When I'm interviewing iOS developers, here is what I'm looking for and what I'm not looking for.

What I look for

* Solid development experience - by this I don't mean "I developed, by myself, thirty one games and pushed them to the app store". I mean you've been through a number of development cycles and encountered (and solved) problems enough times not to get thrown when something weird happens.

* Ability to work solo if need be - Pair programming is the new hotness in some areas but sometimes you have to go it alone. You need to be self-sufficient enough to be given a problem and left alone to solve it.

* Ability to work as part of a team - Along with the previous bullet point, you need to know when you've beaten your head on the desk enough to go ask for help or look on Stack Overflow or Cocoa Controls

* Being able to DIY or adapt - It's one thing to have some manager tell you you can build your app quickly by slapping together a bunch of components (React Native enthusiasts think this way); it's another to do that, have one of them not play nicely with the others and you have to go dig into five or six components you didn't write to fix the problem. You need to be able to take a look at someone else's code (*cough*AFNetworking*cough*) and recreate just as much of it as you need. You avoid the consequences of the old saying "Marry in haste, repent at leisure."

* Hungry - Mobile is fast-moving. You have to be hungry to stay enough ahead of things to be able to bat aside the flavor of the month "write once run anywhere" hybrid app solution until you understand it and can give a reasoned opinion.

* Comfortable Talking about Your Projects and Your Role In Them - Resumes, especially ones that come from recruiters, tend to have weasel-words in them that are red flags - "Involved with" or "Helped" or "Worked with" are an attempt to inflate minimal experience into something that *might* pass an interview if the interviewer isn't technical or experienced. I ask people about what they worked on, what they did, what they encountered and how they solved problems. Sometimes I ask why they made a certain decision - not that I think they're wrong, but how they thought it through.

What I don't look for

* Dead-Enders - It's one thing to be dedicated to solving a problem and shipping software. It's another to think you have to grind yourself into dust to do it. Nobody really needs to make a habit of 80 hour weeks for months at a time.

* Trivia Buffs - We need to write code, not argue over what CFString did eight years ago. I care that you know how to build an app.

* One-Uppers, People with Something to Prove - People like this are never fun to work with and aren't very often worth the trouble they bring.

* <u>BS Artists</u> - There are red flags I mentioned earlier. Others are things like trying to take over the interview and direct it their way, evasive language, the “Gish Gallop” of acronyms and esoteric concepts designed to overwhelm the interviewer(s) and seem a lot smarter than they are. There are lots of other ones but this is a short post.

* <u>Perfection</u> - Nobody is perfect. Everyone makes mistakes.

Technical Proficiency

Where I work we give a four-hour timed task. You build an app from scratch to do something useful. You’re not recreating Paper or Facebook, but you’re not writing “Hello World” or “Flappy Swift” either. That, along with a standard technical interview designed to see where your experience is heavy and where it’s not, gives me a good picture of your skills and abilities.