Mobile Browsers and Cloud Gaming

A response to the CMA's Consultation on Proposed Market Investigation Reference

VERSION 1.1

Open Web Advocacy
contactus@open-web-advocacy.org

1. Table of Contents

2. Introduction

Open Web Advocacy would like to thank the Mobile Ecosystems Market Study team for their incredibly in-depth study and analysis and for bringing attention to the critical issue of Browser and Web App competition to the regulatory world. It is because of this study that there is now a strong focus on both the potential of Web Apps and Browsers as a long term solution to the competition issues within mobile ecosystems. We wholeheartedly agree with the proposal to end the WebKit restriction on iOS. We believe this is an important topic and that only regulation can restore competition to the mobile app ecosystem.

OWA believes that the Market Investigation Reference into the supply of mobile browsers and cloud gaming is not only warranted but highly urgent and **essential**. For the past decade, severe underfunding of Apple's browser Safari combined with a ban of competitive browsers on iOS has removed competitive pressure and resulted in an unstable platform missing critical functionality ensuring that only vendor-specific native apps are competitive. Intervention is essential not only to the future of competition between Browsers but also to deliver a free and open, universal, interoperable application development and distribution platform.

App stores are the primary way that users install apps on mobile devices, and for iOS it is the only way for users to install capable apps. Mobile operating systems and hardware gatekeepers exert extreme control over users' devices long after they are sold. App stores and device restrictions allow gatekeepers to extract extensive revenue from users not by merit or user choice but by blocking the user from installing software from third party providers, including those that compete with their own offerings. They can further leverage this power to charge the user additional fees (up to 42.8%) on software and services from third party providers.

The software market is highly competitive, it is not possible to add 42.8% fees and not have the majority of costs be passed to consumers. OWA views these fees as an unreasonable tax on the user.

> *If a developer for a given number of users requires $7 per a user to support and maintain their product and the gatekeeper demands a 30% cut, in order to receive $7 the developer must increase their price to $10, thus increasing the price the consumer pays by 42.8% (1/(1-.3) = 1.428).*

Various regulatory agencies around the world have proposed forcing the mobile operating system gatekeepers to allow third party app stores. While this would go some way to alleviating these issues, gatekeepers via their app stores would still wield considerable power, as users are unlikely to install many app stores, particularly if the procedure for doing so is daunting.

Additionally, depending on implementation, Native Apps would still have to be written in different programming languages and with different APIs which are bespoke to each operating system. This means that Native Apps would not be interoperable between systems, requiring significant cost to build and maintain multiple versions of the same App while providing the gatekeeper a form of lock-in via non-transferrable investment and expertise.

Web Apps are an untapped source of interoperable competition with the app stores. When allowed to compete fairly, Web Apps *can* provide a compelling substitute to Native Apps. It is on this question of capability that we are most interested. On every OS but iOS, browsers have access to important APIs and features, allowing them to deliver portable applications across platforms. It is only through a set of interlocking restrictions that Apple has prevented this competition from emerging on iOS, and we believe that these restrictions are anti-competitive and unwarranted.

In a world of capable browsers, Gatekeepers would not be able to block the installation of safe, sandboxed and secure Web Apps and would have no leverage to add additional fees or block Apps that compete with their own offerings. Gatekeepers can't force third party providers to pay the tax by threatening to block users' ability to download those Web Apps, as they do with Native Apps.

Web Apps are applications installed from the browser that can provide users an experience on par with Native Apps when modern browsers are permitted. In competent, modern browsers they can work offline and provide a superior security and privacy model. They are, in short, capable of amazing things.

Investment into the Web platform is highly desirable for future competition as it is both open and free and allows any company to build upon it as a platform. Web Apps are increasingly becoming the primary choice for companies on desktop operating systems. Commercial users now spend greater than 60% of their time within a browser and many companies such as Adobe and Microsoft are heavily invested in adopting interoperable Web Applications (Word, Powerpoint, Photoshop, Illustrator) because they allow these businesses to produce higher quality, maintainable applications which can target multiple platforms with near-zero friction to users and low costs of operation for developers.

Apple's original vision for applications on iOS was Web Apps, and today they still claim Web Apps are a viable alternative to the App Store. Apple CEO Tim Cook made a similar claim in Congressional testimony when he suggested the Web offers a viable alternative distribution channel to the iOS App Store. They also claimed this during a court case in Australia with Epic.

Despite this **Web Apps are not currently a viable competitor** on iOS for three reasons:

1. **Browser Ban**
   Apple's Safari is the only browser on iOS as all other browsers have been effectively banned.

2. **Missing Features**
   Apple has refused to implement key features (some for more than 10 years) that would allow Web Apps to compete with Native Apps from the App Store, both on iOS and in Safari.

3. **Bugs**
   Apple's iOS Safari is buggy and unstable compared with rivals, rendering it unviable as a platform for applications.

iOS is a significant player in the mobile ecosystem landscape with 51% in the UK, 46% in the US and nearly 65% of web traffic in Japan. iOS users enjoy high average incomes, creating a market that can not be ignored. This goes a long way to nullifying Web Apps development cost and interoperability advantages; not only for iOS but also for Android. Companies that cannot imagine using the web to address the wealthiest, most valuable users are forced to accept the costly logic of re-developing their services for each native ecosystem.

We have written about this extensively in our paper "*Bringing Competition to Walled Gardens*".

Regulators worldwide (including the CMA) have acknowledged this anti-competitive behaviour and have proposed legislative remedies to counter it.

The latest revision of the EU's Digital Markets Act notes (emphasis added):

"When gatekeepers operate and impose web browser engines, they are in a position to ***determine the functionality and standards that will apply*** *not only to their own web browsers,* ***but also to competing web browsers*** *and, in turn, to* ***web software applications."***

The UKs Competition and Market Authority have highlighted this accurately and extensively in their mobile ecosystems market study interim report and write (emphasis added):

> *"As a result of the WebKit restriction,* ***there is no competition in browser engines on iOS*** *and* ***Apple effectively dictates the features that browsers on iOS can offer*** *(to the extent that they are governed by the browser engine as opposed to by the UI)"*
>
> *"Importantly, due to the WebKit restriction,* ***Apple makes decisions on whether to support features not only for its own browser, but for all browsers on iOS****. This not only restricts competition (as it materially limits the potential for rival browsers to differentiate themselves from Safari on factors such as speed and functionality) but also* ***limits the capability of all browsers on iOS devices, depriving iOS users of useful innovations*** *they might otherwise benefit from."*

They note that Apple has multiple incentives to hold back WebKit, hindering the ability of Web Apps to compete with the iOS App store (emphasis added):

> *"First, Apple receives significant revenue from Google by setting Google Search as the default search engine on Safari, and therefore benefits financially from high usage of Safari. Safari has a strong advantage on iOS over other browsers because it is pre-installed and set as the default browser. The WebKit restriction may* ***help to entrench this position by limiting the scope for other browsers on iOS to differentiate themselves from Safari*** *(for example being less able to accelerate the speed of page loading and not being able to display videos in formats not supported by WebKit). As a result, it is less likely that users will choose other browsers over Safari, which in turn secures Apple's revenues from Google."*
>
> *Second, and as discussed in Competition in the distribution of native apps, Apple generates revenue through its App Store, both by charging developers for access to the App Store and by taking a commission for payments made via Apple IAP.* ***Apple therefore benefits from higher usage of native apps on iOS. By requiring all browsers on iOS to use the WebKit browser engine, Apple is able to exert control over the maximum functionality of all browsers on iOS and, as a consequence, hold up the development and use of web apps.*** *This* ***limits the competitive constraint that web apps pose on native apps, which in turn protects and benefits Apple's App Store revenues****."*

They propose banning Apple from blocking third party competitors from porting their own browsers to iOS, complete with competing engines.

The UK CMA found Apple's security arguments justifying its ban on competitors unconvincing (emphasis added):

> *"Apple's restrictions on competing browser engines: Apple does not permit the use of third-party browser engines within its mobile ecosystem – all browsers are required to use its browser engine, WebKit.* ***We have not identified compelling evidence to date that suggests that, for dedicated browser apps, the potential impacts on competition or consumers from Apple's WebKit restriction are justified on security grounds.*** *We are therefore seeking to assess the merits of a requirement for Apple to allow alternative browser engines on iOS, at least for dedicated browser apps. This could be implemented by requiring Apple to permit third-party browser engines to interoperate with its iOS operating system,* ***subject to those browser engines meeting conditions that would address any risks that might arise from a greater choice of browser engines (for example, complying with appropriate quality and security standards)."***

Apple collected $72.3 billion USD in App Store fees in 2020. While it has not published the costs of App Store review, payment processing, refund handling etc, it has been estimated that the iOS App Store has a nearly 80% profit margin. Industries with healthy competition feature leading firms with profit margins between 5 and 20 percent. This imbalance strongly implies that Apple's removal of functional competition in the App Store and beyond have distorted the mobile phone market for software and services for 51% of UK's consumers, and a substantially higher fraction of mobile commerce.

The most valuable part of preventing competitive Web Apps and Browsers however is not the App Store, it is ecosystem lock-in. By having apps that only work on iOS or work better on iOS Apple locks the customer into upgrading to another Apple device. User's having the ability to easily migrate to other devices along with their apps would harm Apple's business model.

OWA believes that gatekeepers of mobile hardware/operating systems should compete to offer additional services and software to customers on merit and user choice, not by blocking applications that compete with their own or by charging consumers additional fees on applications/services from competitors. OWA proposes a number of remedies (or desired outcomes of remedies) that we believe are vital to restore competition and user choice to the mobile ecosystem.

