App performance is part of the User Experience

Balazs Fonagy
Supercharge
UX Lead

Balazs Kovacs
Supercharge
Solution Architect

Supercharge

“No UX Designer is an island.”

John Donne, 1624

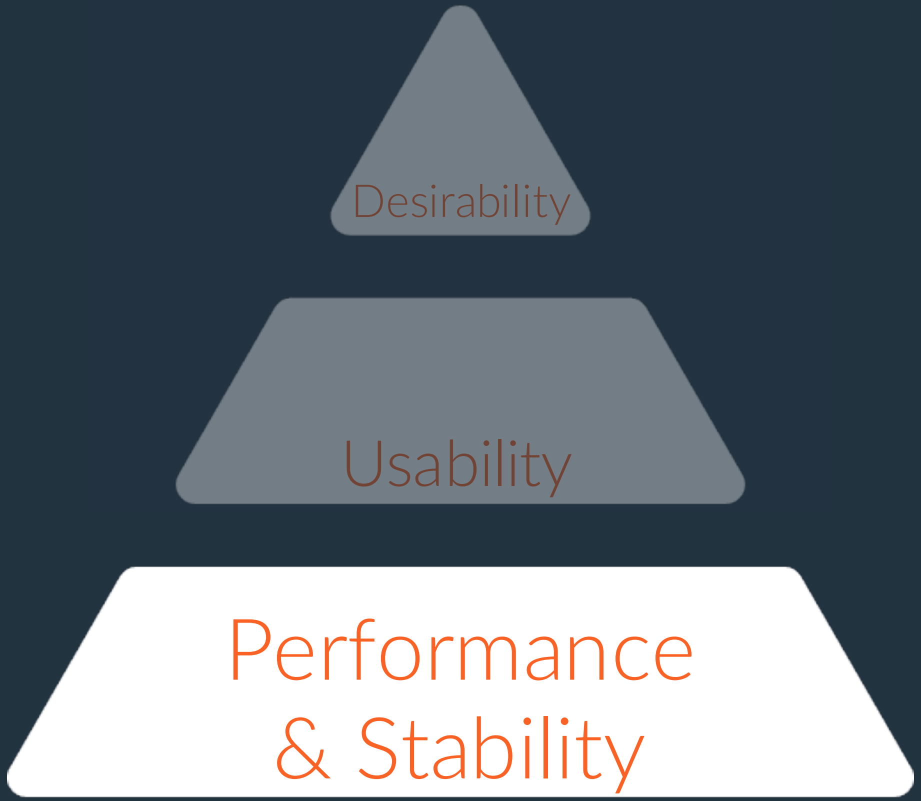
Your design

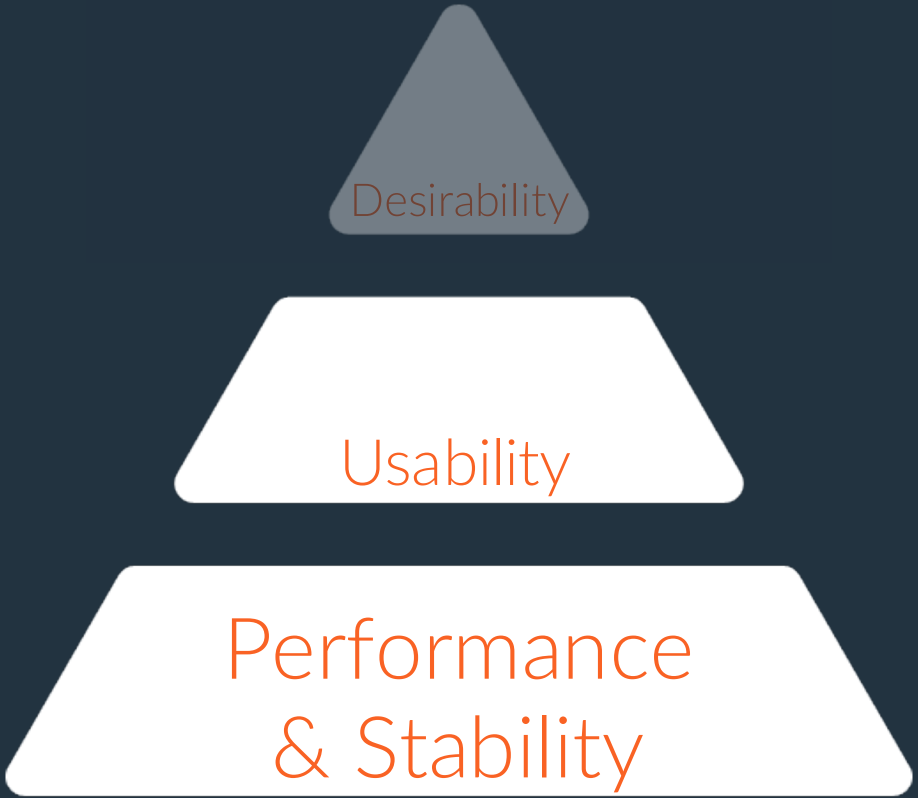
The final software

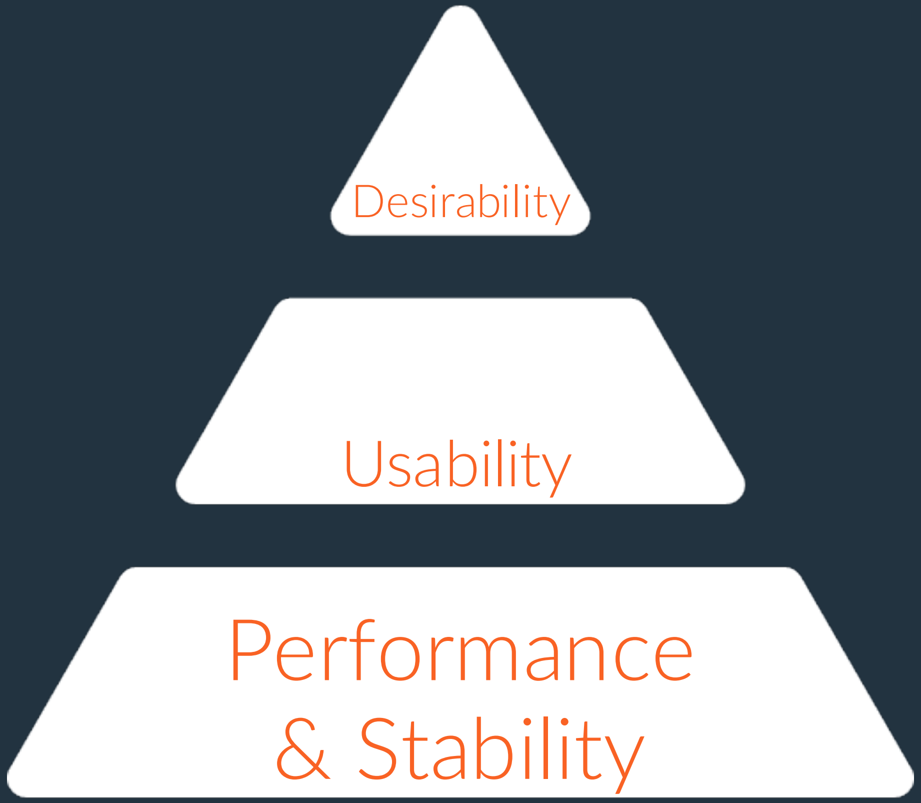
Lack of competent, mobile focused software engineers…

