Guía de preguntas de entrevistas

Objetivo del screening técnico:

Hacer la mayor cantidad de preguntas posibles (acorde al seniority), para saber si amerita bajarlo del proceso anticipadamente. En el caso de que continúe el proceso, en la entrevista niveladora, se abordarán algunos de estos temas con mayor profundidad. Por este motivo las preguntas de este documento, deberían ser lo más cerradas y concretas posibles, para que una respuesta no nos consuma la media hora asignada.

Preguntas:

### Diseño y Diseño OO

1. ¿Qué entiende por objeto, clase, método, herencia, polimorfismo, interfaces, abstracción?
2. ¿Qué es el método equals? ¿Cuáles son las condiciones que debe cumplir? Diferencia entre equals e ==
3. ¿Qué son los métodos estáticos? ¿Dónde y para qué los usaría?
4. ¿Cuál es la diferencia entre static y final?
5. Java > 8 uso de default methods en interfaces ¿Para que lo usaría?
6. ¿Qué es polimorfismo? Dar un ejemplo
7. ¿Que es un objeto Singleton? ¿Cómo lo diseñaría?
8. Diferencia entre statement y expression
9. Diferencia entre mutable e inmutable, que ventaja/desventaja tiene cada uno
10. ¿Para qué sirven los patrones de diseño? Tipos? Utilizó alguno? Ejemplifique
11. Herencia o composición, ¿por que elegiría uno u otro? Ventajas y desventajas
12. ¿Conoce qué es Open Close Principle?
13. ¿Conoce qué es un Code Smell? Si sabe, cuales conoce, y que haría para arreglarlo? Si no conoce, mostrarle algo de código para que nos indique qué problema ve
14. ¿En qué consiste la práctica de clean code? ¿La utiliza?
15. ¿Qué significa hacer un refactor? ¿Hizo alguna vez uno? ¿Qué implicancias tiene o que tiene que tener en cuenta antes de empezar? ¿Que refactors puede hacer que esta 100% seguro que no rompe nada?

Diseño (Programación Funcional)

1. ¿Conoce algún patrón de diseño funcional? ¿Qué es un lambda? ¿Utilizó alguna vez uno? ¿Conoce qué es una monada y un functor?
2. ¿Que es un pattern match?
3. Que es un Algebraic Data Type
4. Que es una higher order function
5. Si la programación funcional propone no tener efecto de lado, como resuelve el problema de la interacción con un sistema? Conoce lo que son los effects? Como diferenciar entre un “efecto” y una función pura con solo ver la firma? (def method: Unit => efecto de lado 100% marcado).
6. ¿Que ventaja/desventaja le ve a la programación funcional vs la orientada a objetos?
7. ¿Como garantiza la programación funcional que no haya efecto de lado dentro de una función? (inmutabilidad)

Lenguajes

1. ¿Conoce o utiliza algún otro lenguaje además de Java? Si lo hace, para que casos de uso usaría cada uno?
2. Si le proponen utilizar algún lenguaje que no conoce, ¿qué haría para aprenderlo?
3. ¿Qué debería evaluar cuando sale un nuevo lenguaje al mercado?

### Colecciones

1. ¿Qué diferencia hay entre un List, un Set, un Vector y un Map?
2. ¿Qué diferencia hay entre un ArrayList y un LinkedList? (ventaja y desventaja de cada uno)
3. ¿Cómo implementarías un ArrayList?
4. ¿Cómo funciona un HashMap?
5. ¿Cuál es la diferencia entre HashMap y ConcurrentHashMap?
6. ¿Por qué se llaman Hash? ¿Qué las caracteriza? ¿Para qué sirve una hashCode? ¿Qué relación tiene con el equals? ¿Qué es una colisión? ¿Cómo resuelve HashMap las colisiones?
7. ¿Qué es un Comparator? ¿Dónde los usaría? ¿Cuál es la diferencia entre y Comparable?
8. ¿Cómo funciona una búsqueda binaria? ¿Cuál es la condición necesaria?

### Exceptions / Errors

1. ¿Qué es una Exception? Y un Throwable?
2. ¿Qué es un Error?
3. ¿Cómo es la jerarquía de Excepciones? ¿Qué diferencia cada “rama” de la jerarquía?
4. ¿Cuándo catchear y cuándo no?
5. Diferencia entre checked y unchecked exception
6. ¿Qué consecuencias tiene utilizar try catch? ¿Cómo las puedo solventar?
7. Si tiene varios request concurrentes, ¿cómo haría para loguear un error?
8. Loguear el stack trace tiene algún impacto de performance cuando el logger está apagado por configuración?
9. Escribir log a disco, tiene algún costo para la aplicación?