Only if regulatory or legislative action is taken can the bright future of software competition be saved, not just for the UK but for the world. Apple and Google were each started in garages by two developers, we need to ensure this dream holds true for the next generation of entrepreneurs. Competition is the key to unlocking maximum benefit for consumers.

3. Direct Answers

1) Do you consider that our analysis is correct with respect to the suspected features of concern in the supply of mobile browsers and cloud gaming in the UK?
Yes. Specifically we agree that:

- The inclusion of mobile browsers and the analysis that there are significant barriers to competition in the supply of mobile browsers
- Apple and Google have high shares in mobile browsers with very strong positions
- Apple's requirement that all browsers must use WebKit (or more specifically an uneditable WebView that Apple exclusively controls) significantly limits browser engine competition and limits the extent of differentiation.
- The barriers to competition include
  - In-App Browsers
  - Pre-Installation and Defaults
  - Restrictions on access to functionality
- Apple and Google are highly likely to retain this market power in browser and browser engines for the foreseeable future and that both have an adverse effect on competition.

  We would like to emphasise that from the viewpoint of Web Developers given the comparable freedom of browsers on Android, Windows and Mac, that Apple's anti-competitive actions on iOS are significantly worse since only on iOS is meaningful browser competition completely prohibited. It is our hope however that the CMA has the capacity to address issues across both ecosystems.

2) Do you consider that our analysis is correct with respect to the reference test being met in relation to the supply of mobile browsers and cloud gaming in the UK?
Yes - In OWA's opinion the reference test for both mobile browsers and cloud gaming has been easily met.

The problems are both of a significant scale and have a significant adverse effect on competition. To our knowledge it is unlikely that there would be other more suitable pathways to fixing these issues.

3) Do you agree with our proposal to exercise the CMA's discretion to make a reference in relation to the supply of mobile browsers and cloud gaming in the UK?
We emphatically agree and believe this is both an important and appropriate action.

4) Do you consider that the proposed scope of the reference, as set out in the draft terms of the reference published alongside this document, would be sufficient to enable any adverse effect on competition (or any resulting or likely detrimental effects on customers) caused by the features referred to above to be effectively and comprehensively remedied?
Yes.

5) Do you have any views on our current thinking on the types of remedies that a MIR could consider (see above and Chapter 8 of the market study final report)?

We have outlined in detail the remedies in this document that we believe are necessary to restore competition and provide a more even playing field for browsers and web apps.

8.116 This could be implemented by requiring Apple to permit dedicated browser apps which use third-party browser engines to be used on its iOS operating system, on condition that those browser engines meet appropriate security standards. This would be likely to result in a limited group of third-party browser engines being available via browser apps on iOS.
We agree that Apple should be required to permit dedicated browser apps with third-party browser engines, on condition that those browser engines meet appropriate security standards. We would also suggest that any standards of security would also need to apply to Apple's own browser.

8.118 (Vendor Recommendation:) Apple would also need to provide third-party browser engines and browsers with the necessary accesses and privileges on iOS, suitable public documentation and support channels.
We agree with this vendor recommendation.

8.118. appropriately vetted third-party browser engines and browsers should ideally be granted access to the same APIs as WebKit and Safari.
We agree with this vendor recommendation, but that this should be set as a minimum and that there should be a principle of open access to all software and hardware functionality subject to narrow scope security concerns.

8.124 As a result, it should be possible for Apple to provide support to a small number of approved third-party browser engines without significantly increasing security risks, at a scale of implementation costs which appears to be lower than suggested by Apple's submissions. These browser engines would be available to dedicated browser apps.
There are some complex subtleties here. Typically the "chrome/ui" layer of a browser needs to be tied to a specific version of the browser engine and Chrome/Edge/Firefox would need to ship a browser engine as part of the browser. Allowing other parties to use these engines without requiring them to receive a special trusted browser entitlement would likely require a system for publishing WebViews onto iOS (such a Chromium or Gecko WebView) which although plausible would not be trivial to design and implement.

8.127 We have considered whether Apple should be required to provide enhanced functionality on WebKit, to allow web developers more comparable access to the technology of the iPhone as native app developers. In the case of some functionality, this might be viewed as a necessary complement to removing Apple's WebKit restriction, but in others it could be viewed as a substitute.
Safari is and is likely to continue to be the dominant browser on iOS and can have a significant effect on the adoption of Web Apps. The statistics show that on MacOS where Safari has legitimate competition that it still has a 63.35% market share in the UK.

As outlined in the CMA's Mobile Ecosystems Final Report, Apple has a clear economic incentive to limit the functionality of Browsers and Web Apps. Although OWA believes that competition is the only long term fix, for Web Apps to be successful it is essential that Apple implement some critical functionality to ensure they will be considered a viable alternative by application developers.

8.128 We have also considered requirements relating to consistent API access to ensure that third-party browsers compete on a level playing field with Apple and Google in their respective ecosystems.
It is in the interests of competition that all browsers receive equal API access to all functionality. Browsers should be provided with all available software / hardware functionality.

8.129 Open Web Advocacy suggested to us that in principle, all the functionalities open to native apps could be made available to web apps, which would resolve issues relating to unfair competition between native apps and web apps.
OWA would advocate for the principle of open access to include functionality to native apps, Apple's own first party apps, system services & browser on a request basis from browser vendors.

8.131 Browser vendors have told us that Apple restricts third-party access to browser APIs and that there are features used by Safari which are not available to other mobile browsers on iOS devices.
Agree - Apple significantly restricts other browsers even from APIs that Safari has access (For example "Add to Home Screen")

8.131 Several stakeholders said that more equitable API access would be a necessary pre-condition for the effectiveness of more direct interventions promoting competition for browsers and web apps.
Agree

8.132 However, as a general principle, we would expect that if obligations to provide functionality were placed on Apple, and examples were identified where Google did not provide comparable functionality, then the same requirements could apply.
OWA has been primarily focused on the issues on iOS as we have identified this as the number one issue holding back the success of Web Apps. Although we realise some aspects may not be under the scope of the Market Investigation Reference, we fully support that any requirements that are placed on Apple on iOS are, where appropriate, placed on other gatekeepers across their mobile and desktop operating systems.

8.137 Following the removal of the WebKit restriction, competition may help to drive up standards on iOS, but there may continue to be some features that would continue to be restricted within iOS or Android, and in some cases there may be a good case for requiring parity with the treatment of native apps.
The principle of open access to software and hardware functionality needs to apply to trusted browsers so they have the ability to innovate and bring cross-platform functionality.

8.138 A principles-led approach could be implemented that would require the mobile ecosystems to give reasonable access by browsers to device functionality that is available to native apps, where that can be achieved at proportionate cost. This requirement, combined with competition on browser engines, should be able to assess questions about whether particular aspects of native app functionality should also be available to web app developers through dedicated browser apps.
The access should be extended to include the gatekeepers' system apps, system functionality and **available** hardware or software APIs. Where the functionality is already available via private APIs browsers should be allowed to use those private APIs (with analysis and reverse engineering if required) until a public API is made available.

The CMA should be aware that Apple is likely to both exaggerate costs and complexity in addition to associated risks with any API or functionality they are requested to expose. The CMA should impose an expectation on Apple that all future APIs should be open and made available to third-party browsers at the same time they are made available to Apple's own browser or system apps.

Are there other measures we should consider?

We have proposed a number of additional remedies including:

- Ensuring equality between Web Apps and Native Apps
- Mandating that Operating Systems ensure that Apps that use an In-App browsers respect a user choice of browser for all third party websites

We write about this in detail in the Remedies section.

6) Do you have any views on areas where we should undertake further analysis or gather further evidence as part of an MIR in relation to the supply of mobile browsers and cloud gaming?

The document contains many ideas about possible areas for investigation. The CMA could undertake further analysis in the areas of:

1. In-App Browsers and how they can be used to undermine competition
2. Specific Functionality on Safari that undermines the uptake of Web Apps such as Install Prompts
3. The ability to set alternative Web Payment Handlers on iOS and general support for the Web Payment APIs

4. Trusted Third Party Browsers on iOS

As mentioned in the introduction, competition between browsers has been stifled to such an extent by Apple that \there is no meaningful competition between browsers on iOS.
This is despite meaningful competition on all other operating systems including Apple's own MacOS as well as Windows and Android. All of these platforms have important issues that should be addressed by regulators, but none have the same severity or impact as the restrictions applied to iOS.

To effectively fix the issues we propose a combination of remedies that will allow third party vendors to bring their browsers along with their vision for the future of the web free of the anti-competitive interference that Apple has applied over the past decade.

This includes creating a new class of browsers called **Trusted Browsers**.

4.1. Trusted Browser Policy

The Trusted Browser Policy outlines in specific detail the requirements and processes the apply to Trusted Browsers including:

1. **Qualification Process**
   The process by which a browser is designated as a Trusted Browser

2. **General Rules**
   The general rules that a Trusted Browser needs to follow to maintain that designation

3. **Disqualification Process**
   The circumstances and process by which a Trusted Browser can lose its designation and the consequences that can occur

It is important that the browser policy indicates that:

1. **Rules should be objective and measurable**
   Each rule should be obvious in how that requirement could be met and what the evidence is required

2. **Rules should be clear, precise and unambiguous**
   The requirement should not leave room for wide interpretation and thus allow gatekeepers to block competitors on vague, poorly defined criteria

3. **Rules should be limited to the following topics:**

   a. **Security**

   b. **Browser as a Primary Purpose**
      Ensuring that the primary purpose of the application is to be a general purpose browser.

