας ορίσματα. *Για παράδειγμα:*

```
// define a method that calculates the area of a circle given its radius as a parameter and returns it as a double value
```

```
public static double areaOfCircle(double radius) {
   double area = Math.PI * radius * radius; // calculate area using
Math.PI constant and radius parameter
   return area; / return area as result
}

// call the method by using its name and passing an argument

double circleArea = areaOfCircle(5); / call areaOfCircle method with
argument ; assign returned value t circleArea variable
System.out.println(circleArea); / print circleArea variable
```

Οι κλάσεις (Classes) είναι προσχέδια ή πρωτότυπα που ορίζονται από τον χρήστη και από τα οποία δημιουργούνται αντικείμενα. Αντιπροσωπεύουν το σύνολο των ιδιοτήτων ή χαρακτηριστικών (πεδία) και των συμπεριφορών ή ενεργειών (μέθοδοι) που είναι κοινά για όλα τα αντικείμενα ενός τύπου. Μπορείτε να ορίσετε μια κλάση χρησιμοποιώντας τη λέξη-κλειδί class ακολουθούμενη από το όνομα και το σώμα της. Μπορείτε επίσης να δημιουργήσετε ένα αντικείμενο χρησιμοποιώντας τη λέξη-κλειδί new ακολουθούμενη από το όνομα της κλάσης και τις παραμέτρους του κατασκευαστή. *Για παράδειγμα:*

```
// define a class that represents a person with fields for name,
age, and gender, and methods for getting and setting these fields

public class Person {
   private String name; / declare private field for name
   private int age; / declare private field for age
   private char gender; / declare private field for gender

   public Person(String name, int age, char gender) { / define
constructor with parameters for name age gender
      this.name = name; / assign parameter value t field using
this keyword
      this.age = age;
      this.gender = gender;
   }

   public String getName() { / define getter method for name field
      return this.name;
   }

   public void setName(String newName) { / define setter method
for name field with parameter for new value
```

```
        this.name = newName;
    }

    public int getAge() { / define getter method for age field
        return this.age;
    }
```

Τα αντικείμενα (Objects) είναι περιπτώσεις κλάσεων που έχουν κατάσταση (πεδία) και συμπεριφορά (μέθοδοι). Μπορείτε να δημιουργήσετε ένα αντικείμενο χρησιμοποιώντας τη λέξη κλειδί new ακολουθούμενη από το όνομα της κλάσης και τις παραμέτρους του κατασκευαστή. Μπορείτε επίσης να προσπελάσετε και να τροποποιήσετε τα πεδία και τις μεθόδους ενός αντικειμένου χρησιμοποιώντας τον τελεστή τελείας (.). *Για παράδειγμα:*

```
// create an object of Person class with name "Bob", age 25, and
gender 'M'
Person bob = new Person("Bob", 25, 'M');

// access and print the name field of bob object
System.out.println(bob.getName());

// modify the name field of bob object to "Robert"
bob.setName("Robert");

// access and print the name field of bob object again
System.out.println(bob.getName());
```

Η κληρονομικότητα (Inheritance) είναι ένας μηχανισμός που επιτρέπει σε μια κλάση να αποκτά ή να κληρονομεί τις ιδιότητες και τις συμπεριφορές μιας άλλης κλάσης. Η κλάση που κληρονομεί ονομάζεται υποκλάση ή κλάση-παιδί. Η κλάση από την οποία κληρονομείται ονομάζεται υπερκλάση ή γονική κλάση. Μπορείτε να χρησιμοποιήσετε την κληρονομικότητα για να επαναχρησιμοποιήσετε υπάρχοντα κώδικα, να αποφύγετε την επανάληψη και να δημιουργήσετε σχέσεις μεταξύ κλάσεων. Μπορείτε να χρησιμοποιήσετε τη λέξη-κλειδί extends για να δηλώσετε ότι μια υποκλάση κληρονομεί από μια υπερκλάση. *Για παράδειγμα:*

```
// define a subclass named Student that inherits from Person
superclass

public class Student extends Person {
```

```
    private int id; // declare private field for id
    private double gpa; // declare private field for gpa

    public Student(String name, int age, char gender, int id,
double gpa) { // define constructor with parameters for name age
gender id gpa
        super(name, age, gender); // call superclass constructor
with parameters for name age gender using super keyword
        this.id = id; // assign parameter value to field using this
keyword
        this.gpa = gpa;
    }

    public int getId() { // define getter method for id field
        return this.id;
    }

    public void setId(int newId) { // define setter method for id
field with parameter for new value
        this.id = newId;
    }

    public double getGpa() { // define getter method for gpa field
        return this.gpa;
    }

    public void setGpa(double newGpa) { // define setter method for
gpa field with parameter for new value
        this.gpa = newGpa;
    }
}

// create an object of Student subclass with name "Alice", age 20,
gender 'F', id 1234, and gpa 3.5

Student alice = new Student("Alice", 20, 'F', 1234, 3.5);

// access and print the fields inherited from Person superclass

System.out.println(alice.getName());
System.out.println(alice.getAge());
System.out.println(alice.getGender());

// access and print the fields defined in Student subclass
```

```
System.out.println(alice.getId());
System.out.println(alice.getGpa());
```

Οι διεπαφές (Interfaces) είναι αφηρημένοι τύποι που καθορίζουν ένα σύνολο αφηρημένων μεθόδων που πρέπει να υλοποιεί μια κλάση αν θέλει να συμμορφώνεται με τη συγκεκριμένη διεπαφή. Οι διεπαφές χρησιμοποιούνται για τον ορισμό συμβάσεων ή συμπεριφορών που πρέπει να ακολουθούν οι κλάσεις χωρίς να προσδιορίζεται ο τρόπος με τον οποίο το κάνουν. Μπορείτε να χρησιμοποιήσετε διεπαφές για να επιτύχετε αφαίρεση, πολυμορφισμό και πολλαπλή κληρονομικότητα στη Java. Μπορείτε να χρησιμοποιήσετε τη λέξη-κλειδί διεπαφή για να ορίσετε μια διεπαφή ακολουθούμενη από το όνομα και το σώμα της. Μπορείτε επίσης να χρησιμοποιήσετε τη λέξη-κλειδί implements για να δηλώσετε ότι μια κλάση υλοποιεί μία ή περισσότερες διεπαφές. *Για παράδειγμα:*

```
// define an interface named Animal that specifies two abstract
methods: makeSound() and move()

public interface Animal {
   public void makeSound(); / declare abstract method makeSound
without body
```

Πώς να χρησιμοποιήσετε διεπαφές στη Java

- ➔ Μια διεπαφή είναι ένα σχέδιο συμπεριφοράς που ορίζει ποιες μεθόδους πρέπει να υλοποιεί μια κλάση.
- ➔ Για να χρησιμοποιήσετε μια διεπαφή, γράφετε μια κλάση που υλοποιεί τη διεπαφή και παρέχει ένα σώμα μεθόδων για κάθε μία από τις μεθόδους που δηλώνονται στη διεπαφή.
- ➔ Μπορείτε να χρησιμοποιήσετε διεπαφές για να επιτύχετε ασφάλεια, αφαίρεση και πολλαπλή κληρονομικότητα στη Java.
- ➔ Μπορείτε επίσης να χρησιμοποιήσετε σταθερές, μεταβλητές, στατικές μεθόδους, προεπιλεγμένες μεθόδους και κενές διεπαφές σε μια διεπαφή.

Πώς να δημιουργήσετε και να υλοποιήσετε διεπαφές (διασυνδέσεις) στη Java

- ➔ Για να δηλώσετε μια διασύνδεση, χρησιμοποιήστε τη λέξη-κλειδί interface ακολουθούμενη από το όνομα της διασύνδεσης. Μπορείτε προαιρετικά να

καθορίσετε άλλες διασυνδέσεις που επεκτείνει αυτή η διασύνδεση χρησιμοποιώντας τη λέξη κλειδί extends.

➔ Για να δημιουργήσετε μια διασύνδεση, μπορείτε να δηλώσετε σταθερά πεδία, αφηρημένες μεθόδους, στατικές μεθόδους και προεπιλεγμένες μεθόδους μέσα σε αγκύλες.

➔ Για να υλοποιήσετε μια διασύνδεση, χρησιμοποιήστε τη λέξη-κλειδί implements ακολουθούμενη από το όνομα της διασύνδεσης. Πρέπει να παρέχετε ένα σώμα μεθόδου για κάθε μία από τις αφηρημένες μεθόδους που δηλώνονται στη διασύνδεση.

Ακολουθεί ένα μπλοκ κώδικα που απεικονίζει αυτά τα βήματα:

```
// Declare an interface named Animal
public interface Animal {
  // Declare a constant field
  public static final int LEGS = 4;
  // Declare an abstract method
  public void makeSound();
  // Declare a static method
  public static void sleep() {
    System.out.println("Zzz...");
  }
  // Declare a default method
  public default void eat() {
    System.out.println("Yum!");
  }
}

// Implement an interface named Animal
public class Dog implements Animal {
  // Provide a method body for makeSound()
  public void makeSound() {
    System.out.println("Woof!");
  }
}

// Create an object of the Dog class and call its methods
public class Main {
  public static void main(String[] args) {
    Dog d = new Dog();
    d.makeSound(); // Woof!
    d.eat(); // Yum!
    Animal.sleep(); // Zzz...
```

```
    System.out.println(Animal.LEGS); // 4
  }
}
```

Πώς να χρησιμοποιείτε τις εξαιρέσεις στη Java

- ➔ Μια εξαίρεση (Exception) είναι ένα ανεπιθύμητο ή απροσδόκητο γεγονός που συμβαίνει κατά τη διάρκεια της εκτέλεσης ενός προγράμματος και διακόπτει την κανονική ροή του.
- ➔ Για να χρησιμοποιήσετε εξαιρέσεις στη Java, πρέπει να χρησιμοποιήσετε μπλοκ try...catch που σας επιτρέπουν να χειρίζεστε διαφορετικούς τύπους εξαιρέσεων.
- ➔ Μπορείτε επίσης να δημιουργήσετε τις δικές σας προσαρμοσμένες εξαιρέσεις επεκτείνοντας τις κλάσεις Exception ή RuntimeException.

Ακολουθεί ένα μπλοκ κώδικα που απεικονίζει αυτά τα βήματα:

```
// Import java.util package
import java.util.*;

// Create an ArrayList of String
List<String> names = new ArrayList<>();
// Add some elements to it
names.add("Alice");
names.add("Bob");
names.add("Charlie");
names.add("David");
// Print the list
System.out.println(names); // [Alice, Bob, Charlie, David]

// Create a HashSet of Integer
Set<Integer> numbers = new HashSet<>();
// Add some elements to it
numbers.add(10);
numbers.add(20);
numbers.add(30);
numbers.add(40);
// Print the set
System.out.println(numbers); // [40, 10 , 30 , 20]

// Create a HashMap of String and Integer
Map<String,Integer> scores = new HashMap<>();
```

```
// Put some key-value pairs to it
scores.put("Alice",90);
scores.put("Bob",80);
scores.put("Charlie",70);
scores.put("David",60);
// Print the map
System.out.println(scores); // {Alice=90 , Bob=80 , Charlie=70 ,
David=60}
```

Πως να χρησιμοποιείτε generics στη Java

- ➔ Τα generics είναι ένα χαρακτηριστικό της Java που σας επιτρέπει να γράφετε κώδικα που μπορεί να λειτουργεί με διαφορετικούς τύπους δεδομένων χωρίς να επαναλαμβάνεστε.
- ➔ Τα generics χρησιμοποιούν παραμέτρους τύπου (όπως <T>) για να αντιπροσωπεύουν γενικούς τύπους που μπορούν να αντικατασταθούν από πραγματικούς τύπους κατά τη χρήση των generics.
- ➔ Τα generics μπορούν να χρησιμοποιηθούν με κλάσεις, διασυνδέσεις, μεθόδους και κατασκευαστές.
- ➔ Τα generics παρέχουν ασφάλεια τύπων, επαναχρησιμοποίηση κώδικα και πλεονεκτήματα απόδοσης.

Ακολουθεί ένα μπλοκ κώδικα που δείχνει πώς να χρησιμοποιείτε generics με μια κλάση:

```
// Define a generic class
class Box<T> {
  // Declare a generic instance variable
  private T item;
  // Define a constructor with a generic parameter
  public Box(T item) {
    this.item = item;
  }
  // Define a getter method for the generic variable
  public T getItem() {
    return item;
  }
}

// Create an object of Box class with Integer type
Box<Integer> intBox = new Box<>(10);
// Get the item from intBox
```

```
int num = intBox.getItem();
// Print the item
System.out.println(num); // 10

// Create an object of Box class with String type
Box<String> strBox = new Box<>("Hello");
// Get the item from strBox
String str = strBox.getItem();
// Print the item
System.out.println(str); // Hello
```

Πώς να χρησιμοποιήσετε επισημειώσεις στη Java

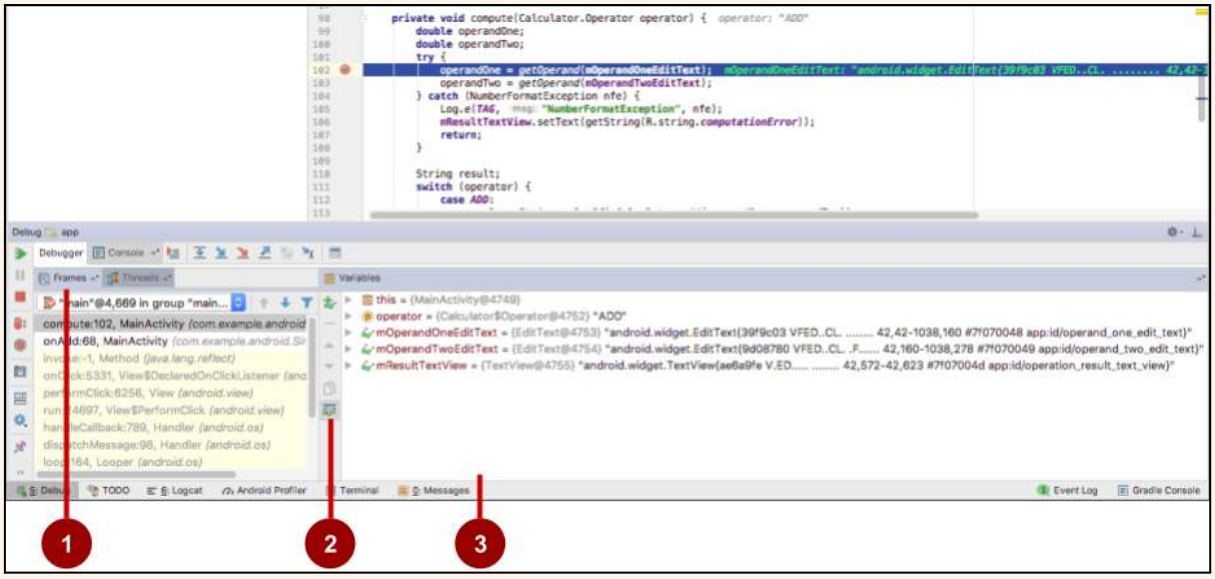
Οι σημειώσεις (Annotations) είναι μια μορφή μεταδεδομένων που παρέχουν δεδομένα σχετικά με ένα πρόγραμμα τα οποία δεν αποτελούν μέρος του ίδιου του προγράμματος. Δεν έχουν άμεση επίδραση στη λειτουργία του κώδικα που σχολιάζουν. Μπορούν να χρησιμοποιηθούν για διάφορους σκοπούς, όπως πληροφορίες για τον μεταγλωττιστή, δημιουργία τεκμηρίωσης, ανάλυση κώδικα, πλαίσια ελέγχου κ.λπ.

