Interview Prep Packet

This packet is your detailed guide to landing a good Android Development job. You will find multiple sections, each addresses a different step in the process or a specific way to prepare. Start by identifying which parts you feel strong in and which you believe you'll need to spend more time on. Then draft up a plan to work accordingly. When you've completed your plan, post it to the discussion board so you can get feedback from others.

Part 1: Talk The Talk

You will want to be familiar enough with the following terminology to describe what it is in relationship to other terminology in the lists. Wikipedia articles like this one on Java and this one on Android are a good place to find many of these terms used in context. But you should already be familiar with most of them through your participation in this course.

	Java	
Arrays	Variables/Member/Instance/ Local	Garbage Collection
Objects/Instances	Loops/Conditional Statements/Control Flow	Primitives/Data Types
Classes/Nested/Abstract	Interfaces	Threads
Methods/Parameters	Generics	Polymorphism
Pointers	Constructors	Libraries
Hashmaps	Fields	Callbacks
Collections	Static	Exceptions
Lists	Packages	Typecasting
Inheritance	Blocks	Recursion
Scope	Operators	Auto-boxing

Platform/IDE/SDK	Android
Android Studio	Activity Lifecycle
GitHub	Views/Layouts/View Hierarchy
Genymotion/Emulators	Fragment Lifecycle
JVM	Android Manifest
version control/git	ListView
Debugger	Services
JSON	AsyncTask
Compiler	Intents/Intent Filters
Run-time	Notifications
Dalvik	Event Listeners
XML	Broadcast Receivers
Databases/SQLite	Custom Adapters
APK	Volley
API	Drawables
RESTful Architecture	Content Provider
	ActionBar
	Toasts

You'll also want to review your blog posts or the notes you took during the course from start to finish. Notice your transformation and the evolution of your thinking. Write a public blog post that describes your transformation from novice to professional and share the link with our online community. This will help you establish your online identity as an Android developer and can become part of your narrative for potential employers.

Part 2: Walk The Walk

You will want to have some code you've written and can talk eloquently about when you are meeting with other developers. This will be helpful in interviews and when meeting people at networking events. Below is our recommendation on the best way to prepare.

1) Work on a Weather App (it's a great all-purpose app that will challenge your skills in many areas, we recommend http://forcast.io as the API to use.)

- Comment your code
- Commit it to Github

2) Work on a personal project that you design for yourself

- Complete your personal project App
- Comment your code
- Commit to Github

The purpose of building these applications is to demonstrate to others that you understand the entire development process, that you can clearly articulate what your code is doing, and that you can write comments and use naming and syntax conventions that will make you a good fit with other developers. When you have them up on Github, practice walking through the code and describing what each line does. When you feel comfortable walking through it yourself, try explaining it to a peer or fellow developer. Make sure to share the Github project links with us so everyone can benefit from each other's efforts.

Part 3: The Interview Process

First Contact

You will want to have some code you've written and can talk eloquently about when you are meeting with other developers. This will be helpful in interviews and when meeting people at networking events. Below is our recommendation on the best way to prepare. The initial contact can be made one of two ways. The first is digitally through email, twitter, LinkedIn or online application. The second is through a face to face meeting at a networking event or through an introduction by your personal or professional network.

If it's an online application, be direct about what you're looking for and your background, skills, etc. Use your LinkedIn resume if possible. If it's an informal introduction from within your network, be polite and make sure to thank whomever introduced you. You also want to be brief and to the point by suggesting times to meet. Here's a sample contact email after an introduction from someone you meet at an event:

Sasha, thanks for the intro!

Zach, I'm in San Francisco this week and next. But I can do Skype or phone. Let me know what works for you. Thursday, Sunday, and Monday are the days I have the least scheduled.