4. **Rules and Requirements should be important**
   All rules and requirements should be important and relevant. Minor requirements including non-security related technical or user-interface requirements should not apply. As is the case on other operating systems, gatekeepers can engage in constructive collaboration if they would like browser vendors to implement specific items.

5. **Processes should include Processing Timeframes**
   Each step in the process should include timeframes that the process must be complete by.

6. **Rules should not preference the gatekeeper**

We would encourage this process of developing these policies to be public to enable all parties to provide feedback. The web has many examples of public processes with which all vendors and developers are familiar.

Based on several rounds of review and consultation the CMA should select the requirements they believe are relevant to the criteria and then include those in the "Trusted Browser Policy" based on critical review of the feedback received.

4.1.1. General Principles / Aims

Trusted Browsers and Trusted Browser Policy should have the following goals in mind:

1. **Minimise Interference and Control**
   The purpose would be to provide a process by which browser vendors could gain access to special functionality to be able to bring their browser to these platforms while minimising the level of interference and control that gatekeepers have on other browser vendors while ensuring that only browsers that are committed to protecting user security are allowed

2. **Level Playing Field**
   To provide a level playing field in relation to the development of Browsers and Web Apps. Prevented side-agreements / deals with Browser Vendors related to limiting or providing access to special functionality in order to provide a transparent playing field.

3. To ensure transparency and an evidence based approach to policy

4. To ensure that Users / Consumers needs come first and that gatekeepers who have already sold devices at high-profit margins are prevented from blocking or interfering with how user's would like to use their own device.

5. To remove underhanded policies and behaviour that block competition

The general principles OWA would like to see incorporate are:

1. Principle of Open Access to Hardware and Software Functionality
2. Consumer needs come before gatekeeper or developer needs
6. Equality - Conditions applied to third-party browsers must also apply to the gatekeepers browser

4.1.2. Qualification Process

The Trusted Browser Policy outlines in specific detail the process by which a browser is to be accepted as a "Trusted Browser" by a specific gatekeeper. We'll label this the "Trusted Browser Qualification Process".

This qualification process could for example include items such as:

1. Ensuring that vendor has the staffing required to deliver a secure browser
   I.e. At least the minimum staff with the technical expertise to maintain a secure browser.

2. That the vendor has a sufficient patching and security policy

3. That the vendor has knowledgeable and contactable individuals when security issues inevitably occur

4. Ensuring that the browser vendor is an authentic identifiable company

5. That the applicant has a working browser whose primary purpose is to be a general purpose browser

Apple and Google should then be required to adopt the "Trusted Browser Policy" and use it to decide which browsers are accepted onto their platforms. Once a browser is accepted as "**Trusted Browser**" it will receive special privileges required to enable competition and to facilitate the web as an interoperable application development and distribution platform.

4.1.3. General Policies

The general policies should be rules or processes the browser's vendor must follow as a "Trusted Browser" to maintain that designation.

The CMA should look to create a document led by principles in consultation with Apple, third party browser vendors and others labelled the "Trusted Browser Policy" that will outline:

1. The process by which a browser is to be accepted as a "Trusted Browser" on a platform
2. A list of rules or principles that a browser vendor must follow as "Trusted Browser"
3. The process by which a browser can be removed as a "Trusted Browser" and removed from the AppStore OR devices OR blocked from AppStore updates
4. The process by which the "Trusted Browser Policy" can be amended or updated.

This "Trusted Browser Policy" should initially apply to iOS and Android but could be expanded to cover other Mobile and Desktop operating systems.

4.1.4. Disqualification Process

The following are some examples of what could be included in a disqualification process for a "Trusted Browser".

If a gatekeeper believes a "Trusted Browser" has breached the conditions of the policy then they can apply to the regulator along with strong evidence for:

1. Removal of the browser from the Trusted Browser List
2. Removal of the browser from the AppStore
3. Removal of the browser from user's devices
4. Blocking the browser from receiving updates
5. Requesting compliance of a specific action by a specific date

This could be breaches such as a browser either:

1. Does not keep their browser to a reasonable level of security by maintaining security patches in reasonable time frame

2. Deliberately and clearly attempts to undermine the security of the device from the perspective of the **consumer**.

3. Maliciously and provebly acts against the interests of the user

4. Its primary purpose is no longer as a general purpose browser

Then the gatekeeper should be allowed, subject to regulatory approval, revoke the browser entitlement and remove the browser from Apple's AppStore. Note that any process of revocation should assume the possibility of sideloading browsers in case that option is available in the future.

It is OWA's opinion that other aspects of browsers can be left to competition. Performance, Battery Life and Privacy are all areas browser vendors can compete in and should not be grounds for disqualification. If Apple wishes to remove a browser from the AppStore or from consumer devices then they must apply to the regulator.

4.1.5. Modifying the Process

Rather than the process and policies being static the process including for Qualification, Disqualification and the general policies should be changeable over-time. Gatekeepers and Browser vendors should be allowed to request changes backed with reasoning and evidence so that the process can be improved with changes ultimately being approved by the regulator

4.2. Trusted Browsers

Once browsers are on the trusted browsers list, they should then receive special abilities that will enable them to compete free of interference

Function	**Current iOS Browser Policy**	**Proposed Trusted Browser Policy**
Can be set as default browser	Yes	Yes
Primary purpose must be as a Browser	Yes	Yes
Can bring own Engine	No	Yes
Presumption of Access to **ALL** Software/Hardware API Functionality[1]	No	Yes
Reverse Engineering Allowed	No	Yes
Has full control to manage Web Apps installed via the Browser	No	Yes
Exempt from AppStore Review	No	Yes
Exempt from AppStore Guidelines	No	Yes
Subject to Trusted Browser Policy	No	Yes
Can run own Update Process	No	Yes
Requires Regulator Approval to Remove from AppStore	No	Yes
Requires Regulator Approval to Block Updates	No	Yes
Requires Regulator Approval to Remove From Devices	No	Yes
Prohibits Side-Agreements	No	Yes

4.2.1. Can be set as Default Browser

All browsers that pass the trusted browser selection process must be able to be set as the default browser.

This should be a true default and be respected by the operating system. Regulators should take care that a user's choice of browser is not silently eroded or undermined by the operating system. In-App Browsers, Voice Assistants & Search Widgets should all respect a user's choice of default browser.

4.2.2. Primary Purpose Must be as a Browser

To be designated a browser (and to maintain the designation) an App's primary purpose should be to act as a general purpose browser for browsing third party content on the web. Apps that are designed to primarily be another type of app such as Social Media, Chat, Commerce should not receive a browser entitlement. This should apply to both Trusted Browsers and Browsers under the current policy.

This is important to ensure that the user retains control over their browser choice so that they can choose a browser that best fits their utility, privacy and security needs but that will also ensure competition between vendors. If non-browser apps start declaring themselves as a browser simply to capture user traffic then this choice is undermined.

To illustrate this purpose imagine a user's default browser is Safari. The user also has Firefox and SocialMedia App installed. If the user taps links within Firefox, even though it's not their default browser, they would always want those links to be opened in Firefox and not to jump out to their default browser. Many users use multiple browsers as an easy way of maintaining different profiles or for different levels of privacy. Developers use multiple browsers for testing.

When users tap a link in SocialMedia App, it is in the user's best interest to have that link open in their default browser so their browser preferences including their privacy and security preferences are respected. This could either be via opening externally in the browser OR by using a "Remote Tab In-App Browser" which uses the user's default browser. If apps are able to circumvent this choice then this will undermine browser competition.

4.2.3. Presumption of Access

Trusted browsers must have an enforceable presumption of access to ALL hardware and software functionality that they require to implement ANY functionality they see fit.

Browsers act as an application platform for Web Apps and are currently the only method of free and open interoperable development between platforms. The functionality of those Apps depends on the capability of the browser and so if that browser doesn't have access to all the software and hardware functionality it needs to perform a function then those Apps can be deprived of those functions ensuring that the gatekeeper, the gatekeepers Apps or the gatekeepers native ecosystem is the only way of delivering that functionality.

Browsers are, of course, special in that they are responsible for **very** personal information and are repositories of both sensitive data (passwords, addresses) and a user's intent (extensions, privacy settings, and accessibility preferences). This is acknowledged in iOS through the recently created Browser Entitlement.

Browsers are special, technically, socially and from a competition perspective..

Based on this categorical uniqueness, it is right and proper, therefore, that applications that qualify as trusted browsers should be allowed to act as browsers. This implies a presumption of access, complete with the ability of vendors to bring their own engines and make use of any underlying system mechanisms that are required to support them; including, but not limited to currently private APIs that OS vendors restrict use of today. It is critical that browser vendors have the freedom to bring new functionality to browsers without being beholden to the gatekeeper as to which functionality should be exclusive or the exact manner in which that functionality should be used.

OSes that support browsers must be technically open to responsible makers of browsers to the same extent that first-party browsers are empowered, and developers of those browsers must be presumed to be acting in the best interests of users.

Said differently, browser vendors are unique in their level of responsibility, and to carry out their responsibilities effectively, must be empowered with full system access. To the extent an OS vendor disagrees with some aspect of this access, it should be subject to action **after the functionality has been implemented or announced** rather than as a gate on release. Removing these gates is critical to the future of browser competition.

Section 11. Detailed Breakdown provides a comprehensive list of software/hardware functionality that browser vendors SHOULD be allowed to request access to if they require it.

4.2.4. Reverse Engineering Allowed

The presumption of access to software and hardware should typically involve public APIs that are available to all trusted browsers.

As new APIs are developed for Safari, those APIs should be made available to other browsers vendors at the same time they are made available to Safari.

For functionality that has not yet been made available to third-party browser vendors, those browser vendors should be allowed to access private APIs and functionality. To discover how those APIs work browser vendors may need to use a process known as reverse engineering.