…during the design

…or the development


Desirability
Usability
Performance & Stability


Desirability
Usability
Performance & Stability


Desirability
Usability
Performance & Stability

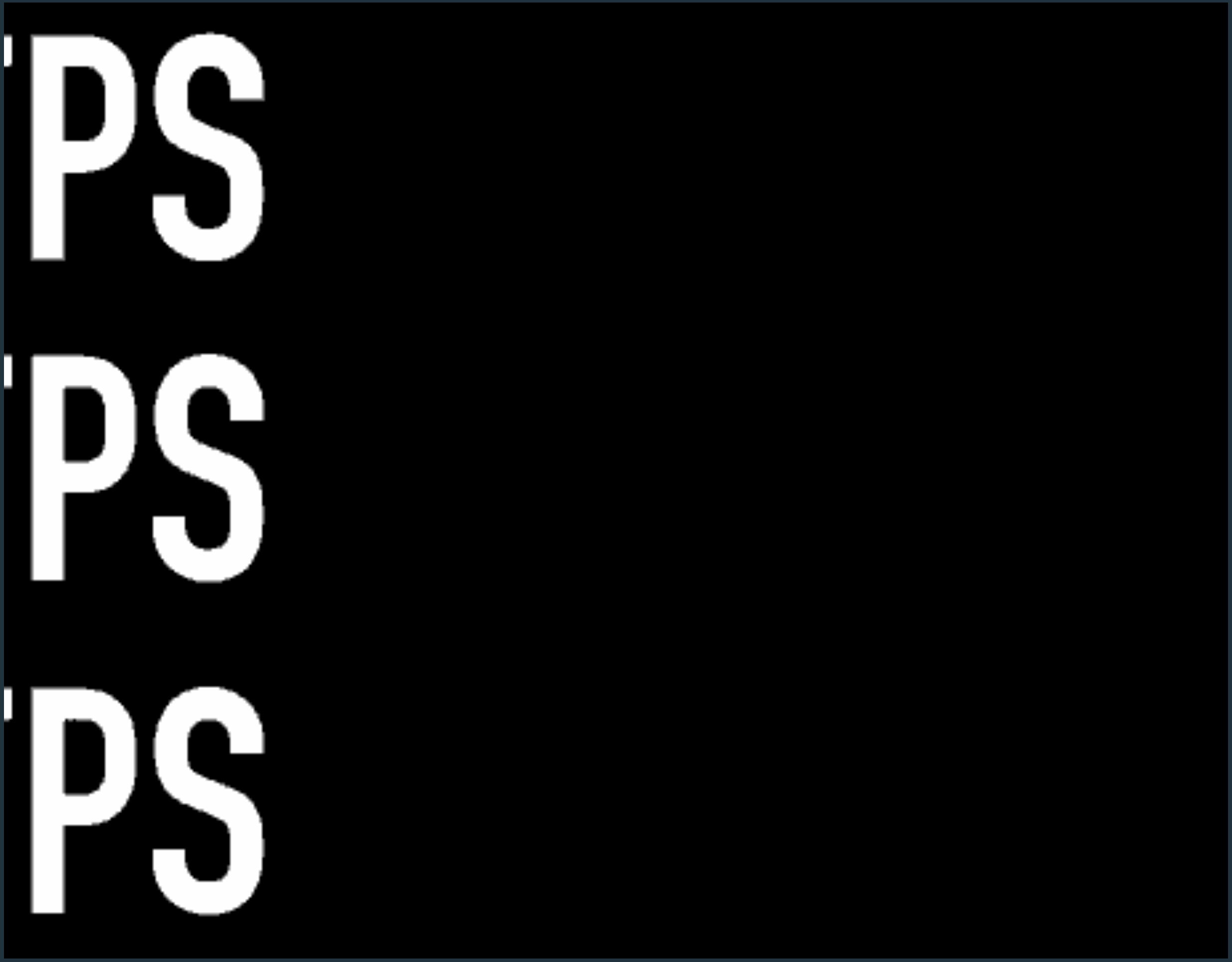
You are responsible for UX. Performance is the foundation of UX on mobile.