Για να χρησιμοποιήσετε τις επισημειώσεις στη Java, πρέπει να τις τοποθετήσετε πάνω από μια δήλωση (όπως κλάση, μέθοδος, πεδίο κ.λπ.) ή πριν από έναν τύπο (όπως ο τύπος επιστροφής μιας μεθόδου). Πρέπει επίσης να εισαγάγετε ή να ορίσετε τον τύπο του σχολίου. *Για παράδειγμα:*

```
import java.lang.annotation.Documented;

@Documented // this is an annotation
public @interface MyAnnotation { // this is an annotation type
  String value(); // this is an annotation element
}

@MyAnnotation("Hello") // this is an annotation with value
public class MyClass {
  @MyAnnotation("World") // this is another annotation with value
  public void myMethod() {
    // some code here
  }
}
```

Δοκιμή και αποσφαλμάτωση με το Android Studio

Πώς να χρησιμοποιήσετε τον εξομοιωτή και το logcat για τη δοκιμή και την αποσφαλμάτωση μιας εφαρμογής ιστού

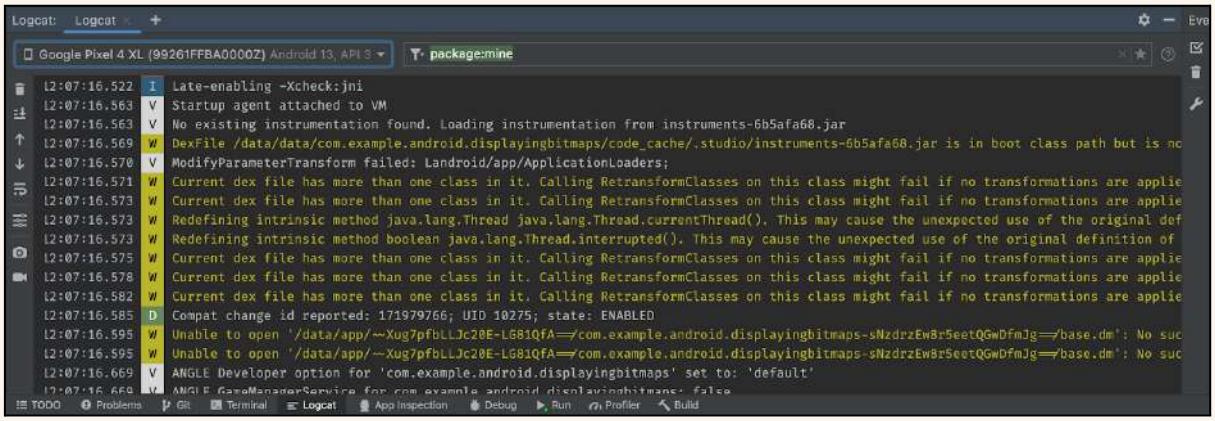


Για τη δοκιμή και την αποσφαλμάτωση (debugging) μιας εφαρμογής ιστού σε διαφορετικές συσκευές και διαμορφώσεις στο Android Studio, μπορείτε να χρησιμοποιήσετε τον εξομοιωτή (emulator) και τα εργαλεία logcat. Ο εξομοιωτής είναι μια εικονική συσκευή που προσομοιώνει μια πραγματική συσκευή Android στον υπολογιστή σας. Μπορείτε να δημιουργήσετε διαφορετικούς εξομοιωτές με διαφορετικές ρυθμίσεις, όπως μέγεθος οθόνης, επίπεδο API, χαρακτηριστικά υλικού κ.λπ. Το Logcat είναι ένα εργαλείο που εμφανίζει τα αρχεία καταγραφής από τη συσκευή ή τον εξομοιωτή σας σε πραγματικό χρόνο. Μπορείτε να χρησιμοποιήσετε το logcat για να δείτε μηνύματα που προσθέσατε στην εφαρμογή σας με την κλάση Log, μηνύματα από υπηρεσίες που εκτελούνται στο Android ή μηνύματα συστήματος.

Για να χρησιμοποιήσετε το logcat με έναν εξομοιωτή, πρέπει να εκτελέσετε το logcat ως εντολή adb ή απευθείας σε μια προτροπή κελύφους στον εξομοιωτή σας. Μπορείτε επίσης να χρησιμοποιήσετε το παράθυρο Logcat στο Android Studio για να προβάλετε και να φιλτράρετε τα αρχεία καταγραφής. Για να χρησιμοποιήσετε το logcat με μια εφαρμογή ιστού, πρέπει να ενεργοποιήσετε την αποσφαλμάτωση WebView στο αρχείο manifest της εφαρμογής σας.

Για να δημιουργήσετε έναν εξομοιωτή στο Android Studio, πρέπει να ακολουθήσετε τα παρακάτω βήματα:

- ➔ Ανοίξτε το Android Studio και κάντε κλικ στο Εργαλεία > Διαχείριση AVD
- ➔ Κάντε κλικ στο κουμπί Create Virtual Device (Δημιουργία εικονικής συσκευής)
- ➔ Επιλέξτε έναν ορισμό συσκευής και κάντε κλικ στο Επόμενο
- ➔ Επιλέξτε μια εικόνα συστήματος και κάντε κλικ στο Επόμενο
- ➔ Επανεξετάσετε τις ρυθμίσεις διαμόρφωσης και κάντε κλικ στο Finish



Για να ενεργοποιήσετε την αποσφαλμάτωση WebView στην εφαρμογή ιστού σας, πρέπει να προσθέσετε αυτή τη γραμμή στο αρχείο manifest της εφαρμογής σας:

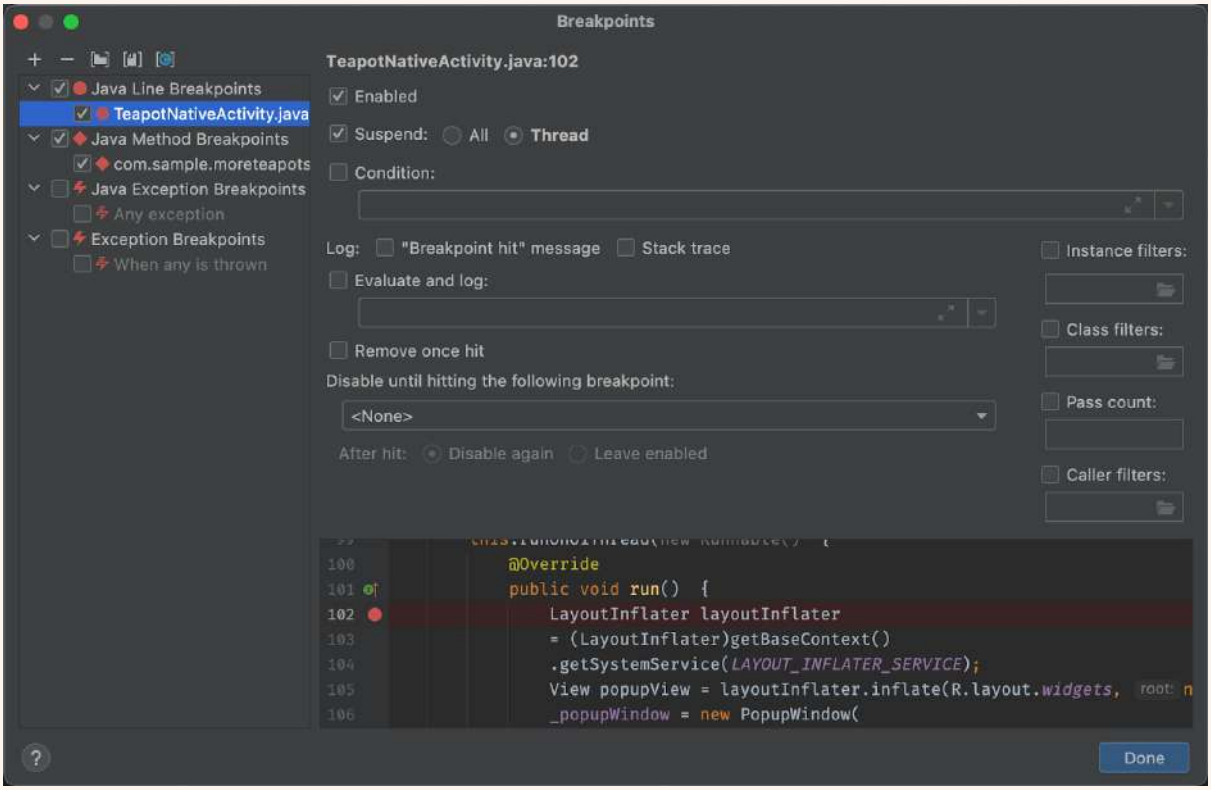
```
<application android:debuggable="true">
```

Και αυτή τη γραμμή στη μέθοδο onCreate της δραστηριότητάς σας:

```
WebView.setWebContentsDebuggingEnabled(true);
```

Αυτό θα σας επιτρέψει να χρησιμοποιήσετε τα εργαλεία Chrome DevTools για να επιθεωρήσετε και να αποσφαλματώσετε την εφαρμογή ιστού σας σε έναν εξομοιωτή ή μια συσκευή.

Πώς να χρησιμοποιήσετε τα σημεία διακοπής (Breakpoints)

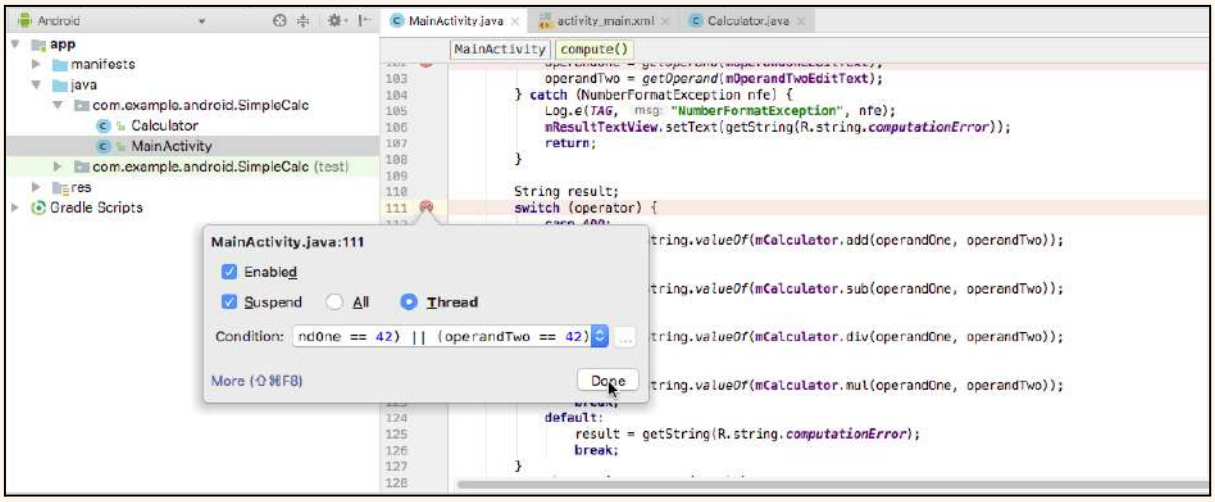


Τα σημεία διακοπής είναι ένα εργαλείο εντοπισμού σφαλμάτων που σας επιτρέπει να διακόψετε την εκτέλεση της εφαρμογής σας σε μια συγκεκριμένη γραμμή κώδικα. Μπορείτε να χρησιμοποιήσετε τα σημεία διακοπής για να επιθεωρήσετε την κατάσταση της εφαρμογής σας, να αξιολογήσετε εκφράσεις, να τροποποιήσετε μεταβλητές και να συνεχίσετε την εκτέλεση με διαφορετικές συνθήκες.

```
98      private void compute(Calculator.Operator operator) {
99          double operandOne;
100         double operandTwo;
101         try {
102             operandOne = getOperand(mOperandOneEditText);
103             operandTwo = getOperand(mOperandTwoEditText);
104         } catch (NumberFormatException nfe) {
105             Log.e(TAG,  msg: "NumberFormatException", nfe);
106             mResultTextView.setText(getString(R.string.computationError));
107             return;
108         }
109
110         String result;
111         switch (operator) {
```

Για να χρησιμοποιήσετε σημεία διακοπής στο Android Studio, πρέπει να ακολουθήσετε τα εξής βήματα:

- ➔ Ανοίξτε το έργο σας και εντοπίστε τον κώδικα όπου θέλετε να ορίσετε ένα σημείο διακοπής
- ➔ Κάντε κλικ στη στενή στήλη στα αριστερά του αριθμού γραμμής του κώδικα. Θα εμφανιστεί μια κόκκινη κουκκίδα που υποδεικνύει ότι έχει οριστεί ένα σημείο διακοπής
- ➔ Εκτελέστε την εφαρμογή σας σε κατάσταση εντοπισμού σφαλμάτων κάνοντας κλικ στο εικονίδιο Debug
- ➔ Όταν η εφαρμογή σας φτάσει στο σημείο διακοπής, θα κάνει παύση και θα σας εμφανίσει το παράθυρο αποσφαλμάτωσης με διάφορα παράθυρα και επιλογές
- ➔ Μπορείτε να χρησιμοποιήσετε τα κουμπιά στο επάνω μέρος του παραθύρου αποσφαλμάτωσης για να συνεχίσετε, να περάσετε, να μπείτε, να βγείτε ή να σταματήσετε την αποσφαλμάτωση
- ➔ Μπορείτε επίσης να κάνετε δεξί κλικ σε ένα σημείο διακοπής για να επεξεργαστείτε τις ιδιότητές του, όπως η συνθήκη, το μήνυμα καταγραφής, η πολιτική αναστολής κ.λπ.



Ακολουθεί ένα παράδειγμα χρήσης σημείων διακοπής στο Android Studio:

Ας υποθέσουμε ότι έχετε μια απλή εφαρμογή που εμφανίζει ένα μήνυμα όταν κάνετε κλικ σε ένα κουμπί. Ο κώδικας για το αρχείο MainActivity.kt μοιάζει ως εξής:

```
package com.example.debugging

import androidx.appcompat.app.AppCompatActivity
import android.os.Bundle
import android.widget.Button
```

```
import android.widget.TextView

class MainActivity : AppCompatActivity() {
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_main)

        val button = findViewById<Button>(R.id.button)
        val textView = findViewById<TextView>(R.id.textView)

        button.setOnClickListener {
            textView.text = "Hello World!"
        }
    }
}
```

Τώρα ας εισάγουμε ένα σφάλμα αλλάζοντας το κείμενο σε "Hello Bug!" αντί για "Hello World!". Για την αποσφαλμάτωση αυτού του σφάλματος, μπορούμε να θέσουμε ένα σημείο διακοπής στη γραμμή 17 όπου αναθέτουμε το κείμενο στο textView. Για να το κάνουμε αυτό, κάνουμε κλικ στη στήλη στα αριστερά της γραμμής 17 και βλέπουμε να εμφανίζεται μια κόκκινη κουκκίδα.

Στη συνέχεια, εκτελούμε την εφαρμογή μας σε κατάσταση εντοπισμού σφαλμάτων κάνοντας κλικ στο εικονίδιο Debug. Όταν η εφαρμογή μας φτάσει στη γραμμή 17, θα κάνει παύση και θα μας εμφανίσει το παράθυρο Debug με διάφορα παράθυρα και επιλογές.

Μπορούμε να δούμε ότι το textView.text είναι "Hello Bug!" το οποίο δεν είναι αυτό που θέλουμε. Μπορούμε επίσης να δούμε ότι το button.text είναι "Click Me" το οποίο είναι σωστό. Μπορούμε να χρησιμοποιήσουμε το παράθυρο Variables για να επιθεωρήσουμε και να τροποποιήσουμε αυτές τις τιμές.

Μπορούμε επίσης να χρησιμοποιήσουμε την επιλογή Evaluate Expression για να εκτελέσουμε οποιαδήποτε έκφραση στο πλαίσιο της εφαρμογής μας. Για παράδειγμα, μπορούμε να πληκτρολογήσουμε textView.text = "Hello World!" και να πατήσουμε Evaluate για να διορθώσουμε προσωρινά το σφάλμα μας.

Για να συνεχίσουμε την εκτέλεση της εφαρμογής μας, μπορούμε να κάνουμε κλικ στο εικονίδιο Συνέχιση. Θα δούμε ότι η εφαρμογή μας εμφανίζει τώρα το "Hello World!" όταν κάνουμε κλικ στο κουμπί.

Με αυτόν τον τρόπο μπορούμε να χρησιμοποιήσουμε σημεία διακοπής για την αποσφαλμάτωση της εφαρμογής μας στο Android Studio.

Πώς να παρακολουθήσετε μία έκφραση (Expression)

Μια έκφραση παρακολούθησης είναι μια έκφραση που θέλετε να αξιολογήσετε κατά την αποσφαλμάτωση της εφαρμογής σας. Μπορείτε να χρησιμοποιήσετε τις εκφράσεις παρακολούθησης για να παρακολουθείτε τις τιμές μεταβλητών ή εκφράσεων που δεν είναι ορατές στο παράθυρο Μεταβλητές.