Apple typically includes terms to prevent Reverse Engineering, except often reverse engineering is the only way to work out how to implement specific functionality. Browsers vendors must be allowed to reverse engineer for the explicit purpose of implementing functionality.

- Reverse Engineering allowed for the purposes of implementing browser functionality
- Apple and Google must provide access to operating system APIs at the same time they provide access to their own browsers
- Eventually Safari should have no special access to operating system functionality

4.2.5. Control over Web Apps

Browsers should be granted explicit control (with or without user-interaction) to control Web Apps that they have installed including but not limited to:

- Install Apps
- Delete Apps that they have installed
- Update Apps that they have installed
- Focus the icon on the homescreen they have installed
- Open the App
- Update the Icon, Text

4.2.6. Exempt from AppStore Review

In the course of normal browser development, browser vendors are empowered to make autonomous decisions to be able to deliver their vision of the Web and Web Apps. They have the ability to make bold decisions from enabling cutting edge functionality to protecting user's privacy, to enabling entire new categories of applications. The combination of these decisions and direct competition between the vendors is what delivers the progressive improvement of the entire web platform to the benefit of consumers, developers and businesses alike.

Browser Vendors enjoy the autonomy to quickly push security updates, reduce initial download sizes on Windows, MacOS and Linux and implement functionality without the interference of any third party gatekeeper. However on Apple's iOS this empowerment is replaced with onerous, subjective rules and oversight which constrains major decisions effectively destroying the benefits of competition and thus the continuous progressive improvement. For this problem to be solved Apple's ability to interfere with the decision making ability of the browser vendors needs to be removed.

For this reason **Trusted Browsers must be exempt from the AppStore Review** process. The AppStore Review process can be (and, indeed, has been) weaponized to stifle competition. The review process can negatively affect the independent decision making at other browser vendors that normally occurs on open platforms.

Instead of the AppStore review process Apple can inform the explicit, non-subjective upfront rules as part of the **"Trusted Browser Policy"** and then apply to the regulator for remedial action if they believe a trusted browser is in breach of this policy. While Apple is allowed to enforce what amounts to arbitrary veto over updates and functionality, competition will be harmed. This veto should be replaced with the collaborative, equal process that browser vendors enjoy on other platforms.

4.2.7. Exempt from AppStore Guidelines

None of the AppStore Guidelines should apply to trusted browsers. Trusted Browsers should be exempt from all conditions not listed in the **"Trusted Browser Policy"**. If Apple thinks some of the AppStore Guidelines are reasonable and have clear, explicit, objective conformance guidelines then they should apply to have those conditions included in the "Trusted Browser Policy" which would then be subject to consultation, review and revision from other parties and ultimately approval from the regulator.

Splitting "Trusted Browser Policy" from the AppStore Guidelines will ensure that each condition that applies to trusted browsers is reasonable and can't be arbitrarily changed without a consultation process.

4.2.8. Subject to Trusted Browser Policy

Browsers that receive a "Trusted Browser" entitlement must abide by the "Trusted Browser Policy".

Each item on the trusted browser policy should be debated with each rule individually approved after consultation.

It is essential that the "Trusted Browser Policy" enables an environment that fosters both competition and collaboration rather than one of compelled conformance where the gatekeepers dictate functionality to browsers.

4.2.9. Can run own Update Process

If Browser Vendors decide that the AppStore distribution process does not meet their needs they should be allowed to run and manage their own update mechanism separate from the AppStore and Google Play Store.

It is essential that browsers are given the flexibility to update their own applications free from interference as they do on Windows and MacOS. This could enable browsers to use binary-diffing to ship far smaller updates and to decrease the window of vulnerability. It could also enable a smaller initial download of a browser, with the ability to download additional functionality. Gatekeepers' browsers are preinstalled so the initial download size of a browser doesn't affect them.

We would expect browser vendors to continue to use the AppStore distribution process initially but it's important to leave this door open as it may become vitally important for future browsers; smaller update sizes is a competitive advantage for browser vendors targeting markets where data is comparatively expensive or network speeds are generally sluggish.

4.2.10. Prohibit Side-Agreements

From anecdotal reports Apple has a history of using the chilling effects of private agreements to limit the capabilities of browser makers. These are secret side agreements, offered on a take-it-or-leave it basis, under NDA which amount to shakedowns that reduce the effectiveness of user choice in the market when products are eventually allowed to ship.

All such secret side agreements must be explicitly forbidden.

OS vendors must, instead, be forced to make the terms and conditions for allowing a browser onto their OS public (which we've labelled the Trusted Browser Policy). No side contracts which constrain the web platform features such as hardware or software API access of browsers should be allowed. The conditions for allowing a browser onto the OS should also be put up for consultation and ultimately approved by the regulator. The aim of removing these side-agreements is to provide a level, transparent playing field that browser vendors can then compete on.

Transparency is one of the best remedies, and ensuring that Operating System vendors understand that in relation to Browsers, Web-Apps and makers must be on notice that it will be applied liberally where questions regarding the presumption of browser access are concerned.

Apple has offered a set of extremely weak arguments to regulators that often seek to conflate its (often laudable) work in privacy with security of other sorts, whilst downplaying the ways in which Apple itself has created a panopticon via APIs it has liberally provided to native apps, e.g. IDFA, a persistent per-application identifier for users that trackers on the web have never had access to. Cupertino demands plaudits for solving problems it created, and many play along.

It cannot be stressed enough that security and privacy are related; there can be no privacy in insecure systems; but that they are not equivalent. Apple's attempts to substitute one for the other are misleading.

Side agreements between the gatekeeper and browser vendors that restrict the functionality the vendor can provide should be prohibited. All restrictions on the functionality that a browser vendor can provide should be contained within the **public Trusted Browser Policy rules.**

Side agreements and NDAs should be prevented from Constraining platform form functionalities, e.g

a. You are not allowed to access this functionality
b. You are not allowed do do X unless Y

If Apple tries to force side-agreements to restrict functionality or features then these should invoke regulatory penalties etc.

Pre-existing agreements that conflict with the Trusted Browser Policy need to be struck out.

4.3. Apple's National Security Argument

In order to discuss bringing competitive third party browsers to iOS it's important to discuss the arguments that Apple uses to prevent competition on iOS. Based on anecdotal accounts from several sources Apple often brings up *National Security* when attempting to defend its anti-competitive behaviour. National Security is likely an important consideration to governments and regulatory agencies so it is worth addressing directly.

Apple may seek to cast its anti-competitive policies regarding Web Apps as an essential ingredient in protecting the national security of states with high share of its devices in circulation. It may point to the more locked down nature of iOS as an improvement in security posture in comparison with poor behaviour of some Android OEMs with regards to how quickly and how long they provide security fixes. Secondly they may use the "secret sauce" security argument, i.e. the other browsers can't provide better security because we at Apple have hardware feature X and we won't provide access to it.

> *"WebKit is the only iOS process capable of accessing the just-in-time compiler or "JIT". The JIT allows apps browsing the web to quickly and efficiently render JavaScript content, which is valuable for users but also exposes a vulnerability that malicious actors can exploit. To mitigate the risks posed by the JIT, WebKit leverages tight integration with iOS hardware. Apple employs a highly effective hardware security extension APRR to prevent attackers gaining access to the JIT. Apple also implements Pointer Authentication Codes PAC to prevent attackers from gaining code execution outside of the JIT. PACs provide cryptographic signatures and authentication to function pointers and return addresses to protect against the exploitation of memory corruption bugs."*

Apple Response to Mobile Ecosystems Interim Report

All of this is a distraction from the ways in which Apple's policies materially impair the ability of organisations, including governments, from managing systemic risks that are introduced and exacerbated by Apple's own engineering and investment choices.

4.3.1. Enterprise and Consumer Risks are NOT Identical

Enterprise fleets of devices under management are different in kind from those owned and operated from those owned and operated by individuals. These devices are controlled and managed by a central authority such as a government organisation and private enterprise.

To the extent that National Security risk comes from devices owned by governments, those risks are managed through policy that sets out both the procurement process and the ongoing device management requirements.

Apple has so deeply under-invested in Safari/Webkit over the past 10 years that in combination with a ban on third party browsers enterprise/government customers are put at high risk with no method of mitigation. As rightly outlined by the CMA's Mobile Ecosystem's Final Report, the ban not only doesn't protect security, it could make it worse. The available evidence certainly suggests that Safari has worse security than both Edge/Chrome and Firefox. We dive into this evidence in Section 8.2 of "Bringing Competition to Walled Gardens".

Apple constantly lags behind others in support for high-security web functionality. Functionality like Chromium's Site Isolation and Microsoft Edge's Enhanced Security Mode puts Safari years behind in delivering the structural improvement already available in competing browser engines

For Nations that are concerned about these threat vectors, the solutions are somewhat straightforward: create a policy that prohibits the acquisition of devices, restrict browser engines, and a sunset policy for those that already exist. Gatekeepers who wish to keep the business of governments will need to do better and will have to adapt to keep that business.

Outside of enterprise/government, consumer risks present a different and unique challenge. Vulnerabilities inside important software leading to wide-scale exploitation or ransomware reduce the ability of governments to respond effectively. Private sector entities bear most of this risk as most national standards are advisory outside of government contracts. National security interests can be improved via diversity in supply chains, user choice and transparency requirements.

4.3.2. Recent Announcements only Highlight Long-Term Inaction

Apple recently announced "Lockdown Mode" and "Rapid Security Response" for its upcoming iOS 16 version. This is a good start, but does not fix the underlying problems that Apple's ban of third party browsers introduces and demonstrates how long Apple has blocked other browsers from shipping important security functionality. Functionality that users could have enjoyed the protection of for years.