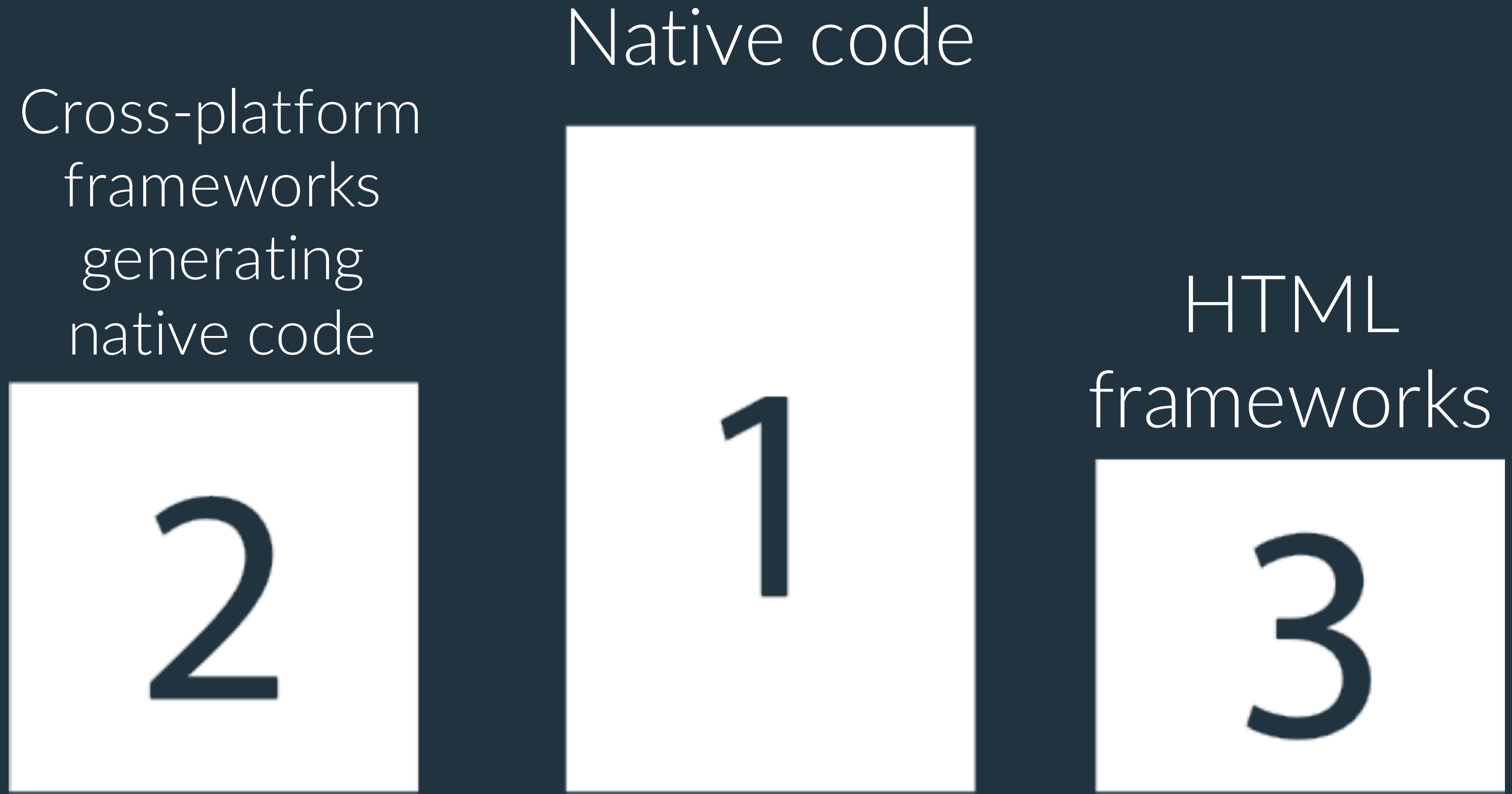
Are you responsible for the performance?

User expects mobile apps to be software, not websites. The benchmark is different.