Looking forward to it,

Adam
512-555-9797

Screening Interview

The screening interview is just a verification that you are a fit for the organization's current openings. They want to make sure that your salary requests are in alignment with what they are paying and that you are legally allowed to work in the US. This interview is usually handled by the HR representative but in smaller companies may be done by the same person that handles the cultural interview described below. This interview may also get bundled with any of the face to face interviews below.

Common Questions:

1. Are you a US (or country you're applying in) citizen?
2. Are you legally allowed to work in this country?
3. What job are you applying for?
4. Who referred you to this opening?
5. What is your previous work experience?
6. What are your contact details?
7. Is there anything you'd like to add to your application or resume at this time?
8. When are you available for additional interviews?
9. Are you able to meet face to face?
10. What are your minimum salary requirements?

Preparation:

There's not a lot of preparation required for this one. Just don't get caught off guard with the salary question. Respond to that one with a question, something like "what is your company's preferred pay range for this level of Android developer?" Then respond with: "that range works for me."

Technical Interview

The technical interview is your opportunity to "geek out" about Android development and the Java programming language. The person conducting this interview is usually looking for someone who knows their stuff, is curious about what they don't know, and can communicate on their level. This doesn't mean you have to be a know it all. Some people you interview with may try to convince you that they are. If they do, just take the high ground and let them know that you respect their knowledge and authority. This interview will usually take place before or during the coding interview. It will most likely be conducted by a senior developer who may be your direct supervisor if you get the job.

Common Questions:

1. Describe the Activity Lifecycle?
2. What is a Fragment and what are they used for in Android?
3. What is an Intent and how are they used?

4. Does Java pass by value or by reference?
5. How is threading handled in Java?
6. Describe how you would sort an array of numbers in increasing order in Java.
7. What is an exception and how should it be used?
8. What is polymorphism?
9. Describe how memory is managed with the JVM.
10. What is a Java Class?
11. What is an Interface in Java?
12. How does a Class differ from an Interface?
13. Describe the differences between string and String, int and Integer, boolean and Boolean?
14. What does the *abstract* keyword do?
15. True or false: every class in Java is of base type Object?
16. How many interfaces may a Java Class inherit from?
17. How many interfaces may a Java Class implement?
18. What is a package?
19. What is an Object?
20. What does the *synchronize* keyword do?

Preparation:

The most important things to remember for this interview are 1) be curious about everything they ask, 2) sound knowledgeable and confident about the things you know, and 3) treat the interviewer with respect and intelligence.

To prepare for (1): Find the part about the jargon and the concepts that really intrigues you and explore it before you go into the room. For example, what fascinates you more, Java or Android? Get excited about both, but really embrace what you love and make a list of all the topics (5-7, no more) you want to learn more about. Start learning about them.

To prepare for (2): Answer the 20 questions above. Practice saying your answer out loud. Do not read from wikipedia or stack overflow, come up with your own wording and practice saying it aloud until it feels comfortable coming out of your mouth.

To prepare for (3): Just remember that the interviewer doesn't really care about the interview as much as he/she cares about getting someone who can contribute to the team. Show them deference and respect but not passive acquiescence. Think of the kind of person you would want to work on a project with and how they would treat you. Treat the interviewer the same way.

<u>Cultural Interview</u>

During the cultural interview they will usually volunteer what their culture is like and ask you if you could see yourself thriving in it. The answer is always an enthusiastic yes. But you can't just leave it there, you have to follow up with some of your own personality and maybe a story or two.

Common Questions:

1. Why do you want to work here?
2. Tell me a time when you had to solve a difficult problem and how you handled it.
3. Describe your ideal working environment.
4. Do you like to work alone or with others?
5. What's a project you've worked on with other developers and/or designers?
6. Have you ever done any XP/Agile/Scrum Development?
7. Can you handle intense projects/deadlines?

Preparation:

Prepare a couple stories about something you've built and how you went about building it (with others and/or alone). Practice telling the story to your friends and family. See how much enthusiasm and nerdy curiosity you can put into it.