Chrome and Firefox have had the ability to update their browsers without requiring an operating system upgrade (akin to Rapid Security Response) since their inception. Microsoft Edge allowed disabling JIT compilation over a year ago. While others provide an open marketplace that allows users to swap in safer applications for common use (e.g. Signal vs the default Android Messages app or Firefox with Extensions vs Chrome), Apple has strongly resisted this on iOS.

4.3.3. The United States Supply Chain Executive Order

Apple has argued in various forums that some browser vendors lack the required skills/displicine to ensure reasonable user safety and security. This is offered as justification as to why Apple should not have to compete with other browser vendors on IOS. Despite the disingenuous framing Apple provides, browsers require functionality that can bring legitimate security risks.

There is however a big difference between responsible browser vendors who invest deeply to maintain a browser and smaller browsers simply trying to make a quick buck through search referral deals or additional advertising. Extreme care is required to build and maintain a functional browser engine, and although Apple trails others in this regard it should still be considered in the "big kids club" from a funding and incident response perspective.

Until recently, there hasn't been a strong way to distinguish between parties inside and outside that club. Pursuant to US Executive Order 14028, the US National Institute of Standards and Technology (NIST) has begun work to define and identify the risks from critical software. This list includes Web Browsers (page 5) under the following rationale:

Category of Software: Web browsers
Description: Standalone and embedded browsers
Types of Products:

- Software that processes content delivered by web servers over a network, and is often used as the user interface to device and service configuration functions

Rationale for Inclusion:

- Performs multiple access management functions
- Supports browser plug-ins and extensions such as password managers for storing credentials for web server resources
- Provides execution environments for code downloaded from remote sources
- Provides access management for stored content, such as an access token which is provided to web servers upon request

As a result of this designation, browser vendors supplying software to the US government are now required to meet a long list of software development requirements including (but not limited to:

- Maintenance of a supply chain inventory for software components
- Specific patch-management practices
- Conformance with "minimum standards" guidance (PDF)

These standards and guidelines create a floor or a minimum level of conformance that browser makers have to adhere to and remove the space available to argue about bad browser vendor actors.

Of course, Apple is fully aware that these requirements have come into force because, presumably, its engineering and legal staff have been busy (like all browser vendors) in both ensuring and providing evidence that they meet these requirements.

While this is specifically a US conformance procedure it could be looked at as part of designing the **Trusted Browser Policy.**

4.3.4. Breaking Apple's Monoculture Can Improve National Security

An important aspect of the NIST's floor on critical software supply is that it does not force a single vendor or solution on all users. Instead, it promotes the adoption of healthy engineering and security practices that might otherwise remain unfunded, allowing both government agencies and other third parties to use their designations to judge risks more credibly and accurately.

One risk it does not address, however, is the risk of monoculture through scale.

Mobile devices are now the dominant computing form factor. Their security could not be more important. In the US and the UK, Apple sells more than 50% of all devices, all of which are forced to run a single browser engine across **all use of the web from within every application** (browser or not).

The risks from this monoculture are a clear danger.

Far from investing the most in necessary precautions to avoid risks to users and government acquirers of their technology, Apple has skimped for many years on essential security research, funding, and engineering. Others have led the way in aggressive sandboxing, code auditing, and hardware-level security for web engines. The fact that Apple trails in web security should not come as a surprise since relative to their primary competitors, they under-funded the development of Safari and Webkit. Security is just another casualty due to a lack of competition.

15 years into the journey of the mobile web, the risks to all parties from Apple's neglect are too grave to ignore. Something must be done to ensure that Apple can never again threaten the development of secure web engines via neglect.

Allowing responsible, motivated browser vendors to compete on iOS is the most important remedy. Providing broad access for vendors that conform to a set of public standards can ensure that governmental and private sector actors have solutions that can improve upon Apple's efforts without also taking on undue or unknown risks. Browsers that meet these public standards can receive an entitlement to access the low level functionality they need, browsers that do not can continue to use system provided WebViews.

4.4. Definition of Browser

An important question for policy makers in this domain is what is the definition of a browser.

> *A browser is an application that can register with an OS to handle http and https navigations by default.*

Another way to think about it is an application that can listen to and handle when a link is clicked/tapped etc. This means apps like Gmail are not browsers, although they occasionally display HTML-formatted email. Operating systems need to ensure that apps that use "In-App Browsers" do not undermine a user's choice of browser (A point on which Apple is likely to agree).

True browsers are products that users actively choose to handle their internet browsing when tapping on links. These products should enjoy regulatory protection as they act as both a gateway to the internet and as an application platform in addition to enforcing and protecting a user's wishes in relation to privacy and security. It is essential that the user retains control over these choices so that they can choose a browser that best fits their utility, privacy and security needs but that will also ensure competition between vendors.

An Application is a browser if:

1. Its **Primary Purpose** is to browse third-party content on the Web.
   - Third Party: Content not owned or controlled by the Browser Vendor

2. It serves as a **handler of http/https links**
   - The browser has the ability to register with the operating system and when a user taps/clicks an http/https link in other applications it can open in the browser

3. The primary interface displayed to the user on load is designed to allow them to browse third-party content with any other non-browsing functionality being strictly secondary.

4. It has the typical trappings of a browser (subject to change/innovation over time) such as URL address bar, the ability to refresh, search bar, multiple tabs etc.

Only browsers that meet these requirements should be allowed to receive a browser entitlement.

4.5. Entitlements

An entitlement, in relation to operating systems, is something that grants an app permission to perform some operation/ function or access some normally restricted hardware or software API. Without the entitlement the app can not access that functionality.

To perform special operations Apps on iOS receive entitlements. From Apple's documentation these can be described as:

> *An **entitlement** is a **right** or **privilege** that grants an **executable particular capabilities**. For example, an app needs the HomeKit Entitlement — along with explicit user consent — to access a user's home automation network. An app stores its entitlements as key-value pairs embedded in the code signature of its binary executable.*

We can divide entitlements into several categories:
1. **Private Entitlements** that are only ever provided to Apple's own Apps
2. **General Entitlements** that are generally available to all apps
3. **Exclusive Entitlements** that Apple only provides exclusively to one company or very few companies.

An example of one of these is "com.apple.developer.web-browser" which allows an app to register itself as a web-browser with the system. We will refer to this entitlement as the "browser" entitlement.

These entitlements can be used to provide access/privileges to some apps that others might not receive such as:

- Bluetooth
- NFC
- Ability to run a server on the phone (i.e. a tcp/udp port on the phone)
- Ability to act as a VPN
- Ability to open specific types of network connections
- Ability to wake the device up in response to specific events
- Ability to access hardware

Apple can use these entitlements to provide special preference for their own apps with functionality that they don't provide to third parties. As an example Cisco in the past used to receive functionality to create special types of VPN that no other VPN app received.

J's Entitlement Database (which was updated for iOS 15.2) contains a list of all the stock apps that ship with iOS and the entitlements they receive.

An example is the entitlements that iOS Safari receives:

http://newosxbook.com/ent.jl?osVer=iOS15.2&exec=/Applications/MobileSafari.app/MobileSafari

4.6. Browser Entitlement

The Browser Entitlement (com.apple.developer.web-browser) is what allows a browser to be set as a default browser and list and register as an app that can handle http/https links.

OWA would recommend a new entitlement called the Trusted Browser Entitlement (com.apple.developer.trusted-web-browser).

OWA recommends that the Browser Entitlement remains as is (com.apple.developer.web-browser) and browser vendors are able to continue to use the system provided WebView to build a browser either by choice or if they are unable to meet the stringent requirements to be required to become a **Trusted Browser**. Apple must provide the Trusted Web Browser entitlement to any browser which has passed the Trusted Browsers qualification. Apple can not revoke a Trusted Web Browser entitlement without explicit approval from the regulator.

5. Gatekeeper Advantage

Apple decides whether or not a browser is allowed to be listed on the AppStore and how it is listed. Apple and Google maintain many advantages for their own browsers over their competitors.

Specific analysis should be given to:

Search Term Order
The order in which the browsers appear in response to specific search terms

The Age Rating (iOS)
All browsers are currently set to 17+ which may stifle installations, but this doesn't affect Safari since it comes pre-installed. Consideration should be given as to whether the age rating can dissuade users from installing a third party browser.

If the Age Rating is found to affect user installation, an appropriate intervention that eliminates the competitive advantage needs to be found either by removing the general age rating or by removing Safari's current and past advantage of being the default browser.

Interventions related to AppStore Listing
In general Apple should be **required** to list any browser in the AppStore that has "Trusted Browser" status and Apple should only be able to remove a browser from the AppStore if that browser has its "Trusted Browser" status revoked.

Preemptive Ban on Dark Patterns
There should be a pre-emptive ban on employing any dark patterns to either promote Apple or Google's own browser or in any way disadvantage competitors browsers.

6. Proposed Remedies

In this section we outline goals that remedies should achieve, in priority order, to restore meaningful competition.

We acknowledge that regulatory progress is often intermittent, and that legal text needs to be written to be broad, while anticipating legal challenges and attempts to delay progress by gatekeepers. It is our hope that by stating exactly what solutions are necessary to problems facing competition in the software industry, it may provide some insight into how to construct statutes and regulations so that they can avoid expensive and (more importantly) slow appeals processes by the gatekeepers. An important aim of proposals should be to create clarity for all parties.

We argue that by default, there should be a *presumption of software and hardware access* for competing browsers and ask regulators to acknowledge the special role of browsers in promoting competition, fair access, and user security through competition. It is important to focus on competition between browser vendors as the mechanism that drives benefits for both users and developers.

