[redacted]

From: Jesper van den Ende [redacted]
Sent: 05 February 2022 16:22
To: Mobile Ecosystems
Subject: Feedback on the mobile ecosystems market study interim report

Categories: [SharePoint] This message was saved in 'Mobile Ecosystems > Documents > Parties > Individuals and Public > Jasper Van den Ende'

Hello,

I'm Jesper van den Ende. I represent Pelican Party Studios and Jesper the End B.V.
We create video games, the majority of which are tailored for the web.

I'd like to provide some of my views as requested in the mobile ecosystems market study interim report.
While I can't comment on all points from box 10.1, I'll try my best to provide you with information where I can.

> 1 Our understanding of the markets within the scope of the study.

The games we have been creating until now have covered a wide variety of genres and platforms.
Some of which are VR games, others are tailored for mobile, but the last few titles have mainly been developed with the web in mind.
The monetization strategy has been quite different for these games as well.
Some of the games are available for purchase by the consumer, others are free and are monetized using ads, and a few are free and don't have any monetization strategy at all.
Since web pages are cross platform generally speaking, all of our web games are available on both desktop and mobile,

> 2 Our initial findings on the competition concerns under each theme:

I think chapter 5 already covers a lot of info, but I'd like to add a few extra points.

First I'd like to point out the fact that WebKit updates, and therefore the engine of all browsers on iOS are bound to iOS updates. While this has already been pointed out several times, I'd like to add that because of this, many users with older devices that do not support newer versions of iOS are now stuck with an older version of WebKit. This further builds upon the argument that WebKit is lacking features. Not only does it lack features, even if WebKit does eventually end up supporting these features, it will generally take roughly two years before these features can actually be used by developers. Compare this to Android, where browser updates are available even for very old devices.
As an example, let's take a look at the store page for Firefox on the Google Play store. It reports being available for Android 5.0 and up. Nowadays roughly 1.2% of the Android users are using a version lower than this. (https://www.appbrain.com/stats/top-android-sdk-versions)
Compare this to statistics from iOS and you'll see a very large portion of its users are not using the latest version available. (https://gs.statcounter.com/ios-version-market-share/) Most of these are caused by users simply choosing not to update. But especially in case of the much older versions, this is caused by older devices not being able to update to newer versions of iOS.
Admittedly, the latest browser version being available to Android users does not mean they are actually using it, but with the iOS statics you can be *certain* users with an older version of iOS also have an older version of WebKit. Whereas with Android this only *might* be the case. And even if a user on android has an older browser version, at least they have the option to upgrade. iOS users are forced to upgrade their OS, or in some cases even to buy a new device.

Regarding security, specifically Apple claiming the following in Appendix F:

> "By mandating the WebKit restriction, Apple can rapidly fix any vulnerability for all apps across the iOS ecosystem"

One example of this not being the case is the recent IndexedDB vulnerability (https://safarileaks.com/). This vulnerability allowed a website to get information it was not supposed to from other sites, if those sites had an open tab. According to the article (https://fingerprintjs.com/blog/indexeddb-api-browser-vulnerability-safari-15/), a bug report was submitted on November 28, 2021. And a fix was available on January 26th 2022. During this time users had no other option than to wait. Had other engines been allowed on iOS, users would have been able to switch their browser during this time.
Note that, while I do think 2 months is too long for a vulnerability with this severity, I'm not trying to argue that Apple never rapidly fixes vulnerabilities. Some bugs just happen to take longer to fix than others. My main point is, that in a situation like this it would have been better for users security wise, if other browser engines had been available.

Another point can be made regarding security. Apple argues security is important, and therefore only WebKit is allowed. But yet at the same time they do allow millions of apps to be downloaded from the app store. These apps do not run in a sandbox like websites do, and therefore are a lot less secure. Apple argues native apps are allowed to run without this sandbox because of the review process, but this argument doesn't hold because the review process isn't water tight. One example is crypto currency scams that have found their way past app review: https://www.washingtonpost.com/technology/2021/03/30/trezor-scam-bitcoin-1-million/
The amount of concern Apple has regarding security for other browser engines doesn't make sense when you compare it to the amount of concern regarding native apps that are allowed in the app store. If Apple really cared about security as much as they say they do in regard to third party browsers, they would have to only allow apps that run in a similar kind of sandbox in their store, so that they can stop screening for malicious apps using the app review system they use today.
Since this is not the case, I argue Apple is only using the security argument to maintain their monopoly on browser engines on iOS. If they are legitimately concerned about security, they would have to apply this level of concern to native apps as well.

Regarding compatibility issues, specifically figure 5.4 of the publication. I agree with the remark from Apple in Appendix F:

> "Apple submitted that a focus on compatibility or web standards compliance does not adequately reflect attributes of browser engine quality, including quality, performance, stability and privacy, and Apple explained that, in the normal course of business, it tests web app responsiveness, JavaScript performance and graphics performance."