Native code vs. HTML/Cross-platform solutions

UI/animation performance and the quest for the magical 60 fps


PS
PS
PS

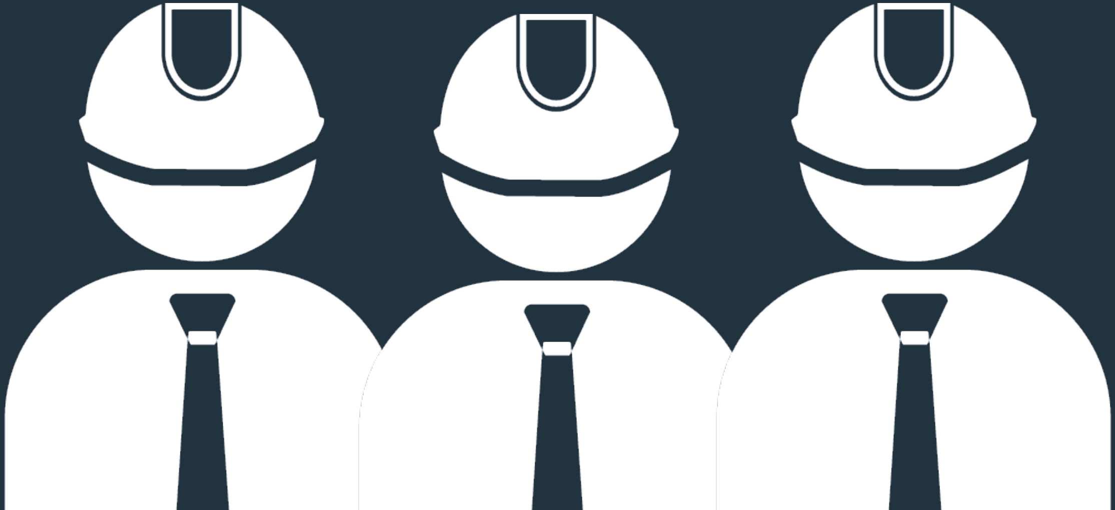

Native code
1
Cross-platform frameworks generating native code
2
HTML frameworks
3

Native code: expensive, best UI performance. No layer above the SDK

Cross-platform frameworks generating native code: can get expensive, decent UI performance, one competence is enough, lots of constraints

HTML frameworks: cheap, but at the end it's a web app, running in a web view. Browsers are not built for performance but for compatibility

Decided to go native? Now you only need someone who doesn’t mess up the UI…



Load……………………………………………

……………………………………………

……………………………………………

……………………………………………

……………………………………………

……………………………………………

……………………………………………ing

Step 0: if the backend is bad you are f… out of luck.

1. Decent HW
2. Good Location
3. Decent SW

A generic API makes the developer proud, a specific API makes the mobile product fast