From most to least urgent, we hope that market interventions achieve:

- ☐ **1. Third Party Browser Engines (iOS)**
  Reversing Apple's ban on competing browsers and browser engines by requiring Apple to allow third party browser engines.

- ☐ **2. First Class Web App Support (iOS / Android)**
  Web Apps integrated into Operating Systems to the same level as Native Apps.

- ☐ **3. In-App Browser Abuse Protection (iOS / Android)**
  Preventing In-App Browsers from hijacking a user's choice of browser.

- ☐ **4. Browser/Web App Equality (iOS / Android)**
  Ensuring Browsers and Web Apps are capable by having the same level of access to hardware/software as Native Apps, the gatekeeper's browser or operating system services.

- ☐ **5. App Store Support for Web Apps (iOS / Android / Windows)**
  The ability to submit Web Apps to each of the app stores easily without having to purchase a specific device (i.e. without a Mac).

- ☐ **6. Safari Core Functionality (iOS)**
  Ensure that Apple (as a **primary** gatekeeper) provides core/basic functionality for Web Apps within Safari to enable them to compete against Native Apps.

- ☐ **7. No Chrome Preferencing (Android)**
  Ensure that Google can not unfairly provide any preference to Chrome via licensing

agreements or app/operating system functionality.

☐ **8. Website Transparency Obligations (All)**
Ensure Websites have to publish reasons to users for either not supporting a particular browser **or** if prompting them to use another browser.

6.1. Rationale

6.1.1. Third Party Browsers (iOS)

Reversing Apple's ban on competing browsers and browser engines by requiring Apple to allow competitive third party browser engines is key to unlocking competition with Apple's walled garden from an open, safe, and interoperable platform (the Web).

Rationale:

The lack of third party browsers on iOS:

- Removes competitive pressure on Apple to invest in their own browser.
- Incentivises Apple to not implement support for web functionalities, in order to protect App Store revenue.
- Provides no incentive to respond to developer needs or ensure that Safari is reliable/not buggy.
- Protects Apple's Google Search revenue but deprives other browsers of that revenue.
- Prevents Web Apps from being viable on iOS
- Removes the Web as an Interoperable Platform between both Android and iOS

Ending this ban will:

- Enable the ability to create functional Web Apps.
- Reduce mobile application development, distribution and maintenance cost for business, fostering innovation and lowering prices for end users.
- Encourage the development of interoperable Web Apps, easing the switch between mobile operating systems, and fostering the emergence of new ones.
- Protect the future against anti-competitive behaviour and remove incentives to stall progress.

IMPORTANT:

> Making a browser work on a new platform is a complicated and expensive task for any browser vendor. To ensure competition is restored as quickly as possible, browser vendors such as Google, Microsoft and Mozilla need to be given the **clear signals** by the CMA and other bodies that a goal of regulation is explicitly to enable such "ports" and to create a clear expectation of Apple to facilitate these ports. Apple must allow responsible browser vendors who receive the Trusted Browser Designation to access **all** system features necessary to bring the best and most complete browsers that they can. This must include access to currently private APIs that Apple does not provide access to. Apple must not be allowed to obstruct other browser vendors with any onerous requirements.
>
> Apple has given nonsensical security arguments to delay compliance in response to other regulatory bodies. Care in messaging should be taken to ensure the plain meaning of regulatory intent is not subverted by unfounded claims. No responsible party will object to any **reasonable** security policies (backed by convincing evidence) to protect users. Security policies need to balance utility and security, and this should always prioritise the users interest over the Gatekeeper or Developers.
>
> All operating system vendors delegate some of the responsibility regarding security to browsers, including Apple. It is important to note that it is not the gatekeepers sole responsibility to protect users. There will be many instances where providing low level system access to competitors with strong security track records and dedicated security teams is massively in the consumers interest, even though this means delegating protecting the user to these competitors. These competitors can abide by reasonable, narrow-scope, non self-preferencial security/privacy rules backed back by compelling evidence.

6.1.2. First Class Web App Support (iOS / Android)

To create credible competition with gatekeeper app stores and platforms, it isn't enough that competing browsers be given the ability to access increased capabilities when users visit sites in a browser. All modern mobile operating systems also support mechanisms for "installing" web sites as apps (a.k.a. Web Apps / PWAs).

All browsers on iOS and Android must be granted the ability to create and manage this class of applications. They must also appear fully integrated with the operating system, and provide the user control over their operation **without preference** to gatekeeper controlled and taxed Native Applications. On Android this would mean providing WebAPK minting to third party browsers. Apple may need to provide new APIs for competing browsers to support this use-case, or access to Apple's internal mechanisms for delivering this functionality through Safari today.

Rationale:

First Class Web Apps on Android and iOS will:

- ○ Ensure that Native Apps do not receive any artificial preferential treatment over Web Apps.
- ○ Provide users a method of controlling the permissions (i.e. notifications, bluetooth, etc.) of Web Apps in the same manner they control Native Apps.
- ○ Ensure that the web ecosystem that is interoperable between all devices and free and open can compete with proprietary platforms that only work for specific ecosystems.
- ○ Substantially decrease costs of developing safe, cross-platform apps by providing a standards-based alternative.
- ○ Place pressure on native ecosystems to bring fees in line with value provided to developers, reducing deadweight economic losses. This is accomplished by providing a credible alternative that developers can switch to if the fees are unreasonable.
- ○ Enable companies and developers to build "apps" on the web. To the extent that Web and Native Apps are indistinguishable, it will increase competition and finally make true Apple's claim that Web Apps are suitable substitutes.

6.1.3. In-App Browser Abuse Protection (iOS / Android)

Apps should not be able to undermine a user's choice of browser or otherwise **intercept, monitor or control** a user's web browsing to third-party sites. Apps that wish to act as a browser should explicitly compete *as* browsers. A user's choice of browser along with preferences, security settings and their extensions (for example privacy enhancing extensions) needs to be respected by every app that loads non-collaborating web content (e.g., excluding first-party pages and ads). This should be enforced by the operating system.

The test to apply should be "*what should happen when a user taps a link inside an app?*"

Rationale:

- Improves browser competition by ensuring that apps that act as browsers explicitly compete with other browsers.
- Ensures that Web Apps can not be broken, or prevented from being installed from within a custom In-App Browser.
- Ensures that a user's choice (and thus competition) is respected.

- Improves user privacy by ensuring their preferences, security settings and extensions are respected and by not letting apps intercept user browser traffic.

- Prevents poorly implemented In-App Browsers from breaking Web App functionality and Web App Installation.

This is covered in detail in our regulatory submission *"In-App Browsers - Subverting User Privacy, Competition and Choice"*.

Regulatory pressure needs to be applied to ensure that operating systems require that apps respect user choice when it comes to browsers.

Browser competition will be significantly harmed if ANY app can take on the role of a browser without having to compete as a browser.

6.1.4. Browser/Web App Equality

Browsers and their Web Apps need sufficient operating system access to provide equivalent functionality to gatekeeper's own apps, browsers, system provided functions, and apps distributed via the gatekeeper's online stores.

Gatekeepers should provide browsers with full access to hardware APIs, including NFC, Bluetooth, USB, serial, HID, accelerometers, proximity sensors, elevation sensors, temperature sensors, light sensors, hardware buttons, location sensors, and other sensors available to Native Applications.

The aim of this remedy is to remove any artificial restrictions on what browsers or Web Apps can do either via software or hardware, owing to the superior untrusted-by-default security model that modern browsers enforce regarding these capabilities. Operating systems restrictions are therefore redundant when content is loaded in a sufficiently secure, modern browser.

The EU's Digital Markets Act contains interesting language in this regard (emphasis added):

> *"If dual roles are used in a manner that prevents alternative service and hardware providers from having access under equal conditions to the same operating system, **hardware or software features that are available or used** by the gatekeeper in the provision of its own complementary or supporting services or hardware, this could significantly undermine innovation by such alternative providers, as well as choice for end users. The gatekeepers should, therefore, be required to ensure, free of charge, effective interoperability with, and access for the purposes of interoperability to, the same operating system, hardware or software features that are available or used in the provision of its own complementary and supporting services and hardware. Such access can equally be required by software applications related to the relevant services provided together with or in support of the core platform service in order to effectively develop and provide functionalities interoperable with those provided by gatekeepers. The aim of the obligations is to allow competing third parties to interconnect through interfaces or similar solutions to the respective features as effectively as the gatekeeper's own services or hardware."*

Draft EU Digital Markets Act (11 May 2022) - Page 46

We recommend that the UK and the CMA adopt similar language in legislation/rulings to enable browsers and Web Apps. While browsers may not use all access immediately, it is important that they be allowed to perform any function that gatekeeper software or hardware enables. We are not suggesting this be true for every class of program on iOS; just browsers, and only browsers with a strong security track record.

The word **available** is important as there may be important functionality that is "available" either in the hardware or in the software that is not used by the gatekeeper but which they still prevent access to. It's important to open the door for Browsers and Web Apps to access this functionality as this will spur innovation.

6.1.5. App Store Support for Web Apps (iOS / Android)

It should be possible to submit Web Apps to app stores without the developer being required to purchase or own a specific device (i.e. a Mac with Xcode installed). To accomplish this, both a web-based process to submit the Web App is needed, along with removal of rules preventing Web Apps which otherwise conform to Gatekeeper policies from being submitted to the App Store.

Gatekeeper's should remove strictly unnecessary steps or requirements from the process of submitting an app.