Για να παρακολουθήσετε μια έκφραση στο Android Studio, πρέπει να ακολουθήσετε τα εξής βήματα:

- ➔ Εκτελέστε την εφαρμογή σας σε λειτουργία εντοπισμού σφαλμάτων και κάντε παύση σε ένα σημείο διακοπής
- ➔ Στο παράθυρο αποσφαλμάτωσης, επιλέξτε την καρτέλα Watches
- ➔ Κάντε κλικ στο εικονίδιο + για να προσθέσετε μια νέα έκφραση ρολογιού
- ➔ Εισάγετε την έκφραση που θέλετε να παρακολουθήσετε στο πεδίο κειμένου και πατήστε Enter
- ➔ Θα δείτε την τιμή της έκφρασής σας να εμφανίζεται από κάτω. Μπορείτε επίσης να επεξεργαστείτε ή να αφαιρέσετε την έκφραση παρακολούθησης κάνοντας δεξί κλικ πάνω της

Για να παρακολουθήσετε εκφράσεις στην αποσφαλμάτωση του Android Studio, πρέπει να χρησιμοποιήσετε σημεία διακοπής και ρολόγια. Τα σημεία διακοπής είναι δείκτες που υποδεικνύουν στον αποσφαλματωτή πού να διακόψει την εκτέλεση της εφαρμογής σας, ώστε να μπορείτε να επιθεωρήσετε την κατάστασή της. Τα watches είναι εκφράσεις που μπορείτε να αξιολογήσετε κατά την αποσφαλμάτωση και να δείτε τις τιμές τους να αλλάζουν καθώς προχωράτε στον κώδικά σας.

Για να προσθέσετε ένα σημείο διακοπής, κάντε κλικ στο αριστερό περιθώριο του επεξεργαστή κώδικα δίπλα στη γραμμή όπου θέλετε να διακόψετε την εφαρμογή σας. Θα εμφανιστεί μια κόκκινη κουκκίδα που υποδεικνύει ένα σημείο διακοπής. Για να προσθέσετε ένα ρολόι, κάντε δεξί κλικ σε μια μεταβλητή στο παράθυρο Μεταβλητές και επιλέξτε "Προσθήκη σε ρολόγια". Μπορείτε επίσης να πληκτρολογήσετε μια έκφραση στο παράθυρο Watches και να πατήσετε Enter.

Ακολουθεί ένα παράδειγμα του τρόπου παρακολούθησης εκφράσεων στην αποσφαλμάτωση του Android Studio:

```
fun main() {
    val numbers = listOf(1, 2, 3)
```

```
    var sum = 0
    for (n in numbers) {
        sum += n // Add a breakpoint here
    }
    println(sum)
}
```

Σε αυτό το απόσπασμα κώδικα, έχουμε προσθέσει ένα σημείο διακοπής στη γραμμή 5 όπου ενημερώνουμε τη μεταβλητή sum. Όταν τρέξουμε την εφαρμογή σε κατάσταση εντοπισμού σφαλμάτων, θα κάνει παύση σε αυτή τη γραμμή και θα μας δείξει τις τιμές των n και sum στο παράθυρο Variables. Μπορούμε επίσης να προσθέσουμε ρολόγια για εκφράσεις όπως `sum / n` ή `numbers.size` και να δούμε πώς αλλάζουν καθώς προχωράμε σε κάθε επανάληψη του βρόχου.

Πώς να περνάτε πάνω/μέσα/έξω από εντολές

Για να μεταβείτε σε/μέσα/έξω από εντολές στην αποσφαλμάτωση του Android Studio, πρέπει να χρησιμοποιήσετε τα κουμπιά στη γραμμή εργαλείων αποσφαλμάτωσης ή τις συντομεύσεις πληκτρολογίου. Αυτές οι εντολές σας επιτρέπουν να ελέγχετε τον τρόπο με τον οποίο η εκτέλεση της εφαρμογής σας διατρέχει τον κώδικά σας και να μεταπηδάτε σε μεθόδους ή να βγαίνετε από μεθόδους.

- ➔ Το Step Over εκτελεί την επόμενη γραμμή κώδικα της τρέχουσας κλάσης και μεθόδου, εκτελώντας όλες τις κλήσεις μεθόδων σε αυτή τη γραμμή και παραμένοντας στο ίδιο αρχείο. Για να χρησιμοποιήσετε αυτή την εντολή, κάντε κλικ στο εικονίδιο Step Over, επιλέξτε Εκτέλεση > Step Over ή πληκτρολογήστε F8.
- ➔ Η εντολή Step Into μεταπηδά στην εκτέλεση μιας κλήσης μεθόδου στην τρέχουσα γραμμή (σε αντίθεση με την απλή εκτέλεση αυτής της μεθόδου και την παραμονή στην ίδια γραμμή). Για να χρησιμοποιήσετε αυτήν την εντολή, κάντε κλικ στο εικονίδιο Step Into, επιλέξτε Run > Step Into ή πληκτρολογήστε F7.
- ➔ Η εντολή Step Out συνεχίζει την εκτέλεση μέχρι να φτάσει σε μια δήλωση επιστροφής σε μια κλήση μεθόδου στην οποία είχε προηγουμένως εισέλθει. Για να χρησιμοποιήσετε αυτήν την εντολή, κάντε κλικ στο εικονίδιο Step Out, επιλέξτε Run > Step Out ή πληκτρολογήστε Shift+F8.

Ακολουθεί ένα παράδειγμα χρήσης αυτών των εντολών στην αποσφαλμάτωση του Android Studio:

```
fun main() {
    val numbers = listOf(1, 2, 3)
    var sum = 0
    for (n in numbers) {
        sum += n // Add a breakpoint here
    }
    println(sum)
}

fun add(a: Int, b: Int): Int {
    return a + b // Add another breakpoint here
}
```

Σε αυτό το απόσπασμα κώδικα, έχουμε προσθέσει δύο σημεία διακοπής: ένα στη γραμμή 5, όπου ενημερώνουμε τη μεταβλητή sum χρησιμοποιώντας έναν βρόχο, και ένα άλλο στη γραμμή 11, όπου ορίζουμε μια απλή συνάρτηση add. Όταν τρέξουμε την εφαρμογή σε κατάσταση εντοπισμού σφαλμάτων, θα κάνει παύση στη γραμμή 5 και θα μας δείξει τις τιμές των n και sum στο παράθυρο Variables.

Αν θέλουμε να δούμε πώς λειτουργεί εσωτερικά η συνάρτηση add, μπορούμε να αντικαταστήσουμε τη γραμμή 5 με `sum = add(sum,n)` και στη συνέχεια να χρησιμοποιήσουμε το **Step Into** για να μεταβούμε σε αυτή τη συνάρτηση. Θα δούμε ότι σταματάει στη γραμμή 11 και μας δείχνει τις τιμές των a και b.

Αν θέλουμε να παρακάμψουμε τη συνάρτηση add και να δούμε απλώς το αποτέλεσμά της χωρίς να μεταπηδήσουμε σε αυτήν, μπορούμε να χρησιμοποιήσουμε **Step Over**. Θα δούμε ότι εκτελεί τη γραμμή 5 χωρίς να κάνει παύση στη γραμμή 11 και ενημερώνει το άθροισμα αναλόγως.

Αν θέλουμε να βγούμε από τη συνάρτησή μας add αφού έχουμε κάνει άλμα σε αυτήν και να συνεχίσουμε την εκτέλεση μέχρι να φτάσει σε μια εντολή return (γραμμή 12), μπορούμε να χρησιμοποιήσουμε το **Step Out**. Θα δούμε ότι επιστρέφει στη γραμμή 5 με ενημερωμένο το άθροισμα.

Πώς να εκτιμήσετε εκφράσεις

Για να αξιολογήσετε εκφράσεις στην αποσφαλμάτωση του Android Studio, πρέπει να χρησιμοποιήσετε το παράθυρο Evaluate Expression. Αυτό το παράθυρο σας επιτρέπει να επικυρώνετε διάφορες εκφράσεις και την τρέχουσα κατάσταση των μεταβλητών σε οποιοδήποτε σημείο διακοπής.

Για να ανοίξετε το παράθυρο Evaluate Expression (Αξιολόγηση έκφρασης), κάντε δεξί κλικ σε μια μεταβλητή ή έκφραση στον κώδικά σας και επιλέξτε Evaluate Expression (Αξιολόγηση έκφρασης). Εναλλακτικά, μπορείτε να πατήσετε Alt+F8 στα Windows ή Option+F8 στον Mac. Μπορείτε επίσης να αποκτήσετε πρόσβαση σε αυτό το παράθυρο από τη γραμμή εργαλείων αποσφαλμάτωσης κάνοντας κλικ στο εικονίδιο [evaluate expression icon] (https://developer.android.com/images/tools/debugging/evaluate-expression.png).

Στο παράθυρο Evaluate Expression (Αξιολόγηση έκφρασης), μπορείτε να πληκτρολογήσετε οποιαδήποτε έκφραση που είναι έγκυρη στο τρέχον πλαίσιο και να πατήσετε Enter για να δείτε την τιμή της. Μπορείτε επίσης να τροποποιήσετε την τιμή των μεταβλητών εκχωρώντας τους νέες τιμές. Για παράδειγμα, αν έχετε μια μεταβλητή με όνομα `name` που περιέχει μια τιμή συμβολοσειράς `"Alice"`, μπορείτε να πληκτρολογήσετε `name = "Bob"` και να πατήσετε Enter για να αλλάξετε την τιμή της.

Ακολουθεί ένα παράδειγμα αξιολόγησης εκφράσεων στην αποσφαλμάτωση του Android Studio:

```
fun main() {
    val numbers = listOf(1, 2, 3)
    var sum = 0
    for (n in numbers) {
        sum += n // Add a breakpoint here
    }
    println(sum)
}
```

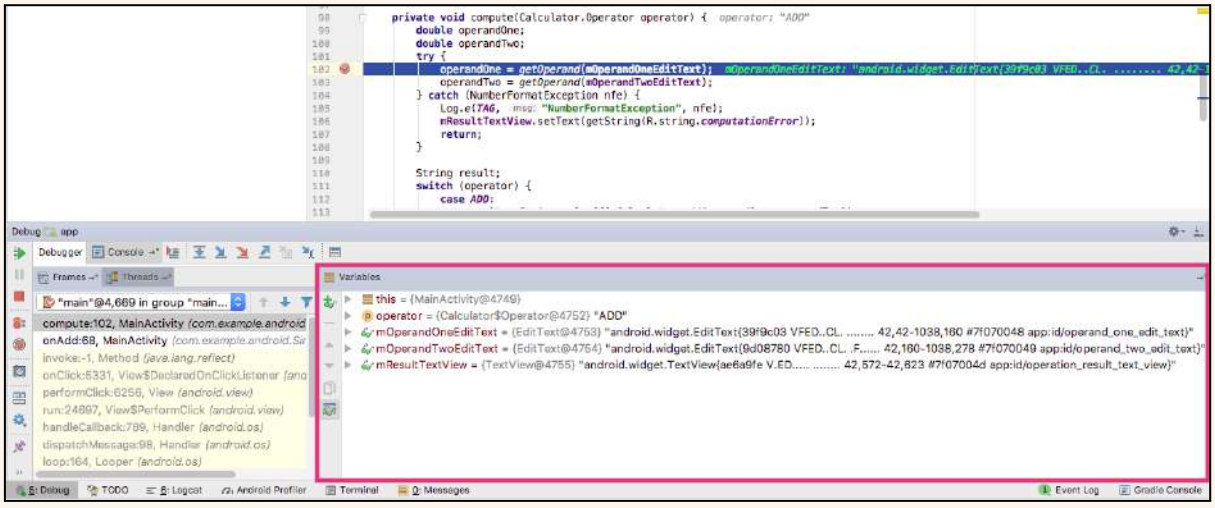
Σε αυτό το απόσπασμα κώδικα, έχουμε προσθέσει ένα σημείο διακοπής στη γραμμή 5, όπου ενημερώνουμε τη μεταβλητή sum χρησιμοποιώντας έναν βρόχο. Όταν τρέξουμε την εφαρμογή σε κατάσταση εντοπισμού σφαλμάτων, θα κάνει παύση σε αυτή τη γραμμή και θα μας δείξει τις τιμές των n και sum στο παράθυρο Variables.

Αν θέλουμε να δούμε ποια θα ήταν η τιμή του αθροίσματος αν προσθέταμε έναν άλλο αριθμό σε αυτό χωρίς να αλλάξουμε την πραγματική του τιμή, μπορούμε να ανοίξουμε το παράθυρο Evaluate Expression και να πληκτρολογήσουμε `sum + 4` και να πατήσουμε Enter. Θα δούμε ότι επιστρέφει `10` όπως αναμενόταν.

Αν θέλουμε να αλλάξουμε μόνιμα την τιμή του αθροίσματος κατά την αποσφαλμάτωση, μπορούμε να πληκτρολογήσουμε `sum = 100` και να πατήσουμε Enter. Θα δούμε ότι ενημερώνει το άθροισμα τόσο στο παράθυρο Evaluate Expression (Εκτίμηση έκφρασης) όσο και στο παράθυρο Variables (Μεταβλητές).

Πώς να επιθεωρήσετε μεταβλητές/τιμές/μνήμη/νήματα/στοίβα κλήσεων

Για να επιθεωρήσετε μεταβλητές/τιμές/μνήμη/νηματία/στοίβα κλήσεων στην αποσφαλμάτωση του Android Studio, πρέπει να χρησιμοποιήσετε το παράθυρο αποσφαλμάτωσης. Αυτό το παράθυρο παρέχει διάφορες καρτέλες και εργαλεία που σας βοηθούν να παρακολουθείτε και να χειρίζεστε την εκτέλεση της εφαρμογής σας κατά την αποσφαλμάτωση.



Για να ανοίξετε το παράθυρο εντοπισμού σφαλμάτων, κάντε κλικ στο [εικονίδιο εντοπισμού σφαλμάτων](https://developer.android.com/images/tools/debugging/debug.png) στη γραμμή εργαλείων ή πατήστε Shift+F9. Εναλλακτικά, μπορείτε να επιλέξετε Εκτέλεση > Αποσφαλμάτωση από τη γραμμή μενού.

Στο παράθυρο Debug, μπορείτε να χρησιμοποιήσετε τις ακόλουθες καρτέλες και εργαλεία για να επιθεωρήσετε διάφορες πτυχές της εφαρμογής σας:

- **Μεταβλητές (Variables)**: Αυτή η καρτέλα σας δείχνει τις τιμές των μεταβλητών στην τρέχουσα εμβέλειά σας. Μπορείτε να επεκτείνετε ή να συμπτύξετε τις μεταβλητές για να δείτε τα πεδία ή τα στοιχεία τους. Μπορείτε επίσης να τροποποιήσετε τις τιμές τους κάνοντας διπλό κλικ πάνω τους και πληκτρολογώντας μια νέα τιμή.
- **Μνήμη (Memory)**: Αυτή η καρτέλα σας δείχνει πόση μνήμη χρησιμοποιεί η εφαρμογή σας και πώς αυτή μεταβάλλεται με την πάροδο του χρόνου. Μπορείτε επίσης να εκτελέσετε συλλογή σκουπιδιών, να καταγράψετε απορρίψεις σωρού και να αναλύσετε διαρροές μνήμης χρησιμοποιώντας αυτή την καρτέλα.
- **Νήματα (Threads)**: Αυτή η καρτέλα σας δείχνει όλα τα νήματα που εκτελούνται στην εφαρμογή σας. Μπορείτε να εναλλάσσεστε μεταξύ των νημάτων κάνοντας κλικ σε αυτά. Μπορείτε επίσης να αναστείλετε ή να συνεχίσετε τα νήματα χρησιμοποιώντας τα κουμπιά αυτής της καρτέλας.
- **Στοίβα κλήσεων (Call Stack)**: Αυτό το εργαλείο σας δείχνει την ακολουθία των κλήσεων μεθόδων που οδήγησαν στο τρέχον σημείο διακοπής σας. Μπορείτε να περιηγηθείτε στη στοίβα κλήσεων κάνοντας κλικ σε κάθε όνομα μεθόδου. Μπορείτε επίσης να μεταβείτε σε μια συγκεκριμένη γραμμή κώδικα κάνοντας διπλό κλικ πάνω της.

Ακολουθεί ένα παράδειγμα για τον τρόπο επιθεώρησης των μεταβλητών/τιμών/μνήμης/νημάτων/στοίβας κλήσεων στο Android Studio debugging:

```
fun main() {
    val numbers = listOf(1, 2, 3)
    var sum = 0
    for (n in numbers) {
        sum += n // Add a breakpoint here
    }
    println(sum)
}

fun add(a: Int, b: Int): Int {
    return a + b // Add another breakpoint here
}
```

Σε αυτό το απόσπασμα κώδικα, έχουμε προσθέσει δύο σημεία διακοπής: ένα στη γραμμή 5, όπου ενημερώνουμε τη μεταβλητή sum χρησιμοποιώντας έναν βρόχο, και ένα άλλο στη γραμμή 11, όπου ορίζουμε μια απλή συνάρτηση add. Όταν τρέξουμε την εφαρμογή σε κατάσταση εντοπισμού σφαλμάτων, θα κάνει παύση στη γραμμή 5 και θα μας εμφανίσει τις ακόλουθες πληροφορίες στο παράθυρο Debug:

- **Μεταβλητές (Variables)**: Μπορούμε να δούμε ότι το n έχει την τιμή 1 και το άθροισμα έχει την τιμή 0 σε αυτό το σημείο. Μπορούμε επίσης να δούμε ότι οι αριθμοί είναι μια λίστα με τρία στοιχεία.