### Concurrencia (solo lo simple)

1. ¿Qué es un Thread?
2. ¿Qué diferencias hay entre extender de Thread e implementar Runnable y cuál considera que es la mejor opción?
3. ¿Qué es un ThreadPoolExecutor? ¿Cómo se instancia? ¿Cuáles son los parámetros básicos de la configuración de un ThreadPoolExecutor? Cuando se termina la app, ¿qué habría que tener en cuenta respecto al threadpool?
4. ¿Qué es un ThreadLocal?
5. ¿Future?
6. ¿Qué es un lock? ¿Qué es un deadlock? ¿Qué es un livelock? ¿Qué es starvation?
7. ¿Optimistic lock vs pessimistic lock?
8. ¿Qué pros y que contras tiene el uso de multithreading?
9. Diferencia entre Runnable y Callable
10. Nombrar las colecciones threadsafe que conozcas
11. ¿Qué mecanismos de control de la concurrencia conocés? ¿Dónde usarías uno u otro?
12. ¿Que es una condición de carrera?
13. Diferencia entre asincronismo, concurrencia y paralelismo. ¿Qué estrategia de paralelismo conoce?
14. Que modelos de concurrencia conoce (shared memory, actor model, csp)
15. Que es un context switch y como afecta nuestra aplicación
16. ¿Conoce el concepto de CPU / IO Bound? Si tengo una app CPU Bound, ¿cómo conviene configurar el threadpool? Y si es IO Bound?
17. Dado un productor y un consumidor, si el productor produce inesperadamente a mayor ritmo que el consumidor, que problema podría llegar? ¿Cómo puedo evitarlo?
18. ¿Conoce el concepto de circuit breaker? Si no lo conoce, explicar que es y preguntar como lo diseñaría si tuviera que hacer uno.
19. ¿Que es un Stream? ¿Qué ventaja tiene y cuándo lo usaría? ¿Si lo usó, que sería un Source, un Flow y un Sink? ¿Que sería un graph?
20. ¿Conoce el ForkJoinPool? ¿Qué ventaja tiene frente a otros Pools? ¿Cuando lo utilizaría?

### Memoria

1. ¿Cómo se divide la memoria de la JVM?
2. ¿Que es la heap, y el stack, y que sería la permgen?
3. ¿Conoce que significa commited memory y por qué lo necesita la jvm?
4. ¿Qué es el garbage collector? ¿Cómo funciona en cada espacio de memoria?
5. ¿Qué herramienta usás para monitorear el uso de memoria de la JVM?
6. ¿Qué es un memory leak? ¿Cómo lo detectás?
7. ¿Qué es una pausa y cómo afecta nuestra app cuando ocurre?
8. ¿Con qué parámetros se configura la JVM?
9. Algoritmos de GC, cuando usaría uno u otro (throughput o latencia?)
10. Off heap, conoce el término y para que lo usaría?
11. Si necesitaras hacer algo sin pausa, ¿qué lenguaje y porque seleccionarías para escribir código?

### Testing

1. ¿Qué tiene que tener un test para ser verdaderamente UNITARIO? ¿Qué diferencias tiene con un test de integración? Y un test de regresión?
2. ¿Qué necesita tener o no tener una clase para ser testeable?
3. ¿Qué es mocking? ¿Para qué y cómo se usa?
4. Inyección de dependencias, para que? Qué mecanismos conoces para inyectar dependencias?
5. Alguna vez hiciste TDD? Ejemplo

### MVC

1. ¿Qué es? ¿Cuál es el objetivo que persigue MVC?
2. ¿Qué responsabilidad debería tener un controller?
3. De qué tipo es la relación entre un model y la view?

Comunicación entre apps (solo lo simple)

1. ¿Qué protocolos de comunicación conoce? (TCP / UDP), características de cada uno. Es más simple de lo que parece
2. ¿Qué es un socket? ¿Conoces algún problema relacionado con sockets? Evaluar si conoce BindException, SocketTimeoutException (read to, connection to)
3. ¿A qué se le llama una api “restful”? ¿Qué verbos existen y para que se usan? ¿Qué status conoces y que significan (2xx, 4xx, 5xx)? ¿Cómo garantiza no romper su propia API?
4. ¿Qué tipo de contenido (content-type) se suele usar en Tal vez el concepto de equals y == no lo explico muy bien, pero parecia entender en lineas generales. ?
5. Si tiene que introducir un cambio en su API y tiene que romper la retrocompatibilidad, ¿qué haría en ese caso? (versionado, hablar con los clientes)