Rationale:

- Ensures that firms with limited resources do not have to choose between building Native Apps for discovery in stores vs. a website for discovery on the open web.
- Increases the likelihood that developers will choose open technologies that will work across multiple ecosystems, lowering costs and improving competition.
- Fosters investment and expertise into the free, open, and interoperable web ecosystem instead of siloed native ecosystems. This directly lowers costs to businesses and users.
- Reduces the cost of developing for the app stores by not requiring the developer to purchase expensive hardware or maintain expensive, per-target-OS application wrappers.

6.1.6. Safari Core Functionality (iOS)

As an operating system gatekeeper with control over the default browser, Apple has the ability to directly influence the uptake of Web Apps. Due to their ownership and collection of fees from iOS's App Store, Apple has strong incentives to limit the viability of Web Apps as an interoperable and rent-free replacement for native iOS applications which can only be discovered within its rent-extracting App Store.

Although **competition between browsers is the most important and primary driver of functionality**, it's essential that Safari, as the default browser, should be required to implement basic functionality needed for Web Apps to ensure they are competitive. These features have been carefully chosen as the **bare minimum** that Safari would need to add for Web Apps to be viable using iOS Safari.

- **Install Prompts/Installability**

  At a minimum, Web Apps should be as easy to discover and require as few steps to install as Native Apps. This includes the ability to display a banner and prompt the user to install an app from Safari. As described in *"Bringing Competition to Walled Gardens"* sections 5.4.5 and 5.4.3.1 the Web App install procedure is deeply obscured, whereas this is not the case on any other operating system.

- **Notifications**

  Push Notifications are essential for a wide range of applications

  Note: This has been announced by Apple but **is not currently available**. It is expected to be released in 2023, although Apple has declined to be more specific than that. It is important to ensure that the implementation of notifications that Apple delivers is competitive with what is available in Native Apps.

- **FullScreen, Badging, Deep Links, Screen Orientation Lock**
  It is important to note that the key driver of future functionality should be competition and that due to the highly complicated nature of developing functionality that browser vendors should in general have autonomy into delivering their vision of the web.

6.1.7. No Chrome Preferencing (Android)

Google should not use their control over the operating system to provide unfair preference to their own browser, Chrome, either through the operating system or agreements with partners (i.e. device manufacturers).

This should include:

- **"Google Search App" (a.k.a. "AGA"/"AGSA") Must Respect Default Browser Choice**
  Google forces manufacturers to place the Google Search App on the first (primary) homescreen of most Android devices. This search box drives an *enormous* amount of web traffic and fails to respect user's choice in default browsers, instead causing Chrome to be invoked whenever users tap on search links that would otherwise take users to their default browser.

  This undermines browser competition on Android and is technically unjustifiable.
  Google can know if the user's default browser is not Chrome and must be made to respect this choice especially with what is arguably the single most valuable browser integration in any program or operating system.

- Mobile Application Distribution Agreement (MADA) should not require Original Equipment Manufacturers (OEMs) to prefer Chrome, as for example requiring Chrome to be the system default and on the first homescreen.

- Users must have a way to disable "In-App Browsers" system-wide and gatekeeper rules should enforce this policy.

Rationale:

For healthy browser competition Chrome should win market **share through user choice and merit**, **not control of the operating system**, dark patterns or via contracts with device partners.

6.1.8. Website Transparency Obligations (All)

Gatekeepers such as Apple, Google and Microsoft control high profile websites. If some of these Web Applications do not work in a specific browser, that can cause users to choose another browser. This has been reported many times as an issue by multiple browser vendors, including Opera and Mozilla.

Imagine the fictional scenario where Google decided not to support any browser other than Chrome for Youtube. This would have a dramatic negative effect on browser competition. There are, however, legitimate reasons why some applications can't work in some browsers including serious bugs and missing features.

OWA suggests that where a Gatekeeper's website does not support a browser which has above a 2% market share, they be compelled to publish a document containing detailed reasoning that prevents support of certain engines. A convention could be established for these files to improve their discovery, e.g. a `compat.txt` file hosted on the website root (or in the `./well-known/` directory). However provided, they should include details regarding:

- **Bugs**
  If the browser contains speed, bugs or stability issues that affect a user then the document must contain a detailed description of these bugs with links to bug tickets filed against engine projects.

- **Functionality**
  If the application requires specific functionality that the browser does not support it must provide links to the missing functionality with a rationale why the application/website requires that feature. For example the current version of photoshop does not yet support browsers other than Chrome, but for legitimate reasons.

If a gatekeeper's website wishes to display a "switch to a better browser" dialog then this must also include specifics as to why it is better in the browser they are suggesting which needs to be accurate. In the banner the gatekeeper should provide a link to a user readable description of these reasons.

Gatekeepers must also provide **"try anyway"** links in these UIs to ensure that users of other browsers that *may* be compatible are able to proceed to their applications, even if they appear slightly broken.

Finally it is important that these rules do not block gatekeepers from providing cutting edge functionality to consumers. For example imagine the following hypothetical example:

> *Microsoft wishes to create a website containing a number of Web App games that use new APIs. Due to the cutting edge nature of the games it is not possible to support browsers that do not support these or equivalent APIs. Microsoft commits to supporting all other browsers where feasible without hindering the quality of the games or excessively increasing the development cost.*

We believe under these circumstances it would be unreasonable to either prohibit Microsoft adding these APIs to their version of blink or prohibiting them from producing a website/Web Apps that rely upon them.

It would also be worth considering applying these rules to all of the largest companies rather than just gatekeepers.

These obligations will help to protect competition, by providing transparency to browser vendors/developers and end users.

7. Web Payments

Apple has not (and does not intend to) support third party payment handlers including through the Web.

As reported by the BBC, "Affinity Credit Union" said in a recent class action lawsuit against Apple's anti-competitive practices stated:

> *"Apple charges issuers a 0.15% fee on credit transactions and a flat 0.5 cent fee on debit transactions using Apple Pay, while Android-based rivals charge nothing"*

> *"card issuers—the proposed class here—pay a reported $1 billion annually in fees on Apple Pay and $0 for accessing functionally identical Android wallets"*

> *"These fees generated a reported $1 billion for Apple in 2019, and this revenue stream—earned from card issuers—is predicted to quadruple by 2023"*

> *"None of these Android tap and pay solutions charges transaction fees to either users or card issuers"*

> *"In the Android ecosystem, where multiple digital wallets compete, there are no issuer fees whatsoever"*

> *"If Apple faced competition, it could not sustain these substantial fees. Alternative mobile wallets, including Google Pay, would be downloaded onto iOS devices, and card issuers would agree to make their cards available on those substitute mobile wallets at zero cost and would not agree to make their cards available on Apple Pay unless and until Apple reduced its price to the competitive level."*

> *"to participate in Apple Pay, an issuer must agree not to impose a surcharge on a cardholder's Apple Pay transactions. This rule prevents issuers from using differential pricing to drive cardholders to lower cost alternative modes of payment."*

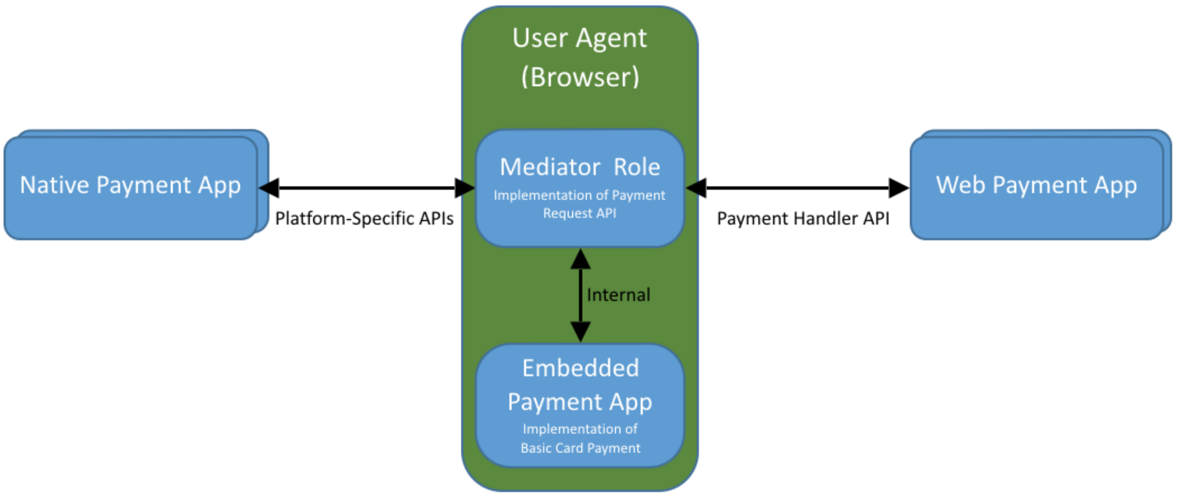
> *"developers are not permitted to use NFC for payment apps that might compete with Apple Pay"*

> *"Without any cost-based justification, Apple charges higher fees on credit than debit (15 basis points (.15%) vs. 0.5 cents ($0.005))."*

Affinity Credit Union

There are a number of core web payment APIs including:

1. Payment Handler API
2. Secure Payment Confirmation (SPC)



It is essential for competition that third-party payment handlers are enabled and that these Web APIs are available in Safari as the default browser.

Apple should be compelled to support these Web APIs for Safari and other browsers including allowing Native Apps and Web Apps to register as Payment Handlers.

8. Detailed Breakdown

This section is designed to contain the granular detail about each desired effect with specific points that can be used to determine if the remedy is successful.