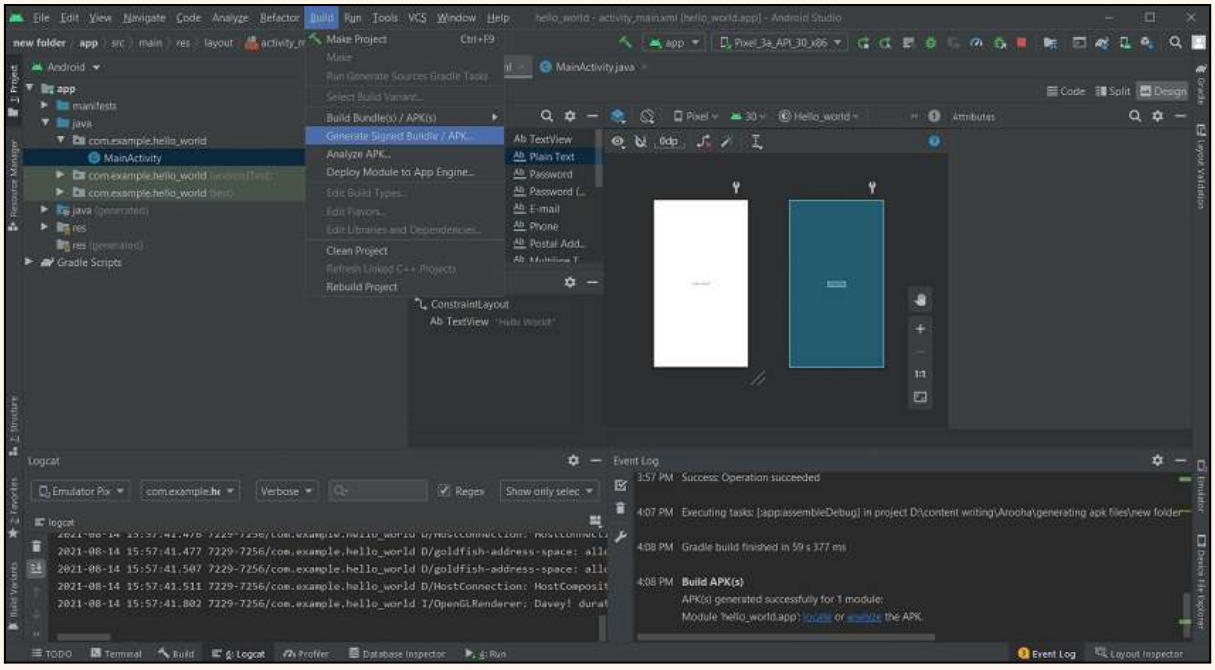
- **Μνήμη (Memory)**: Μπορούμε να δούμε ότι η εφαρμογή μας χρησιμοποιεί περίπου 10 MB μνήμης σε αυτό το σημείο. Μπορούμε επίσης να δούμε ένα γράφημα που δείχνει πώς αλλάζει η χρήση μνήμης με την πάροδο του χρόνου.
- **Νήματα (Threads)**: Μπορούμε να δούμε ότι η εφαρμογή μας έχει ένα νήμα που ονομάζεται main και το οποίο έχει ανασταλεί αυτή τη στιγμή στη γραμμή 5.
- **Στοίβα κλήσεων (Call Stack)**: Μπορούμε να δούμε ότι η εφαρμογή μας έχει καλέσει την main() η οποία έχει καλέσει την for (n σε αριθμούς) η οποία έχει φτάσει στη γραμμή 5.

Αν θέλουμε να δούμε πώς λειτουργεί εσωτερικά η συνάρτηση add, μπορούμε να αντικαταστήσουμε τη γραμμή 5 με `sum = add(sum,n)` και στη συνέχεια να χρησιμοποιήσουμε το **Step Into** για να μεταβούμε σε αυτή τη συνάρτηση. Θα δούμε ότι κάνει παύση στη γραμμή 11 και μας εμφανίζει διαφορετικές πληροφορίες στο παράθυρο αποσφαλμάτωσης:

- **Μεταβλητές**: Μπορούμε να δούμε ότι το a έχει την τιμή 0 και το b έχει την τιμή 1 σε αυτό το σημείο. Μπορούμε επίσης να δούμε ότι οι sum και n είναι εκτός πεδίου εφαρμογής τώρα.
- **Μνήμη**: Μπορούμε να δούμε ότι η εφαρμογή μας εξακολουθεί να χρησιμοποιεί περίπου 10 MB μνήμης σε αυτό το σημείο.
- **Νήματα**: Μπορούμε να δούμε ότι η εφαρμογή μας εξακολουθεί να έχει ένα νήμα που ονομάζεται main και το οποίο τώρα έχει ανασταλεί στη γραμμή 11.
- **Στοίβα κλήσεων**: Μπορούμε να δούμε ότι η εφαρμογή μας έχει καλέσει την main() η οποία έχει καλέσει την for (n σε αριθμούς) η οποία έχει καλέσει την add(a,b) η οποία έχει φτάσει στη γραμμή 11.

Ανάπτυξη και δημοσίευση μιας web εφαρμογής

Πώς να δημιουργήσετε ένα υπογεγραμμένο αρχείο APK για μια διαδικτυακή εφαρμογή

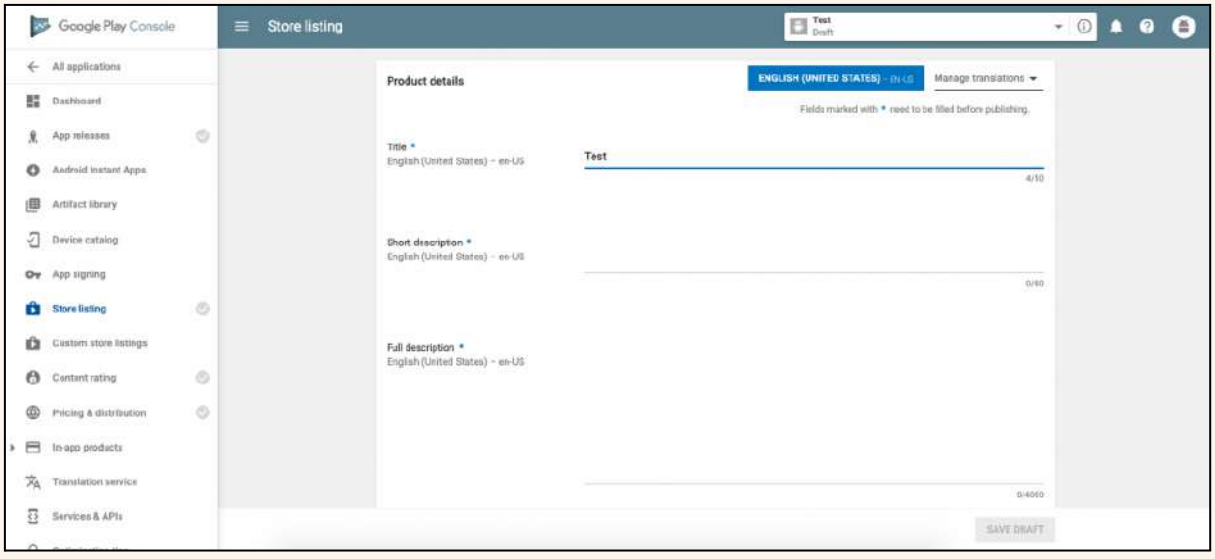


Για να δημιουργήσετε ένα υπογεγραμμένο αρχείο APK για μια διαδικτυακή εφαρμογή χρησιμοποιώντας το Android Studio, πρέπει να ακολουθήσετε τα παρακάτω βήματα:

- ➔ Βεβαιωθείτε ότι η διαδικτυακή σας εφαρμογή είναι έτοιμη για κυκλοφορία και δεν έχει σφάλματα ή προειδοποιήσεις.
- ➔ Μεταβείτε στο μενού του Android Studio και επιλέξτε Build > Generate Signed Bundle/APK.
- ➔ Στο πλαίσιο διαλόγου Generate Signed Bundle/APK (Δημιουργία υπογεγραμμένου πακέτου/APK), επιλέξτε APK και κάντε κλικ στο κουμπί Next (Επόμενο).
- ➔ Στο επόμενο παράθυρο, πρέπει να δώσετε τις πληροφορίες για το κλειδί υπογραφής της εφαρμογής σας. Εάν έχετε ήδη ένα υπάρχον αρχείο αποθήκευσης κλειδιών, μπορείτε να περιηγηθείτε σε αυτό και να εισαγάγετε τον κωδικό πρόσβασης και το ψευδώνυμό του. Αν δεν έχετε, μπορείτε να δημιουργήσετε ένα νέο κάνοντας κλικ στο κουμπί Δημιουργία νέου.... Πρέπει να εισαγάγετε λεπτομέρειες, όπως τη διαδρομή αποθήκευσης κλειδιών, τον κωδικό πρόσβασης αποθήκευσης κλειδιών, το ψευδώνυμο κλειδιού, τον κωδικό πρόσβασης κλειδιού, την περίοδο ισχύος του πιστοποιητικού και τις πληροφορίες του πιστοποιητικού. Κάντε κλικ στο κουμπί OK όταν ολοκληρώσετε.

- ➔ Στο επόμενο παράθυρο, πρέπει να επιλέξετε τον τύπο κατασκευής ως έκδοση και να επιλέξετε ένα φάκελο προορισμού για το υπογεγραμμένο αρχείο APK. Μπορείτε επίσης να ενεργοποιήσετε τις επιλογές V1 (Υπογραφή Jar) και V2 (Πλήρης υπογραφή APK) αν θέλετε[5]. Κάντε κλικ στο κουμπί Finish (Τέλος) όταν τελειώσετε.
- ➔ Περιμένετε μέχρι το Android Studio να δημιουργήσει το υπογεγραμμένο αρχείο APK σας με επιτυχία. Θα δείτε ένα μήνυμα όπως "APK(s) generated successfully" στο κάτω μέρος του Android Studio.
- ➔ Μπορείτε να βρείτε το υπογεγραμμένο αρχείο APK στον φάκελο προορισμού που καθορίσατε νωρίτερα. Μπορείτε επίσης να το εντοπίσετε κάνοντας κλικ στην επιλογή Εντοπισμός στο πλαίσιο του μηνύματος ή κάνοντας κλικ στο κουμπί Εμφάνιση στην Εξερεύνηση/Αναζήτηση στο Finder/Εμφάνιση στα Αρχεία στη δεξιά πλευρά του Android Studio.

Πώς να ανεβάσετε το αρχείο APK στο Google Play Store ή σε άλλες πλατφόρμες για διανομή



Για να ανεβάσετε το αρχείο APK στο Google Play Store για διανομή, πρέπει να ακολουθήσετε τα παρακάτω βήματα:

- ➔ Δημιουργήστε έναν λογαριασμό Google Play Developer αν δεν έχετε ήδη. Πρέπει να πληρώσετε ένα εφάπαξ τέλος εγγραφής ύψους 25 δολαρίων για να αποκτήσετε πρόσβαση στην κονσόλα Play.
- ➔ Συνδέστε το λογαριασμό προγραμματιστή σας με έναν εμπορικό λογαριασμό Google Wallet, αν θέλετε να πουλήσετε την εφαρμογή σας ή να προσφέρετε αγορές εντός της εφαρμογής.

- ➔ Δημιουργήστε μια καταχώρηση εφαρμογής στην Κονσόλα Play κάνοντας κλικ στο κουμπί Δημιουργία εφαρμογής και εισάγοντας το όνομα και την προεπιλεγμένη γλώσσα της εφαρμογής σας.
- ➔ Συμπληρώστε τα στοιχεία της καταχώρισης στο κατάστημα εφαρμογών, όπως τίτλο, περιγραφή, στιγμιότυπα οθόνης, εικονίδιο, κατηγορία κ.λπ. Μπορείτε επίσης να προσθέσετε μεταφράσεις για άλλες γλώσσες αν θέλετε.
- ➔ Ανεβάστε τις δέσμες εφαρμογών ή τα αρχεία APK στο Google Play κάνοντας κλικ στο κουμπί Εξερεύνηση δέσμης εφαρμογών > Ανέβασμα δέσμης εφαρμογών / APK. Μπορείτε επίσης να χρησιμοποιήσετε την επιλογή Build > Generate Signed Bundle/APK του Android Studio για να ανεβάσετε την εφαρμογή σας απευθείας από εκεί. Σημειώστε ότι από τον Αύγουστο του 2021, το Google Play δέχεται μόνο Android App Bundles (.aab files) αντί για αρχεία APK (.apk files) για νέες εφαρμογές. Εάν είχατε μια υπάρχουσα εφαρμογή που χρησιμοποιούσε αρχεία APK, θα μπορούσατε ακόμα να την ενημερώσετε με αρχεία APK μέχρι τον Νοέμβριο του 2021.
- ➔ Παρέχετε μια βαθμολογία περιεχομένου για την εφαρμογή σας, συμπληρώνοντας ένα ερωτηματολόγιο σχετικά με το περιεχόμενο και τα χαρακτηριστικά της. Αυτό βοηθά το Google Play να καθορίσει τις κατάλληλες ηλικιακές ομάδες για την εφαρμογή σας.
- ➔ Ρυθμίστε τις επιλογές τιμολόγησης και διανομής της εφαρμογής σας, όπως χώρες/περιοχές, κατάσταση δωρεάν/πληρωμής, συμβατότητα συσκευών κ.λπ. Μπορείτε επίσης να δηλώσετε συμμετοχή σε διάφορα προγράμματα και λειτουργίες, όπως το Google Play Instant, το Wear OS by Google κ.λπ.
- ➔ Δημοσιεύστε την εφαρμογή σας κάνοντας κλικ στο κουμπί Αναθεώρηση > Έναρξη της μετάβασης στην παραγωγή. Η εφαρμογή σας θα εξεταστεί από το Google Play πριν γίνει διαθέσιμη στους χρήστες. Μπορείτε επίσης να χρησιμοποιήσετε διαδρομές δοκιμών, όπως εσωτερική δοκιμή, κλειστή δοκιμή ή ανοικτή δοκιμή, για να δοκιμάσετε την εφαρμογή σας με περιορισμένο αριθμό χρηστών πριν τη δημοσιεύσετε δημοσίως.

Για να μεταφορτώσετε το αρχείο APK σε άλλες πλατφόρμες για διανομή (όπως το Amazon Appstore ή το Samsung Galaxy Store), πρέπει να ακολουθήσετε τις συγκεκριμένες οδηγίες και απαιτήσεις τους, οι οποίες ενδέχεται να διαφέρουν από το Google Play Store. Μπορείτε να βρείτε περισσότερες πληροφορίες στις αντίστοιχες ιστοσελίδες τους.