### Monitoreo

1. ¿Qué herramientas usas para monitorear tu app/server en producción? (Herramientas)
2. ¿Qué métricas miramos de una app? (throughput, latency, errors)
3. ¿Qué métricas miramos de un host? (load, mem, cpu)
4. ¿Sirve usar promedios en latencia? ¿Sirve usar promedios en Throughput?
5. ¿Si tuviera un problema de tiempos dentro de su aplicación, que harías para encontrar el problema? ¿Conoce alguna herramienta de profiling?
6. Si tuvieras un sistema compuesto por muchas aplicaciones, y cada una tiene su propio log, como harías para ver que pasó con un request a medida que fue pasando por los distintos sistemas? ¿Se te ocurre alguna forma?
7. ¿Que es la run queue? ¿Cómo haces para verla en un host? (vmstat)
8. Para que usarias netstat, lsof?

Bases de datos

1. ¿Qué es una base de datos No relacional? ¿Qué diferencias hay con una base de datos relacional?
2. ¿Qué ventajas aporta una base de datos no relacional?
3. ¿En qué casos usarías un Cassandra? ¿Y un MongoDB? ¿Y un Elasticsearch? ¿Y un Redis? ¿Zookeeper? ¿En qué casos no los usarías?
4. ¿Qué ventajas y desventajas tiene el uso de índices en una base de datos relacional?
5. ¿Qué son los isolation levels? Propagation vs Isolations
6. Explicar los conceptos de Sharding y Replication, qué problemas resuelve cada uno
7. ¿Conoce que es un split brain?
8. ¿Conoce Scylladb?
9. ¿Conoce Timeseries DB, cuál sería su característica principal?
10. Conoce el teorema CAP, si lo conoce, como determinaría el uso de cada uno?

Frontend (podes evitarlo)

1. ¿Qué es ajax?
2. ¿Qué diferencia un request Ajax de un request convencional en un browser?
3. ¿Cuántas operaciones en paralelo puede realizar un browser usando javascript? ¿Por qué? ¿Cómo funciona el event loop de javascript?
4. ¿Qué es polling? ¿Para qué lo usarías?
5. ¿Qué es un long request? ¿Para qué lo usarías?
6. ¿Qué es una promesa?
7. ¿Qué diferencia hay entre una página “tradicional” y una SPA? ¿Y una WPA?
8. Pros y cons de SSR and CSR
9. ¿En qué consiste la métrica DOMContentLoad y Load?
10. Explicar prototype chain. Dar un ejemplo simple
11. ¿Qué herramientas de test de performance conocés?
12. ¿Qué herramientas de buildeo conoces?

Seguridad / autenticación / autorización

1. ¿Qué factores tenés en cuenta para la seguridad de tus aplicaciones?
2. ¿Qué es un SQL Injection? ¿Cómo lo evitas?
3. Validaciones por whitelist vs blacklist. ¿Que approach conviene usar y porque?
4. ¿Qué es un ataque DOS? ¿y DDOS? ¿Se puede evitar? ¿Cómo?
5. Que diferencia hay entre autenticación y autorización?

Docker (tranqui con esto, pero es simple)

1. ¿Qué es un container? ¿qué ventajas/desventajas tiene?
2. Para qué sirve un Dockerfile ¿Qué instrucciones puede contener?
3. Por qué se suelen agrupar varios comandos dentro de un RUN en un Dockerfile
4. Docker-compose y swarm

Training

1. ¿Cómo se mantiene actualizado?
2. ¿Cuál fue el último libro técnico que leyó? ¿Y cuál el que más le gusto?
3. ¿Sigue algún blog? ¿cual?
4. Si tiene que aprender una herramienta nueva, ¿cómo hace para aprenderla?
5. Alguna vez tuvo que leer o curiosidad por leer como funciona alguna de las librerías que usa? Si lo hizo, recuerda cual leyó y que le sorprendió o disgustó?
6. ¿Le dio alguna vez un training a otros, como lo hizo?