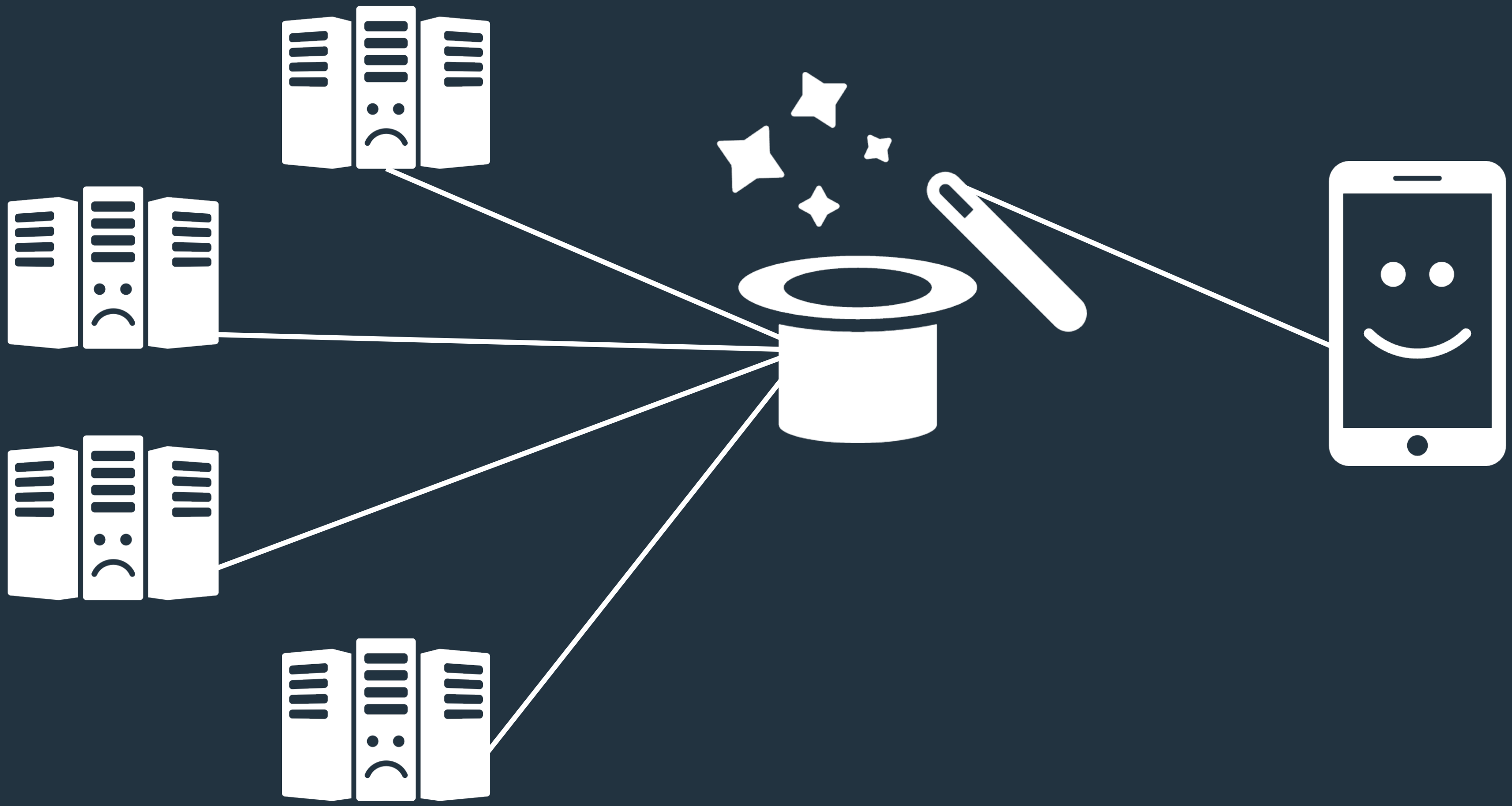
Mobile specific API

1.Only data that you need

2. Data grouped by screens

3. Non-generic API

If the backend is bad create a mobile specific middleware!



Level 1: gather, streamline data and provide a mobile specific API

Level 2: Pre-cache data and do the level 1 stuff

You have to download the data at some point…

1. Download at app install:

✔ For large, almost never changing data

2. Preload at the launch

✔ Middle sized databases that aren't often updated

✔ Can be done in the background

3. Download when the user needs it

✔ For real-time or often changing data

✔ Data that's usage couldn't be predicted

The million dollar question: load before the screen or on the screen?

The easy way: wait and provide a perfect screen

The hard way: build it in front of the user

On screen loading done right

- Prioritize content. Download the small, important elements first
- Cache&reuse as much as possible
- Lazy load & above the fold load
- Use placeholder elements while loading

Progress bars make us more patient

Don’t pull it. Push it!

For the best User Experience:

1. Go native
2. Create a mobile specific API
3. Think hard about loading

BTW: We are hiring ;)

hello@supercharge.io

Balazs Fonagy
Supercharge
UX Lead

Balazs Kovacs
Supercharge
Solution Architect

www.supercharge.io

Supercharge