Συμπέρασμα & παραπομπές

Ανακεφαλαίωση των όσων μάθαμε σε αυτό το ηλεκτρονικό βιβλίο

1. Τι είναι το Android Studio και γιατί είναι ένα ισχυρό εργαλείο για την ανάπτυξη διαδικτυακών εφαρμογών Java
2. Παραδείγματα εφαρμογών ιστού που μπορούν να κατασκευαστούν με το Android Studio
3. Προαπαιτούμενα για την παρακολούθηση αυτού του ηλεκτρονικού βιβλίου
4. Πώς να ξεκινήσετε ένα νέο έργο Android Studio
5. Η δομή ενός έργου Android
6. Πώς να χρησιμοποιείτε τον επεξεργαστή διάταξης και τα widgets με drag-and-drop
7. Πώς να χρησιμοποιείτε χαρακτηριστικά, στυλ, θέματα και πόρους για να προσαρμόσετε την εμφάνιση και τη συμπεριφορά των widgets
8. Πώς να χρησιμοποιείτε τον επεξεργαστή κώδικα και να γράφετε κώδικα Java
9. Πώς να χρησιμοποιείτε μεταβλητές, τύπους δεδομένων, τελεστές, δομές ελέγχου, μεθόδους, κλάσεις, αντικείμενα, κληρονομικότητα, διασυνδέσεις, εξαιρέσεις, συλλογές, γενίκευση και επισημειώσεις στη Java
10. Πώς να χρησιμοποιείτε τον εξομοιωτή και το logcat για τη δοκιμή και την αποσφαλμάτωση μιας διαδικτυακής εφαρμογής
11. Πώς να χρησιμοποιείτε breakpoints, watch expressions, step over/into/out commands, evaluate expressions, inspect variables/values/memory/threads/call stack κ.λπ.
12. Πώς να δημιουργήσετε ένα υπογεγραμμένο αρχείο APK για μια διαδικτυακή εφαρμογή
13. Πώς να μεταφορτώσετε το αρχείο APK στο Google Play Store ή σε άλλες πλατφόρμες για διανομή

Συμβουλές και πόροι για περαιτέρω εκμάθηση

Ακολουθούν ορισμένες συμβουλές, σύνδεσμοι και πηγές για περαιτέρω μάθηση σχετικά με το θέμα "Πώς να δημιουργήσετε διαδικτυακές εφαρμογές Java με το Android Studio":

➔ Μπορείτε να ξεκινήσετε ακολουθώντας αυτό το **codelab** που σας καθοδηγεί στη δημιουργία μιας απλής εφαρμογής στο Android Studio χρησιμοποιώντας Java. Καλύπτει θέματα όπως η δημιουργία ενός νέου έργου, ο σχεδιασμός μιας διεπαφής χρήστη, η προσθήκη διαδραστικότητας και η δοκιμή της εφαρμογής σας σε έναν εξομοιωτή ή μια συσκευή.

- ➔ Μπορείτε επίσης να παρακολουθήσετε αυτό το **μάθημα** που σας διδάσκει τα βασικά για τη δημιουργία εφαρμογών με το Jetpack Compose, τη σύγχρονη εργαλειοθήκη του Android για την ανάπτυξη διεπαφών χρήστη. Θα μάθετε τα βασικά στοιχεία της γλώσσας προγραμματισμού Kotlin και τις βασικές αρχές ανάπτυξης εφαρμογών.
- ➔ Μπορείτε επίσης να δείτε αυτό το άρθρο του **wikiHow** που σας δείχνει πώς να δημιουργήσετε μια εφαρμογή με το Android Studio σε 8 βήματα. Καλύπτει θέματα όπως η επιλογή μιας δραστηριότητας, η διαμόρφωση των ρυθμίσεων του έργου σας, η προσθήκη κώδικα και η εκτέλεση της εφαρμογής σας.

Ο κώδικας Kotlin και τα πλεονεκτήματά του

Ο κώδικας Kotlin είναι μια συνοπτική και εκφραστική **γλώσσα προγραμματισμού** που είναι 100% συμβατή με τη Java και χρησιμοποιείται ευρέως για την ανάπτυξη Android. Επιτρέπει στους προγραμματιστές να γράφουν λιγότερο boilerplate κώδικα και περισσότερο κώδικα με ασφάλεια τύπου. Η Kotlin υποστηρίζεται πλήρως από το Android Studio, το επίσημο IDE για την ανάπτυξη Android. Αυτό σημαίνει ότι οι προγραμματιστές μπορούν να αποκτήσουν συμπλήρωση κώδικα, έλεγχο τύπου, αποσφαλμάτωση, δοκιμές και άλλα χαρακτηριστικά που τους βοηθούν να γράφουν κώδικα Kotlin αποτελεσματικά. Η Kotlin συνεργάζεται επίσης απρόσκοπτα με κώδικα και βιβλιοθήκες Java, ώστε οι προγραμματιστές να μπορούν να χρησιμοποιούν υπάρχοντες πόρους και APIs Java από την Kotlin.

Μερικά από τα πλεονεκτήματα της χρήσης της Kotlin έναντι της Java είναι τα εξής:

- ➔ Η Kotlin διαθέτει χαρακτηριστικά *null safety* που αποτρέπουν τις εξαιρέσεις null pointer, οι οποίες αποτελούν κοινή πηγή σφαλμάτων στη Java.
- ➔ Η Kotlin είναι *πιο συνοπτική* από τη Java, πράγμα που σημαίνει ότι απαιτεί λιγότερο κώδικα για την επίτευξη της ίδιας λειτουργικότητας και μειώνει το boilerplate κώδικα. Αυτό καθιστά τον κώδικα πιο ευανάγνωστο, συντηρήσιμο και λιγότερο επιρρεπή σε σφάλματα.
- ➔ Η Kotlin είναι *πλήρως διαλειτουργική* με τη Java, πράγμα που σημαίνει ότι μπορεί να χρησιμοποιήσει τις υπάρχουσες βιβλιοθήκες, πλαίσια και εργαλεία της Java χωρίς κανένα πρόβλημα. Επιτρέπει επίσης στους προγραμματιστές να αναμειγνύουν κώδικα Kotlin και Java στο ίδιο έργο χωρίς προβλήματα.
- ➔ Η Kotlin διαθέτει ορισμένα *σύγχρονα χαρακτηριστικά της γλώσσας* που δεν διαθέτει η Java, όπως coroutines, κλάσεις δεδομένων, συναρτήσεις επέκτασης, lambdas, έξυπνα casts και πολλά άλλα. Αυτά τα χαρακτηριστικά καθιστούν την Kotlin πιο εκφραστική και ισχυρή από τη Java.

Ακολουθεί ένα απλό παράδειγμα κώδικα Kotlin που εκτυπώνει "Hello World" στην κονσόλα:

```
fun main() {
    println("Hello World")
}
```

Ακολουθεί ένα άλλο παράδειγμα κώδικα Kotlin που ορίζει μια κλάση δεδομένων και δημιουργεί μια εμφάνισή της:

```
data class Person(val name: String, val age: Int)

fun main() {
    val person = Person("Alice", 25)
    println(person.name) // prints Alice
    println(person.age) // prints 25
}
```

Ακολουθεί ένα άλλο παράδειγμα κώδικα Kotlin που χρησιμοποιεί μια coroutine για την εκτέλεση μιας ασύγχρονης εργασίας:

```
import kotlinx.coroutines.*

fun main() {
    GlobalScope.launch { // launch a new coroutine in background
and continue
        delay(1000L) // non-blocking delay for 1 second (default
time unit is ms)
        println("World!") // print after delay
    }
    println("Hello") // main thread continues while coroutine is
delayed
    Thread.sleep(2000L) // block main thread for 2 seconds to keep
JVM alive
}
```

Ακολουθεί ένα παράδειγμα χρησιμοποιώντας κώδικα Kotlin που δημιουργεί μια κλάση με 3 χαρακτηριστικά και 2 μεθόδους καθώς και μια υποκλάση με 1 χαρακτηριστικό και 1 μέθοδο και τέλος δημιουργεί 1 αντικείμενο για κάθε κλάση:

```
// Define a class named Animal with 3 attributes and 2 methods
```

```
class Animal(val name: String, val color: String, var age: Int) {
    // A method to make a sound
    fun makeSound() {
        println("$name makes a sound.")
    }
    // A method to increase the age by one year
    fun growOlder() {
        age++
        println("$name is now $age years old.")
    }
}

// Define a sub class named Dog that inherits from Animal and has
1 additional attribute and 1 additional method
class Dog(name: String, color: String, age: Int, val breed:
String) : Animal(name, color, age) {
    // A method to fetch a ball
    fun fetchBall() {
        println("$name fetches a ball.")
    }
}

// Create an object of Animal class
val lion = Animal("Leo", "golden", 5)
// Call the methods of Animal class on lion object
lion.makeSound()
lion.growOlder()

// Create an object of Dog class
val dog = Dog("Max", "brown", 3, "Labrador")
// Call the methods of Animal and Dog classes on dog object
dog.makeSound()
dog.growOlder()
dog.fetchBall()
```

Ένα παράδειγμα έργου (project) από την αρχή

Εδώ είναι ένα απλό (μέτριας δυσκολίας) νέο έργο με όλο τον απαραίτητο κώδικα και επεξηγήσεις.

Δημιουργήστε μια εφαρμογή Android που μετατρέπει οποιοδήποτε είδος τιμής ενός αριθμητικού συστήματος σε μια άλλη ισοδύναμη τιμή αριθμητικού συστήματος (Numeral Systems Converter).

Βήμα 1: Δημιουργήστε ένα νέο έργο Android Studio

Δημιουργήστε ένα νέο έργο Android Studio επιλέγοντας "Start a new Android Studio project" στην οθόνη καλωσορίσματος ή πηγαίνοντας στο File > New > New Project. Επιλέξτε Empty Activity (κενή δραστηριότητα) ως πρότυπο (προεπιλεγμένο).

Βήμα 2: Ρύθμιση της διεπαφής χρήστη

Στο φάκελο "res" του έργου σας, δημιουργήστε ένα νέο αρχείο διάταξης με όνομα "activity_intro.xml". Σε αυτό το αρχείο, δημιουργήστε δύο TextView για τον τίτλο και τον υπότιτλο της εισαγωγικής σας σελίδας. Στη συνέχεια, θα πρέπει να δημιουργήσετε ένα LinearLayout για να ενσωματώσετε τα 4 checkboxes που αντιπροσωπεύουν 4 διαφορετικά αριθμητικά συστήματα. αμέσως μετά, δημιουργήστε τα 4 πραγματικά στοιχεία CheckBox. Κάτω από το LinearLayout θα πρέπει να προσθέσετε ένα στοιχείο Button που θα οδηγεί στις αντίστοιχες οθόνες ανάλογα με την επιλογή του χρήστη από τα checkboxes. Τέλος, στο κάτω μέρος της εισαγωγικής σελίδας δημιουργήστε αν θέλετε άλλο ένα TextView με οποιοδήποτε μήνυμα θέλετε να εμφανίσετε.

Το τελικό αποτέλεσμα του αρχείου "activity_intro.xml" θα πρέπει να μοιάζει με αυτό:

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:background="@drawable/background_image">

    <TextView
        android:id="@+id/titleTextView"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_centerHorizontal="true"
```

```
        android:layout_marginTop="50dp"
        android:text="Μετατροπέας αριθμητικών συστημάτων"
        android:textColor="@color/purple_700"
        android:textStyle="bold"
        android:gravity="center"
        android:textSize="36sp" />

    <TextView
        android:id="@+id/subtitleTextView"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_below="@id/titleTextView"
        android:layout_centerHorizontal="true"
        android:layout_marginTop="26dp"
        android:text="Επιλέξτε 2 συστήματα"
        android:textStyle="bold"
        android:textColor="@color/purple_700"
        android:textSize="26sp" />

    <LinearLayout
        android:id="@+id/checkBoxLayout"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_below="@id/subtitleTextView"
        android:layout_centerHorizontal="true"
        android:layout_marginTop="30dp"
        android:layout_marginStart="220dp"
        android:orientation="vertical">

        <CheckBox
            android:id="@+id/binaryCheckBox"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:textSize="24sp"
            android:text="2-αδικό"
            android:textColor="@color/purple_700"
            android:textStyle="bold"
            android:button="@drawable/custom_checkbox"
            android:layout_weight="1" />

        <CheckBox
            android:id="@+id/octalCheckBox"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:textSize="24sp"
```

```
            android:text="8-αδικό"
            android:textColor="@color/purple_700"
            android:textStyle="bold"
            android:button="@drawable/custom_checkbox"
            android:layout_marginTop="14dp"
            android:layout_weight="1" />

        <CheckBox
            android:id="@+id/decimalCheckBox"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:textSize="24sp"
            android:text="10-αδικό"
            android:textColor="@color/purple_700"
            android:textStyle="bold"
            android:button="@drawable/custom_checkbox"
            android:layout_marginTop="14dp"
            android:layout_weight="1" />

        <CheckBox
            android:id="@+id/hexadecimalCheckBox"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:textSize="24sp"
            android:text="16-αδικό"
            android:textColor="@color/purple_700"
            android:textStyle="bold"
            android:button="@drawable/custom_checkbox"
            android:layout_marginTop="14dp"
            android:layout_weight="1" />
    </LinearLayout>

    <Button
        android:id="@+id/startButton"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_below="@id/checkBoxLayout"
        android:layout_centerHorizontal="true"
        android:layout_marginTop="30dp"
        android:clickable="true"
        android:text="Start"
        android:background="@color/citro"
        android:textColor="@color/purple_700"
        android:textSize="24sp"
        android:focusable="true" />
```

```
    <TextView
        android:id="@+id/bottomtitleTextView"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_below="@id/titleTextView"
        android:layout_centerHorizontal="true"
        android:layout_marginTop="416dp"
        android:text="Η πίστη σώζει!!!"
        android:textStyle="bold"
        android:textColor="@color/purple_700"
        android:textSize="26sp" />

</RelativeLayout>
```

Ακολουθεί ένα στιγμιότυπο οθόνης της εισαγωγικής σελίδας στην προβολή Design του Android Studio:

Στον κώδικα XML της διάταξης της εισαγωγικής σελίδας, παρατηρείτε ότι χρησιμοποιώ μια εικόνα φόντου που είναι ένα αρχείο με όνομα background_image, και αναφερόμαστε σε αυτό το αρχείο με την καταχώρηση:
android:background="@drawable/background_image" στην κορυφή RelativeLayout.

Στην πραγματικότητα, έχουμε δημιουργήσει εκ των προτέρων ένα άλλο αρχείο xml με όνομα 'background_image.xml' που βρίσκεται στο φάκελο 'drawable' με τον ακόλουθο κώδικα μέσα:

```
<bitmap xmlns:android="http://schemas.android.com/apk/res/android"
    android:src="@drawable/intro_gradient"/>
```

Και φυσικά το αρχείο 'intro_gradient' στον παραπάνω κώδικα, αναφέρεται στο πραγματικό αρχείο εικόνας με όνομα 'intro_gradient.jpg' που θα πρέπει να έχουμε ήδη ανεβάσει στο φάκελο 'drawable'.

Όσον αφορά τα διάφορα χρώματα που παρατηρείτε στην εισαγωγική διάταξη, αυτά έχουν τοποθετηθεί σε ένα άλλο δημιουργημένο αρχείο xml με όνομα 'colors.xml' μέσα στον υποφάκελο 'values' του φακέλου 'drawable' και το περιεχόμενό του έχει ως εξής:

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <color name="purple_200">#FFBB86FC</color>
    <color name="purple_500">#FF6200EE</color>
    <color name="purple_700">#FF3700B3</color>
    <color name="teal_200">#FF03DAC5</color>
    <color name="teal_700">#FF018786</color>
    <color name="orange">#FFF5A65B</color>
    <color name="sage">#FFC1CC99</color>
    <color name="citro">#FFffbe0b</color>
    <color name="pale_brown">#FFbb9457</color>
    <color name="bright_pink">#FFf72585</color>
    <color name="black">#FF000000</color>
    <color name="white">#FFFFFFFF</color>
</resources>
```

Βήμα 3: Δημιουργήστε το αρχείο intro_activity.java

Τώρα ήρθε η ώρα να δημιουργήσετε τον κώδικα java για το περιβάλλον εργασίας διάταξης activity_intro.xml. Το όνομα του αρχείου που δημιουργούμε είναι 'intro_activity.java' και βρίσκεται στο φάκελο java, στον υποφάκελο του που σχετίζεται με

το com.example.'το όνομα του έργου μας'. Το περιεχόμενο αυτού του αρχείου java μοιάζει με το ακόλουθο:

```
package com.example.numeralsystemsconverter;

import androidx.appcompat.app.AppCompatActivity;

import android.annotation.SuppressLint;
import android.content.Intent;
import android.os.Bundle;
import android.widget.Button;
import android.widget.CheckBox;

public class intro_activity extends AppCompatActivity {

    private CheckBox binaryCheckBox;
    private CheckBox octalCheckBox;
    private CheckBox decimalCheckBox;
    private CheckBox hexadecimalCheckBox;

    @SuppressLint("MissingInflatedId")
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_intro);

        // initialize the UI elements
        binaryCheckBox = findViewById(R.id.binaryCheckBox);
        octalCheckBox = findViewById(R.id.octalCheckBox);
        decimalCheckBox = findViewById(R.id.decimalCheckBox);
        hexadecimalCheckBox =
findViewById(R.id.hexadecimalCheckBox);

        Button startButton = findViewById(R.id.startButton);

        // set an OnClickListener for the start button
        startButton.setOnClickListener(v -> {
            // get the number of selected checkboxes
            int numChecked = 0;
            if (binaryCheckBox.isChecked()) {
                numChecked++;
            }
            if (octalCheckBox.isChecked()) {
                numChecked++;
```

```
            }
            if (decimalCheckBox.isChecked()) {
                numChecked++;
            }
            if (hexadecimalCheckBox.isChecked()) {
                numChecked++;
            }

            // create an intent to start the appropriate
conversion activity based on the selected checkboxes
            Intent intent;
            if (numChecked == 2) {
                if (binaryCheckBox.isChecked() &&
octalCheckBox.isChecked()) {
                    intent = new Intent(intro_activity.this,
BinaryAndOctal.class);
                } else if (binaryCheckBox.isChecked() &&
decimalCheckBox.isChecked()) {
                    intent = new Intent(intro_activity.this,
BinaryAndDecimal.class);
                } else if (binaryCheckBox.isChecked() &&
hexadecimalCheckBox.isChecked()) {
                    intent = new Intent(intro_activity.this,
BinaryAndHexadecimal.class);
                } else if (octalCheckBox.isChecked() &&
decimalCheckBox.isChecked()) {
                    intent = new Intent(intro_activity.this,
OctalAndDecimal.class);
                } else if (octalCheckBox.isChecked() &&
hexadecimalCheckBox.isChecked()) {
                    intent = new Intent(intro_activity.this,
OctalAndHexadecimal.class);
                } else {
                    intent = new Intent(intro_activity.this,
DecimalAndHexadecimal.class);
                }
            } else {
                // If more or less than 2 checkboxes are selected,
do nothing
                return;
            }

            // start the selected activity
            startActivity(intent);
        });
```

```
    }
}
```

Το αρχείο java ουσιαστικά διαβάζει την είσοδο των κουτιών ελέγχου που επιλέγει ο χρήστης όταν επιλέγει τα αριθμητικά συστήματα που προτιμά να κάνει μετατροπές αριθμητικών συστημάτων και δημιουργεί την κατάλληλη νέα πρόθεση (ανακατεύθυνση στην κατάλληλη νέα διάταξη/κλάση) που εκτελεί τις πραγματικές μετατροπές μεταξύ των επιλεγμένων 2 αριθμητικών συστημάτων.