The cultural interview is also a big opportunity for you to learn more about the interviewer and the company. Remember to ask these magic bullet questions: "Why do you like working here?" and "how long did it take for you to get acclimated?"

Coding Interview

This is the most important interview. The coding interview will usually involve an algorithm problem for you to solve in Java. It may take place face to face on a white-board using pseudo code or remotely using a text editor. It may even take place during your technical interview using Android Studio. In general, you will not have access to the web or be able to "look stuff up." This is a way for you to demonstrate your thought process for solving problems and your proficiency with the Java language. The best way to prepare for your coding interview is to do practice problems at Coding Bat. The ones below are our suggestions, but feel free to explore around.

Common Questions:

1. http://codingbat.com/prob/p173784
2. http://codingbat.com/prob/p196441
3. http://codingbat.com/prob/p101887
4. http://codingbat.com/prob/p136041

5. http://codingbat.com/prob/p118044
6. http://codingbat.com/prob/p174148
7. http://codingbat.com/prob/p137136
8. http://codingbat.com/prob/p117019
9. http://codingbat.com/prob/p139411
10. http://codingbat.com/prob/p183649
11. http://codingbat.com/prob/p134022

Preparation:

The most important thing to do in this interview is approach solving a problem with comfort and curiosity. You want to demonstrate ease and familiarity with solving a problem using code. You don't have to get it perfectly right, but you do have to show that you are experienced and methodical. Practice the Algorithm Problem Solving Process described below. It's scientifically designed to be the most extensible and reliable approach to solving these problems currently known.

Learnovate Lab's Scientifically Engineered Algorithmic Problem Solving Process:

Step 1: Identify
(1) Identify what you understand
(2) Identify what you don't understand

Step 2: Method
(1) Create a method signature complete with name, parameters, and return type. Also include comments with a short description restating the task in your own words.

Step 3: Test
(1) Come up with a generic example of given input(s) and expected output(s)
(2) Test this example against the stated objective
(3) Come up with 2-3 edge cases and test those as well.
(4) Refine your comments in Step 2 within any discoveries from above.

Step 4: Data
(1) Make a list of the appropriate methods and operators for manipulating the data types specific to this problem

Step 5: Logic
(1) Write the logic
(2) Order the logic
(3) Fill in the code
(4) Make sure to manage your data as it's being manipulated
(5) Proof your code till it works

Step 6: Edge Cases
(1) Test for edge cases and modify your code as needed.

The worst thing you can do in a coding interview is to be quiet. So always articulate what you're thinking as you think about it. It's better to say something, try it, and have it not work, then it is to spend a lot of time silently looking at something and then get it right. Don't let the mood get awkward. If you don't know where to go next, ask a question.

If you want some feedback, try recording yourself (screen capture with audio) while working on one of the Coding Bat problems above. Then you can play it back and determine where it felt awkward and where you sound confident. Practice doing that until you feel confident regardless of the problem being given.

24 Hour App Build

Some companies will ask you to build something. It might be something as simple as a List View application, it might be more complex. Usually they give you a time limit like a day or a weekend. Don't worry if what they ask you to build is beyond your knowledge or abilities. Just tell them that you will do your best and then get to work. Start by reaching out to the other people taking this course and asking for some insights, see if anyone has built this before. Then while you're waiting to hear back, start scoping out the project. The most important thing is to demonstrate that you could eventually build what they asked for even if you can't in the time they give you.

Example Build Request:

Could you build a simple Android app that does the following:

1. Talks to a RESTful API
2. Parse JSON from a RESTful API
3. Build a scroll view with images
4. Remembers where you were scrolled to when screen rotates

Part 4: Prep Materials

Use this course, your fellow participants, and your local Android developer community as resources. You may also want to check out the following:

www.youtube.com/user/androiddevelopers (Youtube channel by the pros at Google)

http://forum.xda-developers.com/coding (Global forum for Android development)