SYNTAX COLLEGE

3-DAY EXECUTIVE COURSE

Android OS Overview

How Android really works – architecture, security, IPC, tooling and the security ecosystem – for technical leaders who manage, test or secure Android products without writing Android code.

DURATION	FORMAT	AUDIENCE	LEVEL
3 days	On-site · ~66% theory, ~33% demos & drills	Managers, product, security, QA	Technical – no Android coding

01 About the course

A high-level architectural course that connects Android's system components, work processes and problem-solving aspects – rather than deep coding. Participants learn the ecosystem (AOSP vs GMS, CDD and CTS), the system layers from the Linux kernel through the HAL to the framework, the boot process and System Server, the security model (sandbox, DAC, SELinux, Verified Boot) and Binder IPC.

The third day focuses on what engineering teams actually do: when to use NDK/JNI, the reverse-engineering world (JADX, Ghidra), Google's security ecosystem (Project Zero, Android Security Rewards, monthly bulletins), memory and power management, logging, and Android's platforms – Wear, TV, Automotive and Cuttlefish virtualization. Every day includes live demos and at least one modular hands-on drill.

Who should attend

- Engineering and R&D managers leading Android teams
- Product managers and system engineers working on Android-based products
- Security, product-security and QA/integration leads
- Technical staff moving to Android from other platforms

What participants will be able to do

- Explain Android's architecture and ecosystem to stakeholders
- Understand the security model and its implications for your product
- Read a manifest and drive components with ADB
- Understand what the team does: native code, IPC, reverse engineering
- Ask the right questions in design and code reviews

02 Day-by-day syllabus

DAY 1 Ecosystem, architecture & system layers

- What is Android? History, business model, AOSP vs GMS
- CDD and CTS – how hardware and software compatibility is ensured
- The stack: Linux kernel, HAL, native layer, Java/Kotlin framework
- The four app components: Activity, Service, Broadcast Receiver, Content Provider
- The manifest as the app's identity document; permissions management

HANDS-ON LABS

- **Driving Android with ADB** (~90m · hands-on)
  Read an app's manifest, launch screens from the terminal, start services and watch the foreground-service promise break, send broadcasts and hit the system's limits.
- **Target SDK & fragmentation** (bonus · hands-on)
  See how the same app behaves differently across Android versions and target-SDK levels.
- **Hunt the manifest of a real app** (bonus · solo)
  Pull an installed app from the device and map its components and permissions.

> **Bonus drill:** Every mission is demo + solo, so mixed-level groups can follow at their own pace.

DAY 2 Security, communication building blocks & system services

- Boot at a high level: bootloader to Zygote; app forking and runtime
- System Server and critical services (AMS, WMS)
- Privilege separation: sandboxing, Linux UID, DAC
- SELinux for managers: MAC, enforcing vs permissive
- Verified Boot and the chain of trust
- Binder & IPC – proxy, stub, transactions as potential bottlenecks; AIDL

HANDS-ON LABS

- **Boot chain & System Server** (~15m · instructor demo)
  Boot order proven by PIDs, boot properties via getprop, Verified Boot state, the System Server Binder service catalogue, and calling a Binder service directly.
- **Feel the permission chain** (~25m · hands-on)
  Find your UID and hit the wall, elevate, read live SELinux denials in the audit log, toggle enforcing/permissive, and prove DAC and SELinux are independent layers.
- **AIDL – drive a cross-process call** (~25m · hands-on)
  Install a client and a server app, find the bindable service, connect them, then kill the server and watch the client surface the disconnect – the whole Binder model on one screen.

DAY 3 Development management, infrastructure & advanced tools

- When to use NDK/JNI – native C++ vs Java/Kotlin trade-offs
- Reverse engineering overview: JADX and Ghidra
- Google Project Zero and the 90-day disclosure policy
- Android Security Rewards and monthly security bulletins (CVEs)
- Kernel changes: Low Memory Killer and wakelocks
- Logging and fault analysis with Logcat
- Platforms: Wear, TV, Android Automotive; Cuttlefish virtualization

HANDS-ON LABS

- **Think like a pentester – 4 vulnerabilities** (~45m · hands-on · aapt2, curl, JADX, apktool)
  A wide-open cloud database, code downloaded and run at runtime, a privileged action behind an exported component, and why "we obfuscated it" is not protection – ending in a findings report.
- **Hunt the hidden secrets** (~45m · hands-on · JADX, CyberChef, Ghidra)
  Find disguised secrets in Kotlin and native code and crack them – learning to tell encoding from encryption at a glance.
- **Into the native** (bonus · guided)
  An app that talks straight to the hardware – descending the layers and seeing which DAC and SELinux protections were bypassed.

8	3	~33%	6+	0	1
drills & demos	days	hands-on	lab apps	coding required	findings report

What participants receive

- Day-by-day presentations
- Step-by-step drill pages with command reference cards
- Purpose-built lab apps (AIDL client/server, vulnerable apps, native probe)
- Glossary of Android terms
- Software requirements sheet sent in advance
- Answer keys reviewed in class

03 Format & logistics

Length	3 full days
Method	~66% theory · ~33% demos & drills
Language	Hebrew or English
Materials	Presentations, glossary, drills and sample apps

Requirements

- A training room with a projector or large screen
- Laptops with ADB and JADX installed (requirements sheet sent in advance)
- No Android coding background required

- Drills are modular – suitable for mixed technical levels

The course can be shortened to 1–2 days for management audiences or extended with deeper security content.

04 Your instructor

All Syntax College courses are designed and taught personally by **Eran Katsav**, founder of Syntax College. Eran holds a B.Sc. in Computer Science from the Hebrew University of Jerusalem and an M.Sc. with honors from Tel Aviv University, and has more than 20 years of experience training engineers in industry, the defense sector and academia.

He lectures on native Android development in Kotlin and Java at Reichman University and at the Holon Institute of Technology (HIT), where he also heads the mobile lab. As an independent developer he created **Bip it**, the first Hebrew voice assistant, which reached about one million downloads in Israel; white-label versions of its engine ship in all Suzuki vehicles sold in Israel.

Selected clients: **Samsung, General Motors, Cisco, ironSource, CWT, Ophir Optronics**. Authorized supplier of the Israeli Ministry of Defense.

Bring this course to your team
Tailored syllabus, on-site at your offices, in Hebrew or English.

Eran Katsav
+972-50-6944922
info@syntax.org.il
www.syntax.org.il