And unfortunately I can't give any hard statistics on compatibility and general availability of features either. But from my day to day experience with developing for the web I can clearly tell that WebKit has had the most amount of issues in regards to compatibility. Sometimes this is an obscure issue that I manage to fix after finding a workaround from other users online. But other times it is an issue that can simply not be fixed, and we have to live with the consequences.
The Fullscreen API not being available for non-video elements for instance. Or not being able to play audio while a website is in the background (https://bugs.webkit.org/show_bug.cgi?id=198277).
Others are features where it has deliberately been decided that these will not be implemented. Such as the Web Bluetooth API. In those cases, creating an application with certain functionality is genuinely impossible without creating a native app and publishing it to the app store. I agree that security implications for APIs like these are definitely a fair point. But Apple is using the web as an excuse to be able to disallow certain apps from the App Store.
E.g. from review guideline 4.9: "Of course, there is always the open Internet and web browser apps to reach all users outside of the App Store." This seems unfair to me because using the web is not always possible. This issue can be circumvented by allowing multiple browser engines in my opinion. That way WebKit doesn't have to implement APIs like these and remain privacy focused. And users can opt for another browser engine if they want to that does support these features.

I'd also like to add to the following:

> "the primary concern that Apple raised with us was about the very large number of apps that use a browser engine to render web pages. In particular, Apple told us that if these apps were allowed to use a non-WebKit browser engine, Apple would have to require each of those developers to update their own app, and that this would cause some vulnerabilities to persist for months, if not years."

The publication already mentions a few great points, but I'd like to add some more.

- Apple acts as if every app using an in app browser is going to bundle its own browser engine, causing thousands of apps to include an engine that would need to be updated in case of a vulnerability. But this doesn't necessarily have to be the case. On Android for instance, only a few apps include an engine, Firefox, Chrome etc, and any app that needs to use an in app browser is able to access these engines from those installed browser apps. That way if any of the browsers are updated, they are automatically updated across all apps that use them as well.
- Apple also acts as if it is solely their responsibility to keep users secure. But I think that if a user chooses to install a browser from a third party, the responsibility can be expected to shift as well. I think it makes sense that an app store has a high responsibility to not allow any apps that are malicious. But we're talking about complex applications here. At that point, if a browser contains a vulnerability, it no longer makes sense to complain to Apple that they're not doing anything about it, and instead the developer of the browser engine should be held accountable.
- Apple mentions that determining what counts as a browser app is an issue in regard to the review process. But it seems they are already making this distinction, as can be seen in an interview with Craig Federighi: https://youtu.be/Q2aaCDNjWEg?t=241 (4:01). And even still, the review guidelines contain a few general guidelines as well, that would be able to guard against this. For instance "5.6.4 App Quality" mentions that apps should maintain a high quality.
- And as a last resort, I don't think it is unreasonable for Apple to disallow browser engines when they are not maintained frequently. Or even going so far to include a kill switch for apps that bundle their own browser. That way if an app contains a vulnerability and the developers don't fix it, Apple can remotely disable the app.

> 3 The merits and challenges of the range of potential interventions that we have identified in this interim report.

Regarding the effectiveness of the interventions. I think enforcing multiple browsers on iOS has a high potential of being effective competition wise. As it stands right now Apple has a low incentive to improve WebKit, and it shows. Though I do worry that if multiple browser engines are allowed on iOS, Chromium might quickly take over WebKit in usage, similarly to what it has done on desktop. This would essentially create a monopoly for Chromium. But my hope is that if multiple browsers exist on iOS, that Apple would respond to the much higher incentive to innovate. I'm confident the latter is more likely to happen. Looking at the current statistics, the vast majority of iOS users are using Safari. So I think the chance is low that Chrome will be able to outcompete Safari.

Lastly I'd like to point out that the following point in the interim report is no longer true:
"iOS mutes web apps by default: and touch input from users is required for audio to work"
This is still the behaviour in WebKit, but is the default behaviour in Chrome as well, so this is not an incompatibility. The issue I mentioned earlier about not being able to play audio while apps are in the background still exists though.

I hope this information helps with further investigation of the study.
Feel free to contact me if you have any questions about the views I have provided.

Further reading:
https://infrequently.org/2021/04/progress-delayed/
A comprehensive list of missing features in WebKit.

https://httptoolkit.tech/blog/safari-is-killing-the-web/
Contains a less comprehensive list of missing features as well as a few show-stopping bugs.
This article also mentions that allowing multiple browser engines on iOS might actually be a bad thing:
"The one saving grace today is that Apple blocks use of any non-WebKit engine on iOS, which protects that one environment, and the iOS market (in the US at least) is large enough that this means Safari must be prioritized."

https://www.intego.com/mac-security-blog/apples-poor-patching-policies-potentially-make-users-security-and-privacy-precarious/
An article questioning the security of iOS versions older than the most recent one.

Best regards,
Jesper van den Ende