Βήμα 4: Δημιουργία των 6 ζευγών διατάξεων (αρχεία xml) και αρχείων java (κλάσεις) που εκτελούν τις πραγματικές μετατροπές.

File: activity_binary_and_decimal.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical"
    android:background="@color/purple_200"
    android:padding="16dp">

    <TextView
        android:id="@+id/title_text_view"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:textSize="38sp"
        android:gravity="center"
        android:textColor="@color/black"
        android:text="Δυαδικό σε Δεκαδικό"
        android:textStyle="bold"
        android:layout_marginTop="50dp"/>


    <EditText
        android:id="@+id/binary_edit_text"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:inputType="numberSigned|numberDecimal"
        android:textSize="26sp"
        android:hint="Εισάγετε 2-αδικό αριθμό"
```

```
        android:layout_marginTop="56dp"/>


    <EditText
        android:id="@+id/decimal_edit_text"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:inputType="numberSigned|numberDecimal"
        android:textSize="26sp"
        android:hint="Εισάγετε 10-αδικό αριθμό"
        android:layout_marginTop="36dp"/>

    <Button
        android:id="@+id/convert_button"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_gravity="center"
        android:text="Μετατροπή"
        android:layout_marginTop="46dp"/>

    <TextView
        android:id="@+id/output_text_view"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:textSize="26sp"
        android:textColor="@color/black"
        android:gravity="center"
        android:layout_marginTop="55dp"/>

</LinearLayout>
```

File: activity_binary_and_hexadecimal.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical"
    android:background="@color/orange"
    android:padding="16dp">

    <TextView
        android:id="@+id/title_text_view"
```

```
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:textSize="38sp"
        android:gravity="center"
        android:textColor="@color/black"
        android:text="Δυαδικό σε Δεκαεξαδικό"
        android:textStyle="bold"
        android:layout_marginTop="50dp"/>

    <EditText
        android:id="@+id/binary_edit_text"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:inputType="numberSigned|numberDecimal"
        android:textSize="26sp"
        android:hint="Εισάγετε 2-αδικό αριθμό"
        android:layout_marginTop="56dp"/>

    <EditText
        android:id="@+id/hexadecimal_edit_text"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:inputType="text"
        android:textSize="26sp"
        android:hint="Εισάγετε 16-αδικό αριθμό"
        android:layout_marginTop="36dp"/>

    <Button
        android:id="@+id/convert_button"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_gravity="center"
        android:text="Μετατροπή"
        android:layout_marginTop="46dp"/>

    <TextView
        android:id="@+id/output_text_view"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:textSize="26sp"
        android:textColor="@color/black"
        android:gravity="center"
        android:layout_marginTop="55dp"/>
</LinearLayout>
```

File: activity_binary_and_octal.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical"
    android:background="@color/sage"
    android:padding="16dp">

    <TextView
        android:id="@+id/title_text_view"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:textSize="38sp"
        android:gravity="center"
        android:textColor="@color/black"
        android:text="Δυαδικό σε Οκταδικό"
        android:textStyle="bold"
        android:layout_marginTop="50dp"/>


    <EditText
        android:id="@+id/binary_edit_text"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:inputType="numberSigned|numberDecimal"
        android:textSize="26sp"
        android:hint="Εισάγετε 2-αδικό αριθμό"
        android:layout_marginTop="56dp"/>


    <EditText
        android:id="@+id/octal_edit_text"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:inputType="numberSigned|numberDecimal"
        android:textSize="26sp"
        android:hint="Εισάγετε 8-αδικό αριθμό"
        android:layout_marginTop="36dp"/>

    <Button
        android:id="@+id/convert_button"
        android:layout_width="wrap_content"
```

```
        android:layout_height="wrap_content"
        android:layout_gravity="center"
        android:text="Μετατροπή"
        android:layout_marginTop="46dp"/>

    <TextView
        android:id="@+id/output_text_view"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:textSize="26sp"
        android:textColor="@color/black"
        android:gravity="center"
        android:layout_marginTop="55dp"/>

</LinearLayout>
```

File: activity_decimal_and_hexadecimal.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical"
    android:background="@color/citro"
    android:padding="16dp">

    <TextView
        android:id="@+id/title_text_view"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:textSize="38sp"
        android:gravity="center"
        android:textColor="@color/black"
        android:text="Δεκαδικό σε Δεκαεξαδικό"
        android:textStyle="bold"
        android:layout_marginTop="50dp"/>


    <EditText
        android:id="@+id/decimal_edit_text"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:inputType="numberSigned|numberDecimal"
```

```
        android:textSize="26sp"
        android:hint="Εισάγετε 10-αδικό αριθμό"
        android:layout_marginTop="56dp"/>


    <EditText
        android:id="@+id/hexadecimal_edit_text"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:inputType="text"
        android:textSize="26sp"
        android:hint="Εισάγετε 16-αδικό αριθμό"
        android:layout_marginTop="36dp"/>

    <Button
        android:id="@+id/convert_button"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_gravity="center"
        android:text="Μετατροπή"
        android:layout_marginTop="46dp"/>

    <TextView
        android:id="@+id/output_text_view"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:textSize="26sp"
        android:textColor="@color/black"
        android:gravity="center"
        android:layout_marginTop="55dp"/>

</LinearLayout>
```

File: activity_octal_and_decimal.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical"
    android:background="@color/pale_brown"
    android:padding="16dp">
```

```
<TextView
    android:id="@+id/title_text_view"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:textSize="38sp"
    android:gravity="center"
    android:textColor="@color/black"
    android:text="Οκταδικό σε Δεκαδικό"
    android:textStyle="bold"
    android:layout_marginTop="50dp"/>


<EditText
    android:id="@+id/octal_edit_text"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:inputType="numberSigned|numberDecimal"
    android:textSize="26sp"
    android:hint="Εισάγετε 8-αδικό αριθμό"
    android:layout_marginTop="56dp"/>


<EditText
    android:id="@+id/decimal_edit_text"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:inputType="numberSigned|numberDecimal"
    android:textSize="26sp"
    android:hint="Εισάγετε 10-αδικό αριθμό"
    android:layout_marginTop="36dp"/>

<Button
    android:id="@+id/convert_button"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:layout_gravity="center"
    android:text="Μετατροπή"
    android:layout_marginTop="46dp"/>

<TextView
    android:id="@+id/output_text_view"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:textSize="26sp"
    android:textColor="@color/black"
```

```
        android:gravity="center"
        android:layout_marginTop="55dp"/>

</LinearLayout>
```

File: activity_octal_and_hexadecimal.xml

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical"
    android:background="@color/bright_pink"
    android:padding="16dp">

    <TextView
        android:id="@+id/title_text_view"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:textSize="38sp"
        android:gravity="center"
        android:textColor="@color/black"
        android:text="Οκταδικό σε Δεκαεξαδικό"
        android:textStyle="bold"
        android:layout_marginTop="50dp"/>


    <EditText
        android:id="@+id/octal_edit_text"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:inputType="numberSigned|numberDecimal"
        android:textSize="26sp"
        android:hint="Εισάγετε 8-αδικό αριθμό"
        android:layout_marginTop="56dp"/>


    <EditText
        android:id="@+id/hexadecimal_edit_text"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:inputType="text"
        android:textSize="26sp"
```

```
        android:hint="Εισάγετε 16-αδικό αριθμό"
        android:layout_marginTop="36dp"/>

    <Button
        android:id="@+id/convert_button"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_gravity="center"
        android:text="Μετατροπή"
        android:layout_marginTop="46dp"/>

    <TextView
        android:id="@+id/output_text_view"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:textSize="26sp"
        android:textColor="@color/black"
        android:gravity="center"
        android:layout_marginTop="55dp"/>

</LinearLayout>
```

File: BinaryAndDecimal.java

```
package com.example.numeralsystemsconverter;

import android.annotation.SuppressLint;
import android.os.Bundle;
import android.widget.Button;
import android.widget.EditText;
import android.widget.TextView;

import androidx.appcompat.app.AppCompatActivity;

public class BinaryAndDecimal extends AppCompatActivity {

    EditText binaryEditText, decimalEditText;
    TextView outputTextView, titleTextView;
    Button convertButton;

    @SuppressLint({"MissingInflatedId", "SetTextI18n"})
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
```

```
        setContentView(R.layout.activity_binary_and_decimal);

        // Get references to the UI elements
        titleTextView = findViewById(R.id.title_text_view);
        binaryEditText = findViewById(R.id.binary_edit_text);
        decimalEditText = findViewById(R.id.decimal_edit_text);
        outputTextView = findViewById(R.id.output_text_view);
        convertButton = findViewById(R.id.convert_button);

        // Set click listener for the convert button
        convertButton.setOnClickListener(v -> {

            // Convert binary to decimal
            String binaryString =
binaryEditText.getText().toString();
            if (!binaryString.isEmpty()) {
                if (validateBinaryInput(binaryString)) {
                    double decimal = 0.0;
                    boolean isNegative = false;
                    if (binaryString.startsWith("-")) {
                        isNegative = true;
                        binaryString = binaryString.substring(1);
                    }
                    if (binaryString.contains(".")) {
                        // Handle decimal binary numbers
                        String[] parts =
binaryString.split("\\.");
                        for (int i = 0; i < parts[0].length();
i++) {
                            if (parts[0].charAt(i) == '1') {
                                decimal += Math.pow(2,
parts[0].length() - i - 1);
                            }
                        }
                        for (int i = 0; i < parts[1].length();
i++) {
                            if (parts[1].charAt(i) == '1') {
                                decimal += Math.pow(2, -i - 1);
                            }
                        }
                    } else {
                        decimal = Integer.parseInt(binaryString,
2);
                    }
                    if (isNegative) {
```

```
                    decimal = -decimal;
                }
                if (decimal % 1 == 0) {

outputTextView.setText(Integer.toString((int) decimal));
                } else {

outputTextView.setText(Double.toString(decimal));
                }

            } else {
                outputTextView.setText("μή έγκυρη τιμή");
            }
        }

        // Convert decimal to binary
        String decimalString =
decimalEditText.getText().toString();
        if (!decimalString.isEmpty()) {
            double decimal = 0.0;
            try {
                decimal = Double.parseDouble(decimalString);
            } catch (NumberFormatException e) {
                outputTextView.setText("μή έγκυρη τιμή");
                return;
            }
            String binary = "";
            if (decimal < 0) {
                binary = "-";
                decimal = -decimal;
            }
            int integerPart = (int) decimal;
            double decimalPart = decimal - integerPart;

            // Add condition to check if there is a fractional
part before appending the decimal point
            if (decimalPart != 0) {
                binary += Integer.toBinaryString(integerPart)
+ ".";
            } else {
                binary += Integer.toBinaryString(integerPart);
            }

            int maxDecimalPlaces = 10; // Set a maximum number
of decimal places
```

```
                while (decimalPart > 0 && maxDecimalPlaces > 0) {
                    decimalPart *= 2;
                    if (decimalPart >= 1) {
                        binary += "1";
                        decimalPart -= 1;
                    } else {
                        binary += "0";
                    }
                    maxDecimalPlaces--;
                }

                outputTextView.setText(binary);
            }
        });
    }
    private boolean validateBinaryInput(String binaryString) {
        if (binaryString.isEmpty()) {
            return false;
        }
        boolean isNegative = false;
        if (binaryString.charAt(0) == '-') {
            isNegative = true;
            binaryString = binaryString.substring(1);
        }
        boolean hasDecimalPoint = false;
        for (int i = 0; i < binaryString.length(); i++) {
            char c = binaryString.charAt(i);
            if (c == '.') {
                if (hasDecimalPoint) {
                    return false; // only one decimal point
allowed
                }
                hasDecimalPoint = true;
            } else if (c != '0' && c != '1') {
                return false; // invalid character
            }
        }
        if (isNegative) {
            binaryString = '-' + binaryString;
        }
        return true;
    }
}
```

