PHẦN 1 — CÂU HỎI DỄ (Warm-up)

---

1. Activity là gì?

Đáp án: Activity là một màn hình giao diện mà người dùng tương tác trực tiếp. Nó quản lý UI và logic của một screen trong ứng dụng Android.

---

2. Khi nào `onPause()` được gọi?

Đáp án: Khi Activity bị che một phần (mở dialog, chuyển sang Activity khác) và mất focus tạm thời.

---

3. Fragment là gì?

Đáp án: Fragment là một phần giao diện có vòng đời riêng nhưng phải nằm trong Activity. Thường dùng trong bottom navigation, viewpager hoặc Single Activity Architecture.

---

4. Explicit Intent là gì?

Đáp án: Intent chỉ rõ component đích (Activity/Service cụ thể). Dùng trong nội bộ ứng dụng.

---

5. Implicit Intent là gì?

Đáp án: Intent mô tả hành động, hệ thống sẽ tìm app có thể xử lý (vd: mở web, gọi điện, chọn ảnh).

---

6. ViewBinding dùng để làm gì?

Đáp án: Truy cập view theo kiểu an toàn type-safe, tránh `findViewById`. Không hỗ trợ binding logic vào XML.

---

7. RecyclerView dùng để làm gì?

Đáp án: Dùng để hiển thị danh sách lớn với hiệu năng cao nhờ tái sử dụng ViewHolder.

---

8. DiffUtil là gì?

Đáp án: Lớp tính toán sự khác nhau giữa 2 danh sách → dùng để cập nhật RecyclerView mượt mà mà không cần `notifyDataSetChanged()`.

---

9. SharedPreferences dùng để làm gì?

Đáp án: Lưu dữ liệu dạng key-value nhẹ nhàng như token, flag, cấu hình đơn giản.

▨ PHẦN 2 — MỨC TRUNG BÌNH (phỏng vấn Fresher/Junior)

---

10. Khác nhau giữa Activity và Fragment?

Đáp án:

- Activity: màn hình độc lập, khai báo trong manifest.
- Fragment: phụ thuộc Activity, UI con bên trong Activity.
- Fragment có thể xoay vòng, replace, reuse linh hoạt hơn.

---

11. Khi nào dùng Service thay vì Thread?

Đáp án: Khi cần tác vụ chạy lâu **kể cả khi user thoát khỏi app**, ví dụ:

- nhạc nền
- upload/download dài
- GPS tracking

Thread chỉ chạy khi process hoạt động.

---

12. ContentProvider dùng để làm gì?

Đáp án: Chia sẻ dữ liệu giữa các app thông qua URI (vd: danh bạ, ảnh, media). App cũng có thể tạo provider riêng.

---

13. Vì sao background service bị hạn chế từ Android O?

Đáp án: Để tiết kiệm pin và tài nguyên. Android O yêu cầu:

- foreground service
- WorkManager
- JobScheduler

→ không cho phép app chạy background lâu tùy ý.

---

14. Vì sao cần runtime permission?

Đáp án: Từ Android 6+, để tăng bảo mật. Người dùng phải tự cấp quyền camera, location… thay vì cho phép mặc định lúc cài.

---

15. Khác nhau giữa static broadcast và dynamic broadcast?

Static Broadcast

- Khai báo trong Manifest
- Nhận sự kiện khi app chưa chạy
- Nhiều broadcast implicit bị hạn chế từ Android 8+

Dynamic Broadcast

- Đăng ký trong code (runtime)
- Chỉ hoạt động khi app đang chạy
- Nên dùng cho hầu hết trường hợp hiện nay

---

16. DataBinding khác ViewBinding?

Đáp án:

- DataBinding: bind logic trực tiếp trong XML (`@{}`), hỗ trợ MVVM.
- ViewBinding: chỉ truy cập view an toàn, không có binding expression.

---

17. ViewHolder trong RecyclerView là gì?

Đáp án: Class giữ reference tới view để tái sử dụng → giảm gọi `findViewById`, tăng hiệu năng.

---

18. WorkManager dùng để làm gì?

Đáp án: Chạy tác vụ nền đảm bảo, kể cả khi app bị kill hoặc reboot (upload, sync…).

▧ PHẦN 3 — CÂU HỎI KHÓ (Junior–Mid / kỹ năng sâu)

---

19. Giải thích vòng đời Fragment?

Đáp án tóm gọn chuẩn phỏng vấn:

- `onAttach()`
- `onCreate()`
- `onCreateView()`
- `onViewCreated()`
- `onStart()`
- `onResume()`
- `onPause()`
- `onStop()`
- `onDestroyView()`
- `onDestroy()`
- `onDetach()`

Điểm quan trọng: UI được tạo ở `onCreateView()` và bị hủy ở `onDestroyView()` → khác Activity.

---

20. Làm sao để tránh memory leak trong Android?

Đáp án:

- Không giữ reference đến context không cần thiết
- Dùng applicationContext nếu phù hợp
- Hủy listener khi Activity/Fragment destroy
- Tránh lạm dụng `static`
- Sử dụng lifecycle-aware component
- Dùng LeakCanary để kiểm tra

---

21. Tại sao FragmentTransaction phải gọi commit() sau cùng?

Đáp án: Vì commit() ghi thay đổi lên backstack và UI. Nếu thao tác giao diện xong mà quên commit → Fragment không được hiển thị.

---

22. Khi nào nên dùng `getApplicationContext()` thay vì `this`?

Đáp án: Khi cần context tồn tại lâu hơn vòng đời Activity (vd: Database, Singleton). Không dùng cho UI vì applicationContext không có theme.

---

23. Sự khác nhau giữa Handler, Coroutine, ExecutorService?

- **Handler:** xử lý message queue + runnable trên thread
- **ExecutorService:** quản lý pool thread
- **Coroutine:** bất đồng bộ nhẹ, dễ quản lý lifecycle, ít tốn tài nguyên nhất

---

24. Tại sao không dùng `notifyDataSetChanged()`?

Đáp án: Vì gây refresh toàn bộ list → mất animation, không tối ưu. Nên dùng:

- DiffUtil
- ListAdapter

---

25. LiveData vs StateFlow?

Đáp án:

LiveData	StateFlow
lifecycle-aware	không lifecycle-aware (nhưng an toàn hơn)
dùng nhiều với ViewModel cũ	modern (flow-based)
không hỗ trợ backpressure	flow hỗ trợ

26. onSaveInstanceState() được gọi khi nào?

Đáp án: Khi Activity có thể bị hủy do cấu hình thay đổi (xoay màn) hoặc hệ thống thu hồi RAM. Dùng để lưu state tạm thời.

---

27. Application class dùng để làm gì?

Đáp án: Khởi tạo global dependency như:

- Retrofit instance
- Database
- WorkManager
- Logging setup

---

28. Sự khác nhau giữa Service, ForegroundService, IntentService?

Đáp án:

- **Service:** chạy ngầm, không tự tạo thread
- **ForegroundService:** có notification, ưu tiên cao
- **IntentService:** chạy trên background thread + tự dừng → đã deprecated, giờ dùng WorkManager/Coroutine.

---

29. Tối ưu RecyclerView như thế nào?

Đáp án:

- Dùng DiffUtil/ListAdapter
- Dùng setHasStableIds(true)
- Tối ưu layout (ConstraintLayout)
- Giảm nested layout
- Dùng Paging library nếu list lớn

---

30. Scoped Storage là gì?

Đáp án: Cơ chế truy cập file an toàn từ Android 10+. App chỉ xem được folder riêng của mình, muốn truy cập ảnh/video phải dùng MediaStore hoặc SAF.