Anexo: Preguntas Devops (esto revisalo, hay cosas interesantes)

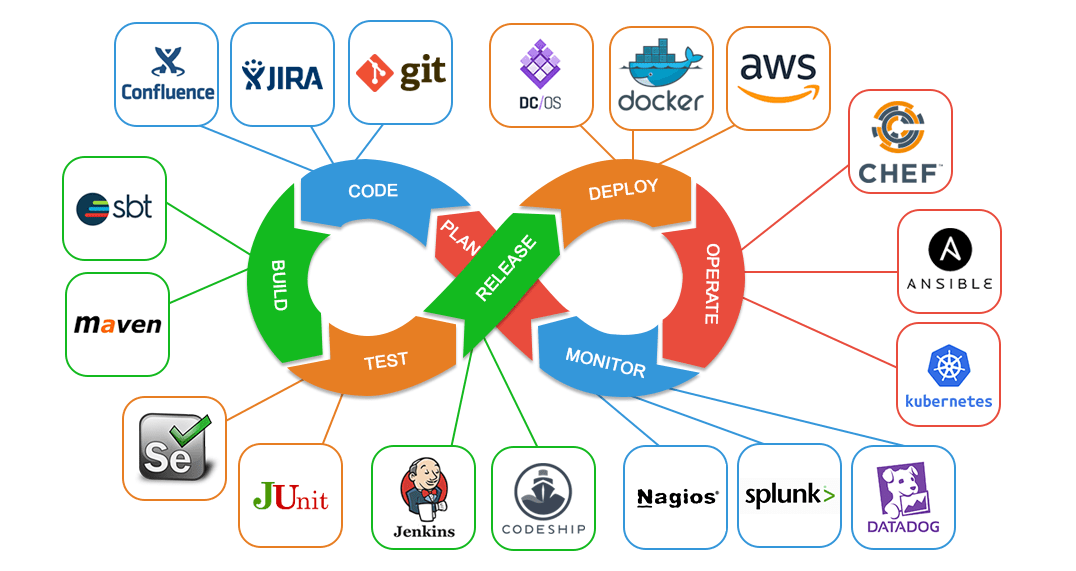
Infra

1. ¿Qué es un load balancer?
2. ¿Qué estrategias de balanceo implementarías si tuvieras que desarrollar un load balancer? ¿Qué ventajas y desventajas tiene cada estrategia?
3. ¿Cómo lográs tener alta disponibilidad en una aplicación? ¿Cómo te ayuda un load balancer?
4. ¿Cómo hacés un deploy sin downtime? ¿Y un gracefull stop?
5. ¿Qué es un DNS? ¿Como se configura?
6. ¿Cómo es la interacción entre una aplicación que quiere hacer un request y el server de DNS?
7. ¿Qué es una cola de mensajes? ¿Cómo funciona? ¿Para qué la usarías?
8. ¿Qué pasa cuando una cola de mensajes tiene múltiples consumidores?
9. ¿Qué es un lock distribuído? ¿Para qué lo usarías? ¿Con qué herramienta lo implementarías?
10. ¿Se puede asegurar el orden de los mensajes en una cola?.

AWS

* ¿Qué tipo de instancias conocés? ¿Qué factores afectan la elección?
* ¿Qué funcionalidades brinda un Autoscaling group?
* Diferencia entre On-Demand, Spot, Reserved
* ¿Qué es una VPC?
* ¿Qué genera costos en AWS?

### DevOps tools



**Anexo**:

**Ejercicios de Diseño OO**

1. Nos piden poder diseñar una aplicación que nos permita ordenar el resultado de una búsqueda de hoteles. La búsqueda va a contar con datos de un codigo de país, un id de destino fecha de entrada y salida, y una lista de hoteles con id, precio. Para todos los países se pide ordenar y loguear:

* destino RIO ordenar de menor a mayor precio.
* destino BUE, ordenar de mayor a menor precio.

Pensar el diseño para poder agregar otros mecanismos de ordenamiento.

2. Nos piden un cambio, para destino CUN, nos piden devolver el mismo orden, pero tenemos que poder loguear como si hubiera sido ordenado por precio.

## NARANJA X

En el ultimo trabajo que trabajaste, ¿cuantas personas habia en el equipo?

¿En cuantos proyectos te toco trabajar en equipo, o hubo un proyecto donde trabajaste solamente tu?

¿como te organizabas las tareas en tu ultimo trabajo?

Con respecto a Spring Boot ¿Te toco hacer toda la vida de un servicio? ¿desarrollaste un API? ¿como lo hiciste?

Si tienes que hacer algo nuevo para resolver un problema que no tiene mucha documentacion, como lo resolverias?

¿Conoces kafka? ¿que hiciste?

¿Que bases de datos utilizaste?

En cuanto a Servicios REST, ¿implementaste todos los tipos de verbos?

¿Usaste gitlab o github?

¿tuviste oportunidad de usar el pipeline de gitlab?

¿conoces Docker? ¿que hiciste?

¿Que patrones de Diseño utilizaste y como los usaste?

¿como crearías un servicio en SpringBoot?

¿Conoces la arquitectura hexagonal?

¿Trabajaste con AWS?

¿Trabajaste con stacks en AWS?

¿Usaste el Serverless Framework de AWS?