File: BinaryAndHexadecimal.java

```
package com.example.numeralsystemsconverter;

import android.annotation.SuppressLint;
import android.os.Bundle;
import android.util.Log;
import android.widget.Button;
import android.widget.EditText;
import android.widget.TextView;
import androidx.appcompat.app.AppCompatActivity;

public class BinaryAndHexadecimal extends AppCompatActivity {

    EditText binaryEditText, hexadecimalEditText;
    TextView outputTextView, titleTextView;
    Button convertButton;

    @SuppressLint({"MissingInflatedId", "SetTextI18n"})
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_binary_and_hexadecimal);

        // Get references to the UI elements
        titleTextView = findViewById(R.id.title_text_view);
        binaryEditText = findViewById(R.id.binary_edit_text);
        hexadecimalEditText =
findViewById(R.id.hexadecimal_edit_text);
        outputTextView = findViewById(R.id.output_text_view);
        convertButton = findViewById(R.id.convert_button);

        // Set click listener for the convert button
        convertButton.setOnClickListener(v -> {

            // Convert binary to hexadecimal

            String binaryString =
binaryEditText.getText().toString();

            if (!isValidBinary(binaryString)) {
                outputTextView.setText("μή έγκυρη τιμή");
            } else {
                // Continue with conversion
                // Check if the binary number is negative
                if (binaryString.length() > 0) {
```

```
                    boolean isNegative = false;
                    if (binaryString.charAt(0) == '-') {
                        isNegative = true;
                        binaryString = binaryString.substring(1);
                    }
                    // Split the binary number into integer and
fractional parts
                    String[] parts = binaryString.split("\\.");
                    String integerPart = parts[0];
                    String fractionalPart = parts.length > 1 ?
parts[1] : "";

                    // Convert the integer part to hexadecimal
                    StringBuilder hexBuilder = new
StringBuilder();
                    int missingZeros = 4 - integerPart.length() %
4;
                    if (missingZeros != 4) {
                        integerPart = "0".repeat(missingZeros) +
integerPart;
                    }
                    for (int i = 0; i < integerPart.length(); i +=
4) {
                        String fourBits = integerPart.substring(i,
i + 4);
                        int decimalValue =
Integer.parseInt(fourBits, 2);

hexBuilder.append(Integer.toHexString(decimalValue).toUpperCase())
;
                    }
                    // Convert the fractional part to hexadecimal
                    if (!fractionalPart.isEmpty()) {
                        hexBuilder.append(".");
                        missingZeros = 4 - fractionalPart.length()
% 4;
                        if (missingZeros != 4) {
                            fractionalPart +=
"0".repeat(missingZeros);
                        }
                        for (int i = 0; i <
fractionalPart.length(); i += 4) {
                            String fourBits =
fractionalPart.substring(i, i + 4);
                            int decimalValue =
```

```
Integer.parseInt(fourBits, 2);

hexBuilder.append(Integer.toHexString(decimalValue).toUpperCase())
;
                        }
                    }
                    // Add a negative sign if the original binary
number was negative
                    if (isNegative) {
                        hexBuilder.insert(0, "-");
                    }
                    // The final hexadecimal representation of the
binary number
                    String hexString = hexBuilder.toString();
                    outputTextView.setText(hexString);
                }
            }

            // Convert Hexadecimal to Binary

            String hexadecimalString =
hexadecimalEditText.getText().toString();

            if (hexadecimalString.matches("-?[0-9a-fA-F]+")) {
                // hexadecimalString is a valid hexadecimal number
                // Check if the hexadecimal number is negative
                if (hexadecimalString.length() > 0) {
                    boolean isHexNegative = false;
                    if (hexadecimalString.charAt(0) == '-') {
                        isHexNegative = true;
                        hexadecimalString =
hexadecimalString.substring(1);
                    }

                    // Split the hexadecimal number into integer
and fractional parts
                    String[] parts2 =
hexadecimalString.split("\\.");
                    String integerPart2 = parts2[0];
                    String fractionalPart2 = parts2.length > 1 ?
parts2[1] : "";

                    // Convert the integer part to binary
                    StringBuilder binaryBuilder2 = new
StringBuilder();
```

```
                    for (int i = 0; i < integerPart2.length();
i++) {
                        char hexDigit = integerPart2.charAt(i);
                        int decimalValue2 =
Integer.parseInt(String.valueOf(hexDigit), 16);
                        String fourBits =
Integer.toBinaryString(decimalValue2);
                        binaryBuilder2.append("0".repeat(4 -
fourBits.length())).append(fourBits);
                    }

                    // Remove leading zeros
                    while (binaryBuilder2.length() > 1 &&
binaryBuilder2.charAt(0) == '0') {
                        binaryBuilder2.deleteCharAt(0);
                    }

                    // Convert the fractional part to binary
                    if (!fractionalPart2.isEmpty()) {
                        binaryBuilder2.append(".");
                        for (int i = 0; i <
fractionalPart2.length(); i++) {
                            char hexDigit =
fractionalPart2.charAt(i);
                            int decimalValue =
Integer.parseInt(String.valueOf(hexDigit), 16);
                            String fourBits =
Integer.toBinaryString(decimalValue);
                            binaryBuilder2.append("0".repeat(4 -
fourBits.length())).append(fourBits);
                        }
                    }

                    // Add a negative sign if the original
hexadecimal number was negative
                    if (isHexNegative) {
                        binaryBuilder2.insert(0, "-");
                    }

                    // The final binary representation of the
hexadecimal number
                    String binaryString2 =
binaryBuilder2.toString();
                    outputTextView.setText(binaryString2);
            } else {
```

```
                    outputTextView.setText("μή έγκυρη τιμή");
                }
                }
            });
        }

        private boolean isValidBinary(String binaryString) {
            // Check if the binary string is empty
            if (binaryString.isEmpty()) {
                return false;
            }
            // Check if the binary string contains only 0s, 1s and a
single decimal point
            int pointCount = 0;
            for (int i = 0; i < binaryString.length(); i++) {
                char c = binaryString.charAt(i);
                if (c != '0' && c != '1' && c != '.' && c != '-') {
                    return false;
                }
                if (c == '.') {
                    pointCount++;
                    if (pointCount > 1) {
                        return false;
                    }
                }
                // Check that minus sign is only at the beginning of
the string
                if (c == '-' && i != 0) {
                    return false;
                }
            }
            return true;
        }
}
```

File: BinaryAndOctal.java

```
package com.example.numeralsystemsconverter;

import android.annotation.SuppressLint;
import android.os.Bundle;
import android.widget.Button;
import android.widget.EditText;
import android.widget.TextView;
```

```
import androidx.appcompat.app.AppCompatActivity;

public class BinaryAndOctal extends AppCompatActivity {

    private EditText binaryEditText;
    private EditText octalEditText;
    private TextView outputTextView;

    @SuppressLint("SetTextI18n")
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_binary_and_octal);

        binaryEditText = findViewById(R.id.binary_edit_text);
        octalEditText = findViewById(R.id.octal_edit_text);
        Button convertButton = findViewById(R.id.convert_button);
        outputTextView = findViewById(R.id.output_text_view);

        convertButton.setOnClickListener(view -> {

            // Convert Binary to Octal
            String binaryString =
binaryEditText.getText().toString().trim();

            if (isValidBinary(binaryString)) {
                // The input binaryString is valid
                if (binaryString.length() > 0) {
                    boolean isNegative = false;
                    if (binaryString.charAt(0) == '-') {
                        isNegative = true;
                        binaryString = binaryString.substring(1);
                    }

                    int pointIndex = binaryString.indexOf('.');
                    if (pointIndex == -1) {
                        pointIndex = binaryString.length();
                    }

                    // Convert the integer part to decimal
                    int intPartDecimal = 0;
                    for (int i = 0; i < pointIndex; i++) {
                        intPartDecimal +=
Character.getNumericValue(binaryString.charAt(i)) * Math.pow(2,
```

```
pointIndex - i - 1);
                    }

                    // Convert the fractional part to decimal
                    double fracPartDecimal = 0;
                    for (int i = pointIndex + 1; i <
binaryString.length(); i++) {
                        fracPartDecimal +=
Character.getNumericValue(binaryString.charAt(i)) * Math.pow(2,
pointIndex - i);
                    }

                    // Convert the integer part to octal
                    StringBuilder intPartOctalBuilder = new
StringBuilder();
                    while (intPartDecimal > 0) {
                        int remainder = intPartDecimal % 8;
                        intPartOctalBuilder.append(remainder);
                        intPartDecimal /= 8;
                    }
                    intPartOctalBuilder.reverse();

                    // Convert the fractional part to octal
                    StringBuilder fracPartOctalBuilder = new
StringBuilder(".");
                    while (fracPartDecimal > 0) {
                        fracPartDecimal *= 8;
                        int digit = (int) fracPartDecimal;
                        fracPartOctalBuilder.append(digit);
                        fracPartDecimal -= digit;
                    }

                    // Combine the integer and fractional parts
and add negative sign if needed
                    StringBuilder resultBuilder = new
StringBuilder();
                    if (isNegative) {
                        resultBuilder.append('-');
                    }
                    resultBuilder.append(intPartOctalBuilder);
                    if (fracPartOctalBuilder.length() > 1) { //
check if there are any digits after the decimal point

resultBuilder.append(fracPartOctalBuilder);
                    }
```

```
outputTextView.setText(resultBuilder.toString());
                }
            } else {
                // The input binaryString is not valid
                outputTextView.setText("μή έγκυρη τιμή");
            }



            // Convert Octal to Binary

            String octalString =
octalEditText.getText().toString().trim();

            if (isValidOctal(octalString)) {
                // The input octalString is valid
                if (octalString.length() > 0) {
                    boolean isOctalNegative = false;
                    if (octalString.charAt(0) == '-') {
                        isOctalNegative = true;
                        octalString = octalString.substring(1);
                    }

                    int pointOctalIndex =
octalString.indexOf('.');
                    if (pointOctalIndex == -1) {
                        pointOctalIndex = octalString.length();
                    }

                    // Convert the integer part to decimal
                    int intOctalPartDecimal = 0;
                    for (int i = 0; i < pointOctalIndex; i++) {
                        intOctalPartDecimal +=
Character.getNumericValue(octalString.charAt(i)) * Math.pow(8,
pointOctalIndex - i - 1);
                    }

                    // Convert the fractional part to decimal
                    double fracOctalPartDecimal = 0;
                    for (int i = pointOctalIndex + 1; i <
octalString.length(); i++) {
                        fracOctalPartDecimal +=
Character.getNumericValue(octalString.charAt(i)) * Math.pow(8,
```

```
pointOctalIndex - i);
                    }

                    // Convert the integer part to binary
                    StringBuilder intPartBinaryBuilder = new
StringBuilder();
                    while (intOctalPartDecimal > 0) {
                        int remainder = intOctalPartDecimal % 2;
                        intPartBinaryBuilder.append(remainder);
                        intOctalPartDecimal /= 2;
                    }
                    intPartBinaryBuilder.reverse();

                    // Convert the fractional part to binary
                    StringBuilder fracPartBinaryBuilder = new
StringBuilder(".");
                    while (fracOctalPartDecimal > 0) {
                        fracOctalPartDecimal *= 2;
                        int digit = (int) fracOctalPartDecimal;
                        fracPartBinaryBuilder.append(digit);
                        fracOctalPartDecimal -= digit;
                    }

                    // Combine the integer and fractional parts
and add negative sign if needed
                    StringBuilder resultBinaryBuilder = new
StringBuilder();
                    if (isOctalNegative) {
                        resultBinaryBuilder.append('-');
                    }

resultBinaryBuilder.append(intPartBinaryBuilder);
                    if (fracPartBinaryBuilder.length() > 1) { //
check if there are any digits after the decimal point

resultBinaryBuilder.append(fracPartBinaryBuilder);
                    }


outputTextView.setText(resultBinaryBuilder.toString());
            } else {
                // The input octalString is not valid
                outputTextView.setText("μή έγκυρη τιμή");
            }
```

```
            }
        });
    }

    private boolean isValidBinary(String binaryString) {
        // Check if the binary string is empty
        if (binaryString.isEmpty()) {
            return false;
        }
        // Check if the binary string contains only 0s, 1s and a
single decimal point
        int pointCount = 0;
        for (int i = 0; i < binaryString.length(); i++) {
            char c = binaryString.charAt(i);
            if (c != '0' && c != '1' && c != '.' && c != '-') {
                return false;
            }
            if (c == '.') {
                pointCount++;
                if (pointCount > 1) {
                    return false;
                }
            }
            // Check that minus sign is only at the beginning of
the string
            if (c == '-' && i != 0) {
                return false;
            }
        }
        return true;
    }

    private boolean isValidOctal(String octalString) {
        // Check if the octal string is empty
        if (octalString.isEmpty()) {
            return false;
        }
        // Check if the octal string contains only valid digits of
the octal numeral system plus accepting fractional part and plus
accepting negative sign at front
        int pointCount = 0;
        for (int i = 0; i < octalString.length(); i++) {
            char c = octalString.charAt(i);
            if ((c < '0' || c > '7') && c != '.' && c != '-') {
```

```
                return false;
            }
            if (c == '.') {
                pointCount++;
                if (pointCount > 1) {
                    return false;
                }
            }
            // Check that minus sign is only at the beginning of
the string
            if (c == '-' && i != 0) {
                return false;
            }
        }
        return true;
    }
}
```

File: DecimalAndHexadecimal.java

```
package com.example.numeralsystemsconverter;

import android.annotation.SuppressLint;
import android.os.Bundle;
import android.text.TextUtils;
import android.widget.Button;
import android.widget.EditText;
import android.widget.TextView;

import androidx.appcompat.app.AppCompatActivity;

public class DecimalAndHexadecimal extends AppCompatActivity {

    private EditText decimalEditText;
    private EditText hexadecimalEditText;
    private TextView outputTextView;

    @SuppressLint("SetTextI18n")
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_decimal_and_hexadecimal);

        decimalEditText = findViewById(R.id.decimal_edit_text);
```

```
        hexadecimalEditText =
findViewById(R.id.hexadecimal_edit_text);
        outputTextView = findViewById(R.id.output_text_view);

        Button convertButton = findViewById(R.id.convert_button);
        convertButton.setOnClickListener(v -> {
            String decimalString =
decimalEditText.getText().toString().trim();
            String hexadecimalString =
hexadecimalEditText.getText().toString().trim();

            // Check if both fields are empty
            if (TextUtils.isEmpty(decimalString) &&
TextUtils.isEmpty(hexadecimalString)) {
                outputTextView.setText("Εισάγετε μια τιμή για
μετατροπή.");
                return;
            }

            // Check if both fields are filled in
            if (!TextUtils.isEmpty(decimalString) &&
!TextUtils.isEmpty(hexadecimalString)) {
                outputTextView.setText("Εισάγετε μόνο μια τιμή σε
ένα πεδίο.");
                return;
            }

            // Check if the decimal input is valid
            if (!TextUtils.isEmpty(decimalString) &&
!isValidDecimal(decimalString)) {
                outputTextView.setText("Εισάγετε έναν έγκυρο
δεκαδικό αριθμό.");
                return;
            }

            // Check if the hexadecimal input is valid
            if (!TextUtils.isEmpty(hexadecimalString) &&
!isValidHexadecimal(hexadecimalString)) {
                outputTextView.setText("Εισάγετε έναν έγκυρο
δεκαεξαδικό αριθμό.");
                return;
            }

            // Perform the conversion
            if (!TextUtils.isEmpty(decimalString)) {
```

```
                double decimal =
Double.parseDouble(decimalString);
                int integerPart = (int) decimal;
                double fractionalPart = Math.abs(decimal -
integerPart);
                String hexadecimal;
                if (decimal < 0) {
                    hexadecimal = "-" +
Integer.toHexString(Math.abs(integerPart));
                } else {
                    hexadecimal =
Integer.toHexString(integerPart);
                }
                if (fractionalPart > 0) {
                    hexadecimal += ".";
                    while (fractionalPart > 0) {
                        fractionalPart *= 16;
                        int digit = (int) fractionalPart;
                        hexadecimal += Integer.toHexString(digit);
                        fractionalPart -= digit;
                    }
                }
                outputTextView.setText(hexadecimal.toUpperCase());
            } else {
                int decimal = Integer.parseInt(hexadecimalString,
16);
                outputTextView.setText(Integer.toString(decimal));
            }
        });
    }

    private boolean isValidDecimal(String decimalString) {
        // Regular expression for a decimal string with fractional
part and optional negative sign
        String regex = "-?[0-9]+(\\.[0-9]+)?";
        return decimalString.matches(regex);
    }

    private boolean isValidHexadecimal(String hexadecimalString) {
        // Regular expression for a hexadecimal string with
optional negative sign
        String regex = "-?[0-9A-Fa-f]+";
        return !hexadecimalString.contains(".") &&
hexadecimalString.matches(regex);
    }
```

```
}
```

File: OctalAndDecimal.java

```
package com.example.numeralsystemsconverter;

import android.os.Bundle;
import android.text.TextUtils;
import android.widget.Button;
import android.widget.EditText;
import android.widget.TextView;

import androidx.appcompat.app.AppCompatActivity;

public class OctalAndDecimal extends AppCompatActivity {
    private EditText octalEditText, decimalEditText;
    private TextView outputTextView;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_octal_and_decimal);

        octalEditText = findViewById(R.id.octal_edit_text);
        decimalEditText = findViewById(R.id.decimal_edit_text);
        outputTextView = findViewById(R.id.output_text_view);
        Button convertButton = findViewById(R.id.convert_button);

        convertButton.setOnClickListener(v -> {
            String octalString =
octalEditText.getText().toString().trim();
            String decimalString =
decimalEditText.getText().toString().trim();
            if (TextUtils.isEmpty(octalString) &&
TextUtils.isEmpty(decimalString)) {
                outputTextView.setText("Εισάγετε έναν οκταδικό ή
δεκαδικό αριθμό.");
            } else if (!TextUtils.isEmpty(octalString) &&
!TextUtils.isEmpty(decimalString)) {
                outputTextView.setText("Εισάγετε μόνο, έναν
οκταδικό ή δεκαδικό αριθμό.");
            } else if (!TextUtils.isEmpty(octalString)) {
                if (!isValidOctal(octalString)) {
                    outputTextView.setText("Εισάγετε έναν έγκυρο
```

```
οκταδικό αριθμό.");
                } else {
                    double decimalValue =
octalToDecimal(octalString);

outputTextView.setText(String.valueOf(decimalValue));
                }
            } else if (!TextUtils.isEmpty(decimalString)) {
                if (!isValidDecimal(decimalString)) {
                    outputTextView.setText("Εισάγετε έναν έγκυρο
δεκαδικό αριθμό.");
                } else {
                    double decimalValue =
Double.parseDouble(decimalString);
                    String octalValue =
decimalToOctal(decimalValue);
                    outputTextView.setText(octalValue);
                }
            }
        });
    }

    private boolean isValidOctal(String octalString) {
        if (octalString.charAt(0) == '-') {
            octalString = octalString.substring(1);
        }
        String[] parts = octalString.split("\\.");
        for (int i = 0; i < parts[0].length(); i++) {
            char c = parts[0].charAt(i);
            if (c < '0' || c > '7') {
                return false;
            }
        }
        if (parts.length > 1) {
            for (int i = 0; i < parts[1].length(); i++) {
                char c = parts[1].charAt(i);
                if (c < '0' || c > '7') {
                    return false;
                }
            }
        }
        return true;
    }

    private double octalToDecimal(String octalString) {
```

```
    boolean isNegative = false;
    if (octalString.charAt(0) == '-') {
        isNegative = true;
        octalString = octalString.substring(1);
    }
    String[] parts = octalString.split("\\.");
    double decimalValue = 0.0;
    int n = parts[0].length();
    for (int i = 0; i < n; i++) {
        char c = parts[0].charAt(i);
        int digit = c - '0';
        decimalValue += digit * Math.pow(8, n - i - 1);
    }
    if (parts.length > 1) {
        n = parts[1].length();
        for (int i = 0; i < n; i++) {
            char c = parts[1].charAt(i);
            int digit = c - '0';
            decimalValue += digit * Math.pow(8, -(i + 1));
        }
    }
    return isNegative ? -decimalValue : decimalValue;
}

private boolean isValidDecimal(String decimalString) {
    try {
        Double.parseDouble(decimalString);
        return true;
    } catch (NumberFormatException e) {
        return false;
    }
}

private String decimalToOctal(double decimalValue) {
    StringBuilder octalValue = new StringBuilder();
    if (decimalValue == 0) {
        return "0";
    }
    boolean isNegative = false;
    if (decimalValue < 0) {
        isNegative = true;
        decimalValue = -decimalValue;
    }
    int integerPart = (int) decimalValue;
    double fractionalPart = decimalValue - integerPart;
```

```
        while (integerPart > 0) {
            int remainder = Math.floorMod(integerPart, 8);
            octalValue.append(remainder);
            integerPart /= 8;
        }
        if (isNegative) {
            octalValue.append("-");
        }
        octalValue.reverse();

        if (fractionalPart > 0) {
            octalValue.append(".");
            while (fractionalPart > 0 && octalValue.length() < 20)
{ // Limit the length of the result
                fractionalPart *= 8;
                int digit = (int) fractionalPart;
                octalValue.append(digit);
                fractionalPart -= digit;
            }
        }

        return octalValue.toString();
    }
}
```

