Mobile Straight pipe Script

Hi, I'm __ and with my colleagues on the phone we're researching the development of mobile apps in our ecosystem, in particular how users like you debug/troubleshoot issues involving data being sent between a machine and the internet.

Ultimately, we want to use calls like these to build up a general model for how *you* deliver mobile experiences that work properly, so when we make new designs or workflows we can ask, "Does this work for [walgreens Person?] What about his/her stakeholders and technology environment?" in other words, we want a kind of [walgreens/home depot/whatever] 'simulator' in our heads that we can use to stress test the software we're working on.

So lets start with some basic background questions. Can you give us a quick 5 min version of your educational background & how did you got to this point in your career?

Another opening question we like to ask is - what is your formal job description/responsibilities and is there a better way to describe what you actually do? For example, people often think designers are just pushing pixels around all day every day, when in fact they're really in a lot of meetings, learning about users, talking to engineers and Product managers, collaborating, and then once all of the human groundwork has been laid, they can get to designing workflows once everyone's on board.

Can you list some projects you've worked on in the past 6 months? Which of those were "typical"? which of those were not? Why did you characterize them that way?

- Is there a formal process for developing these mobile projects at your company? If so, what is it/how would you describe it?

Which of those (if any) were "smooth" vs "difficult" projects? Why?

- want to understand if the *kinds* of problems they are debugging are primary (i.e. due to code or things in their control) or secondary (i.e. due to the design/implementaion of systems outside of their control)

What tools/processes are you using to test/troubleshoot/debug?

- What were the reasons behind using Tool X?
- if Tool X was home-grown, why was that more valuable than something off the shelf?
- do you use a Device library? Why/why not?
- Did you use these tools on all of the projects you mentioned to us?

Lets drill into one of the more typical projects you described earlier

- I know this is kind of a cliche way to ask this question, but I still like using this format. On a scale from 1-10 10 being the best possible, how easy was it to debug errors/fix code/solve problems on [one of the projects they menioned]? Why did you rate it this way?
- Were there any initial get-off-the-ground issues/problems? If so, what were they?
- What about *polish/optimization stage* - How did you know when were these projects "finished"/"ready to deploy?
  - (trying to assess time driven vs. quality driven decision making)

Are there any examples of projects that were released that should have had more time in development?

[if at all possible share screen/share workflows]

Debriefing Survey

Individual skills

On a scale from 1-7, what is this user's TECHNICAL expertise (1 = none at all, 7 is extremely high)

On a scale from 1-7, what is this user's MARKETING expertise (1 = none at all, 7 is extremely high)

1-7 agree/disagree - This user has to fight many problems that are in their PRIMARY control (primary means they have the appropriate permissions/knowledge/whatever to solve those problems)

Team/Stakeholder Environment

1-7 agree/disagree - This user has to fight many problems that are in their SECONDARY control (secondary refers to problems that may come from complexities in the system, issues they cannot fix themselves, etc.)

1-7 agree/disagree - This user is empowered by their workplace to make any/all changes necessary to their app

1-7 agree/disagree - The technology environment the user is in can accommodate all of the changes the user needs

Motivation here? Team culture/KPIs/rewards/whatever geared around delivering a bug-free product?

Best practices/structured process? Team concern or an individual?

Notable Quotes:

Priorities

GENERAL INTERFACE

Customer interface to provide for core capabilities that help with usability
Provisioning - Adobe ID
Inspecting
Scripting
Saving
Sharing & Exporting

OBSERVATION

Provide real-time, pre- and post-production device and app observability capabilities.

Device – provide device diagnostic detail on model, storage, enabled/disabled device capabilities
Network – View and inspect HTTP(S) traffic requests and responses along with associated body and headers.
App – View and inspect app logs, Adobe and 3rd party vendor mobile SDK events
Database – Browse data, track queries in real-time, show DB schemas, and query app SQLite databases.
Usage charts on CPU, memory, storage, battery, WIFI/cell-strength, device diagnostics, and device logs

VALIDATION

Inspect, review, and validate data collected and transmitted by Experience Cloud solutions and Platform services. Validate and test app messaging and optimization campaigns for Experience Cloud and Platform services functionality.
Adobe Analytics – Review app configuration for Adobe Analytics. View and inspect data being posted to Analytics servers, show processing rules associated with hits being sent.
Adobe Audience Manager – Review app configuration for Adobe Audience Manager. View and inspect data posted to and syncs requested of Audience Manager servers.
Adobe Target – Review app configuration for Adobe Target. View and inspect data posted to and responses received from Target servers.
Adobe Campaign – Review app configuration for Adobe Campaign. View and inspect tracking data and device tokens sent to Campaign. Validate notification and in-app message campaigns for quality, accuracy, and consistency.

SIMULATION

Test mobile apps for real-world conditions by simulating user behavior along with device & environment conditions.
Memory – controls for users to simulate changes in memory
Location – macro and micro location
Bandwidth – throttling, distant/rural network conditions
Battery – battery v performance
User profiles
Other unusual events – weather, traffic, etc.

charles and debugger apps have the pieces...

What pieces are valuable

not good use cases