File: OctalAndHexadecimal.java

```
package com.example.numeralsystemsconverter;

import android.os.Bundle;
import android.text.TextUtils;
import android.view.View;
import android.widget.Button;
import android.widget.EditText;
import android.widget.TextView;

import androidx.appcompat.app.AppCompatActivity;

public class OctalAndHexadecimal extends AppCompatActivity {

    private EditText octalEditText;
    private EditText hexadecimalEditText;
    private TextView outputTextView;
```

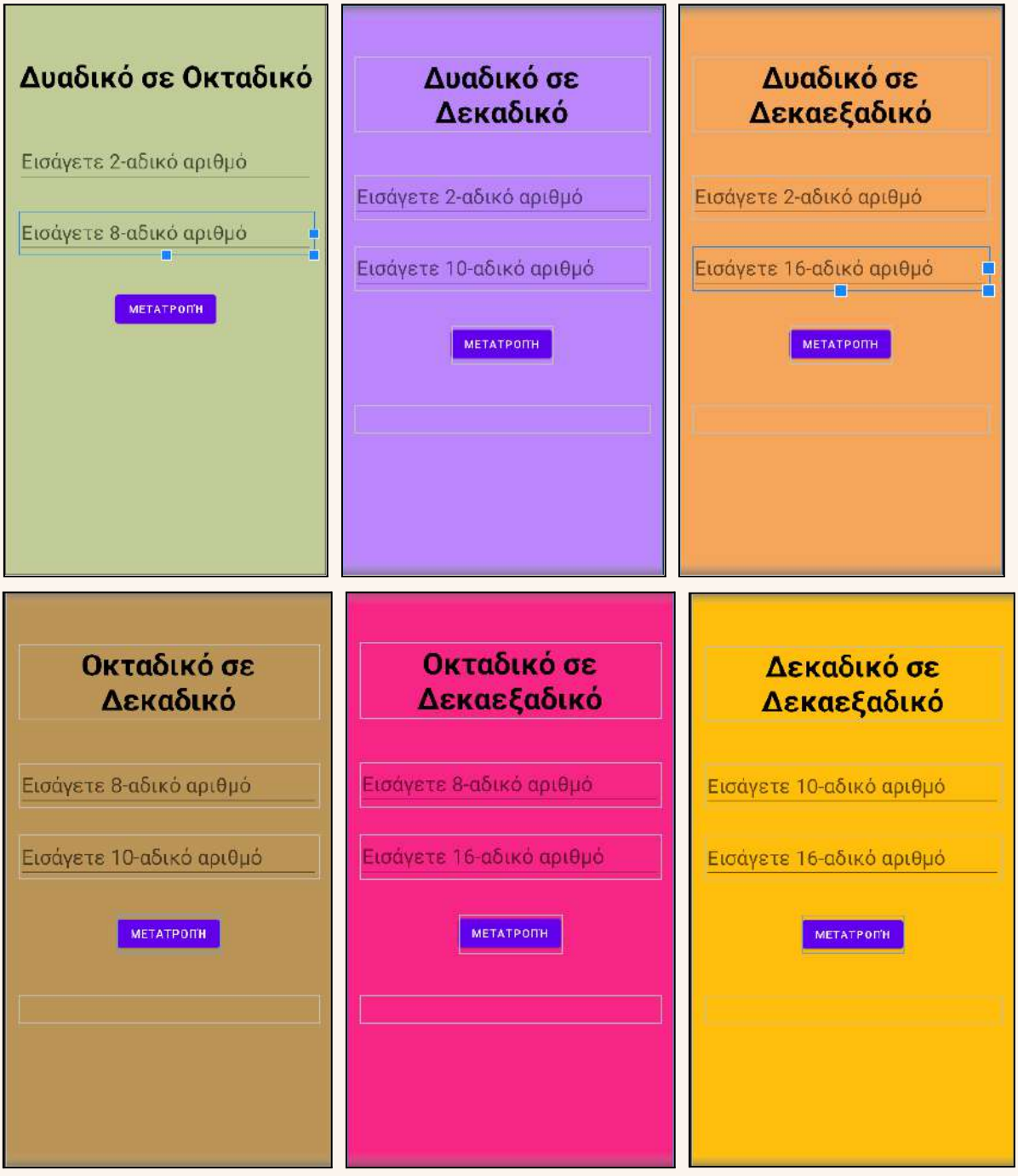
```
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_octal_and_hexadecimal);

        octalEditText = findViewById(R.id.octal_edit_text);
        hexadecimalEditText =
findViewById(R.id.hexadecimal_edit_text);
        outputTextView = findViewById(R.id.output_text_view);

        Button convertButton = findViewById(R.id.convert_button);
        convertButton.setOnClickListener(new
View.OnClickListener() {
            @Override
            public void onClick(View v) {
                String octalString =
octalEditText.getText().toString().trim();
                String hexadecimalString =
hexadecimalEditText.getText().toString().trim();

                // Check if both fields are empty
                if (TextUtils.isEmpty(octalString) &&
TextUtils.isEmpty(hexadecimalString)) {
                    outputTextView.setText("Εισάγετε μια τιμή για
μετατροπή.");
                    return;
                }

                // Check if both fields are filled in
                if (!TextUtils.isEmpty(octalString) &&
!TextUtils.isEmpty(hexadecimalString)) {
                    outputTextView.setText("Εισάγετε μόνο μια τιμή
σε ένα πεδίο.");
                    return;
                }

                // Check if the octal input is valid
                if (!TextUtils.isEmpty(octalString) &&
!isValidOctal(octalString)) {
                    outputTextView.setText("Εισάγετε έναν έγκυρο
οκταδικό αριθμό.");
                    return;
                }

                // Check if the hexadecimal input is valid
```

```
                if (!TextUtils.isEmpty(hexadecimalString) &&
!isValidHexadecimal(hexadecimalString)) {
                    outputTextView.setText("Εισάγετε έναν έγκυρο
δεκαεξαδικό αριθμό.");
                    return;
                }

                // Perform the conversion
                if (!TextUtils.isEmpty(octalString)) {
                    boolean isNegative = false;
                    if (octalString.charAt(0) == '-') {
                        isNegative = true;
                        octalString = octalString.substring(1);
                    }
                    String[] parts = octalString.split("\\.");
                    int decimalValue = Integer.parseInt(parts[0],
8);
                    String hexadecimalValue =
Integer.toHexString(decimalValue).toUpperCase();

                    if (parts.length > 1) {
                        StringBuilder fractionalHexadecimalValue =
new StringBuilder();
                        double fractionalPart = 0.0;
                        for (int i = 0; i < parts[1].length();
i++) {
                            char c = parts[1].charAt(i);
                            int digit = c - '0';
                            fractionalPart += digit * Math.pow(8,
-(i + 1));
                        }
                        while (fractionalPart > 0 &&
fractionalHexadecimalValue.length() < 6) { // Limit the length of
the result
                            fractionalPart *= 16;
                            int digit = (int) fractionalPart;
                            String hexDigit =
Integer.toHexString(digit).toUpperCase();

fractionalHexadecimalValue.append(hexDigit);
                            fractionalPart -= digit;
                        }
                        hexadecimalValue += "." +
fractionalHexadecimalValue.toString();
                    }
```

```
                    if (isNegative) {
                        hexadecimalValue = "-" + hexadecimalValue;
                    }

                    outputTextView.setText(hexadecimalValue);
                } else {
                    boolean isNegative = false;
                    if (hexadecimalString.charAt(0) == '-') {
                        isNegative = true;
                        hexadecimalString =
hexadecimalString.substring(1);
                    }
                    int decimalValue =
Integer.parseInt(hexadecimalString, 16);
                    String octalValue =
Integer.toOctalString(decimalValue);
                    if (isNegative) {
                        octalValue = "-" + octalValue;
                    }
                    outputTextView.setText(octalValue);
                }
            }
        });
    }

    private boolean isValidOctal(String octalString) {
        // Regular expression for an octal string with optional
negative sign and fractional part
        String regex = "-?[0-7]+(\\.[0-7]+)?";
        return octalString.matches(regex);
    }

    private boolean isValidHexadecimal(String hexadecimalString) {
        // Regular expression for a hexadecimal string with
optional negative sign
        String regex = "-?[0-9A-Fa-f]+";
        return hexadecimalString.matches(regex);
    }
}
```

Είναι προφανές ότι το πιο δύσκολο μέρος όσον αφορά την κωδικοποίηση και για τα 6 παραπάνω αρχεία java είναι οι αλγόριθμοι που απαιτούνται για την εκτέλεση των υπολογισμών μετατροπής, ειδικά όταν οι τιμές εισόδου περιέχουν τόσο ακέραια όσο και κλασματικά μέρη.

Ακολουθούν τα στιγμιότυπα οθόνης των 6 ζευγών παραγόμενων διατάξεων:



Βήμα 5: Δημιουργία του αρχείου AndroidManifest.xml

Ένα τελευταίο βήμα πριν οριστικοποιήσουμε την εφαρμογή και προχωρήσουμε στις δοκιμές είναι να δημιουργήσουμε το αρχείο AndroidManifest.xml στο φάκελο 'manifests':

```
<?xml version="1.0" encoding="utf-8"?>
<manifest
xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools">

    <application
        android:allowBackup="true"
        android:dataExtractionRules="@xml/data_extraction_rules"
        android:fullBackupContent="@xml/backup_rules"
        android:icon="@mipmap/ic_launcher"
        android:label="@string/app_name"
        android:supportsRtl="true"
        android:theme="@style/Theme.NumeralSystemsConverter"
        tools:targetApi="31">
        <activity
            android:name=".DecimalAndHexadecimal"
            android:label="Decimal and Hexadecimal Conversion"
            android:exported="true">
            <intent-filter>
                <action
android:name="com.example.numeralsystemsconverter.DECIMAL_AND_HEXA
DECIMAL" />
                <category
android:name="android.intent.category.DEFAULT" />
            </intent-filter>
        </activity>
        <activity
            android:name=".OctalAndHexadecimal"
            android:label="Octal and Hexadecimal Conversion"
            android:exported="true">
            <intent-filter>
                <action
android:name="com.example.numeralsystemsconverter.OCTAL_AND_HEXADE
CIMAL" />
                <category
android:name="android.intent.category.DEFAULT" />
            </intent-filter>
        </activity>
        <activity
            android:name=".OctalAndDecimal"
            android:label="Octal and Decimal Conversion"
            android:exported="true">
            <intent-filter>
                <action
android:name="com.example.numeralsystemsconverter.OCTAL_AND_DECIMA
```

```
L" />
                <category
android:name="android.intent.category.DEFAULT" />
            </intent-filter>
        </activity>
        <activity
            android:name=".BinaryAndHexadecimal"
            android:label="Binary and Hexadecimal Conversion"
            android:exported="true">
            <intent-filter>
                <action
android:name="com.example.numeralsystemsconverter.BINARY_AND_HEXAD
ECIMAL" />
                <category
android:name="android.intent.category.DEFAULT" />
            </intent-filter>
        </activity>
        <activity
            android:name=".BinaryAndDecimal"
            android:label="Binary and Decimal Conversion"
            android:exported="true">
            <intent-filter>
                <action
android:name="com.example.numeralsystemsconverter.BINARY_AND_DECIM
AL" />
                <category
android:name="android.intent.category.DEFAULT" />
            </intent-filter>
        </activity>
        <activity
            android:name=".BinaryAndOctal"
            android:label="Binary and Octal Conversion"
            android:exported="true">
            <intent-filter>
                <action
android:name="com.example.numeralsystemsconverter.BINARY_AND_OCTAL
" />
                <category
android:name="android.intent.category.DEFAULT" />
            </intent-filter>
        </activity>
        <activity
            android:name=".MainActivity"
            android:label="Main Activity"
            android:exported="true">
```

```
            <intent-filter>
                <action
android:name="com.example.numeralsystemsconverter.MAIN_ACTIVITY"
/>
                <category
android:name="android.intent.category.DEFAULT" />
            </intent-filter>
        </activity>
        <activity
            android:name=".intro_activity"
            android:exported="true">
            <intent-filter>
                <action android:name="android.intent.action.MAIN"
/>

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

</manifest>
```

Εδώ δηλώνουμε ποια θα είναι η εισαγωγική οθόνη της εφαρμογής μας καθώς και όλες τις υπόλοιπες σελίδες/οθόνες της εφαρμογής μας.

Βήμα 6: Δοκιμάστε την εφαρμογή

Εκτελέστε την εφαρμογή σε έναν εξομοιωτή ή μια συσκευή Android και δοκιμάστε τη λειτουργία μετατροπής εισάγοντας έγκυρους δυαδικούς, οκταδικούς, δεκαδικούς ή δεκαεξαδικούς αριθμούς στις κατάλληλες προβολές EditText και πατώντας το κουμπί "Μετατροπή". Βεβαιωθείτε ότι το αποτέλεσμα της μετατροπής εμφανίζεται σωστά στο TextView "Result" (Αποτέλεσμα).

Συγχαρητήρια! Έχετε πλέον δημιουργήσει μια εφαρμογή Android που μπορεί να μετατρέψει οποιαδήποτε τιμή αριθμητικού συστήματος σε άλλο αριθμητικό σύστημα χρησιμοποιώντας Java.


Binary and Octal Conversion
Δυαδικό σε Οκταδικό
-11.001
Εισάγετε 8-αδικό αριθμό
ΜΕΤΑΤΡΟΠΗ
-3.1
Binary and Decimal Conversion
Δυαδικό σε Δεκαδικό
-101.001
Εισάγετε 10-αδικό αριθμό
ΜΕΤΑΤΡΟΠΗ
-5.125
Binary and Hexadecimal Conversion
Δυαδικό σε Δεκαεξαδικό
-1001.1101
Εισάγετε 16-αδικό αριθμό
ΜΕΤΑΤΡΟΠΗ
-9.D
Octal and Decimal Conversion
Οκταδικό σε Δεκαδικό
-56.75
Εισάγετε 10-αδικό αριθμό
ΜΕΤΑΤΡΟΠΗ
-46.953125
Decimal and Hexadecimal Conversion
Δεκαδικό σε Δεκαεξαδικό
-456.234
Εισάγετε 16-αδικό αριθμό
ΜΕΤΑΤΡΟΠΗ
-1C8.3BE76C8B439
Octal and Hexadecimal Conversion
Οκταδικό σε Δεκαεξαδικό
-343.671
Εισάγετε 16-αδικό αριθμό
ΜΕΤΑΤΡΟΠΗ
-E3.DC8

GOOD LUCK!