- ☐ **1.Third Party Browsers (iOS)**
  - ☐ **1.1. With own Rendering Engine**
    Browser's must be allowed to choose and ship their own rendering engines without restriction to how the rendering engine should work or behave. The rendering engine is critical for pushing web capabilities forward.
  - ☐ **1.2. With own Javascript Engine**
    Browser's must be allowed to ship their own javascript engine or for any other programming language and have access to any hardware functionality required to make the engine performant and secure.
    - ☐ **1.2.1. Including JIT**
      Javascript Engines with JIT (Just in Time Compilation) must be allowed for performance and compatibility reasons.
  - ☐ **1.3. No App Store Restrictions**
    Browsers should not be subject to the AppStore Guidelines. For competition it's essential that browsers should be able to bring their vision of the web and apps without interference from the operating system except on narrow scope and heavily justified security issues which are in the end users interest. It is important that app stores' rules impose no restrictions related to the user Interface, functionality or technology. This is covered in detail in the Trusted Browser Policy section.
    - ☐ **1.3.1. Must be installable from the App Store**
      Apple must not be allowed to block browsers from being in the AppStore
    - ☐ **1.3.2 No Dark Patterns**
      Apple must not be allowed to use any method to dissuade or discourage users from installing another browser, such as warning messages or similar prompts. Apple should win browser share without resorting to using their control of the operating system to do it.
  - ☐ **1.4. Quick Update**
    Browsers need the ability to push updates very quickly and should be exempt from AppStore Review This is critical for minimising the end user's window of vulnerability (i.e. the length of time an end user is exposed for) when bugs need to be patched. This is covered in detail in the Trusted Browser Policy section.

☐ **1.5. Ability to Install Web Apps**

Browsers need the ability to install Web Apps and for those Web Apps to be deeply integrated into the operating system.

An installed Web App should be indistinguishable from a Native App.
It should be technically possible (but not required) for a browser to create a Web App that also has the ability to install other Web Apps to act as app stores.

☐ **1.5.1 No User/Operating System Interaction**

The browser must be given full control to be able to install and update a Web App by itself without the operating system interrupting the process. Browsers need the ability to install Web Apps without interaction from the user for specific use cases such as syncing Web Apps installed between devices, and when restoring from backups.

☐ **1.5.2 Ability to Open Apps**

Third party browsers need the ability to open / switch to any web app they have installed via an API.

1. So that after install a browser can automatically open an app

2. So that browsers can optionally build lists of apps which they can open

3. This is an essential for building Web App stores.

☐ **1.5.3 Ability to Focus Icon of Installed Apps**

Browsers need the ability to optionally bring into focus the pane of the installed app on the dock. This means switching to the correct pane of the dock for the installed app. This can be inline with the placement behaviour of native apps or "add to homescreen" but it should be up-to the browser to decide whether to open the app after install or whether to focus the icon.

☐ **1.5.4 Ability to Delete Apps**

Browsers need the ability to delete Web Apps that they have installed. This could be used for syncing Web Apps between devices, for managing Web Apps from within the browser and for browsers or Web Apps that maintain Web App / PWA App Stores.

☐ **1.5.5 Icon/Text Control**

The installing browser must have control to set and update both the name and the icon of the Web App. The installing browser should be able to indicate different icons/text for each system language.

- [ ] **2. First Class Web App Support (iOS / Android)**
  Web Apps once installed should have the same level of integration into the operating system as Native Apps.

  This includes:

  - [ ] **2.1. Control over Web App Settings**
    The installed Web App should appear as an app on the Settings.app page, and on each of the privacy menus (where relevant).

    - [ ] **2.1.1.Custom Settings per Browser**
      Each browser can add individual settings per to its installed Web Apps.

    - [ ] **2.1.2.Custom Settings per App**
      Each app can add custom settings to its Settings page.

  - [ ] **2.2. Notifications**
    - [ ] 2.2.1. Must be Free
    - [ ] 2.2.2. Must not Require an Apple Developer Account
    - [ ] 2.2.3. Delivery time equal to Native Apps.
    - [ ] 2.2.4. Users can easily enable/disable them.
    - [ ] 2.2.5. Full support of the Push API specification.
    - [ ] 2.2.6. Must work in installed Web Apps
    - [ ] 2.2.7. Must work in all Third Party Browsers
      - [ ] 2.2.7.1. Webkit WebView Based Browsers
      - [ ] 2.2.7.2. Browsers with Own Engine

  - [ ] **2.3. No Double Permission Prompts**
    Only the Web App should need permission to perform an action. For example if the Web App has permission to send notifications and its installing browser app does not, then the Web App can still post a notification.

  - [ ] **2.4. Equivalent Controls/Integration to Native Applications**

    - [ ] **2.4.1. Context Menu on Home Screen**
      The long-hold side menu needs to be supported.

    - [ ] **2.4.2. Integration with VoiceAssistants**
      Browsers and Web Apps should be able to fully integrate with Voice Assistants the same way Native Apps can.

  - [ ] **2.5. Appears in all areas Native Applications are listed**
    Browsers and their installed Web Apps should be given sufficient integration

into the OS that they can be listed anywhere Native Apps are.

☐ **2.6. Storage Control**
Users should be able to control a Web Apps storage if they want too. Web Apps should appear on the iOS "iPhone Storage" list as separate entities.

☐ **2.6.1 Offload App**
Consideration should be given as to whether Offload App should be implemented for Web Apps.

☐ **2.7. Reliable/Permanent Storage**
The operating system can **not** delete Web App storage even under storage pressure, unless specifically allowed by the browser that installed it **or** by specific user request either via the installing browser or by operating system storage controls such as "Delete App".

☐ **2.7.1. No Artificial Storage Limits**
Web Apps should not be placed under any storage limitations subject to the needs/wants of the user.

☐ **2.8. WebAPK Minting or Equivalent (Android)**

Android must provide other browsers the ability to mint WebAPKs or an equivalent mechanism of installing fully integrated Web Apps.

Longer term, there is room for debate as to whether Google should be allowed to continue having **exclusive** access to mint WebAPKs from their server infrastructure and whether they should delegate some of this control to competing browser vendors similar to the trust delegation process used by SSL certificate chains.

☐ **2.8.1 WebAPK Minting must be fast**
If Google is to retain exclusive control over the minting service and provides no alternative for browsers to be able to install Web Apps locally, they should be responsible for minimising the time taken to install a Web App. Currently, there is often a delay of several seconds while waiting for Google's minting server to respond which could adversely affect all Web Apps.

- ☐ **3. In-App Browser Abuse Protection (iOS / Android)**

    - ☐ **3.1. Third-Party Content only open via User's Default Browser**
      Operating systems need to protect users and competition by ensuring that if a user visits a web page that it opens that web page using the user's default browser. The only exception should be for first-party content so that apps can use a WebView to render their own content or from co-operating third parties such as ad providers.

    - ☐ **3.2. User can choose Remote Tab In-App Browser (IAB) or their Default Browser Per App**
      Web content should always use their default browser to render content although a user should be able to choose whether web content should open within the app or whether it should open externally in another browser.

        - ☐ **3.2.1 Default Behaviour**
          The user should have easy control to set the default behaviour of all apps to either open in an IAB or in their default browser.

    - ☐ **3.3. First/Second Party Content Opt-In Mechanism**
      A mechanism to ensure that WebViews can only load first and second party content would need to be implemented to ensure they can strictly only open first and second party content.

    - ☐ **3.4.Only actual Browsers can be a Default Browser**
      It is important for competition that popular apps defer browsing the web to a user's default browser. Only apps that are actual browsers should be able to receive a browser entitlement. For example, popular social media apps and messaging services should **not** be able to receive a browser entitlement.

    - ☐ **3.5. User's must be able to install Web Apps directly from within IABs**
      When a user is browsing the web from within an In-App Browser they must be able to directly install a Web App.

- ☐ **4. Browser/Web App Equality on (iOS / Android)**
  Browser's should be able to access the following functionality upon request. Web Apps should be able to access that functionality as allowed via the Web Apps installing browser.
  - ☐ **4.1. General**
    - ☐ **4.1.1. Hardware**
      Browsers and Web Apps should be able to make general use of **any** hardware the device provides including functions enabled by the hardware but disabled at the software layer.
    - ☐ **4.1.2.Same access as Gatekeeper's Browser**
      Browsers should receive (but not limited too) all the same abilities/the gatekeepers browser receives.
    - ☐ **4.1.3. Same access as Native Apps**
      Browsers should be able to (but not be limited to) use/access any function Native Apps use.
    - ☐ **4.1.4 Same access to System Apps and Functionality**
      Subject to narrow scope security concerns, browsers should have access to (but not be limited to) the same functionality that System Apps use.
  - ☐ **4.2. Communication Protocols**
    - ☐ NFC
    - ☐ Bluetooth
    - ☐ USB
    - ☐ Serial
    - ☐ HID
    - ☐ **Networking**
      - ☐ Low Level Networking (TCP/UDP)
      - ☐ Firewalls
      - ☐ VPNs (including Apple's Private Relay feature)
  - ☐ **4.3. Sensors**
    - ☐ GPS Location/GPS
    - ☐ Accelerometer
    - ☐ Step Counters
    - ☐ Proximity Sensors
    - ☐ Elevation Sensors
    - ☐ Temperature Sensors
    - ☐ Light Sensors
    - ☐ Other

- ☐ **4.4. Health Related Sensors**
    - ☐ Heart Rate
    - ☐ Glucose Monitoring
    - ☐ Other

The user should be in control of their health data including the ability to use other apps to collect, manage and store that data.

- ☐ **4.5. Hardware Buttons**
    - ☐ Volume Up/Down
    - ☐ Off/On
    - ☐ Ring/Silent
    - ☐ Back Tap

- ☐ **4.6. Lower Level Functionality**
    - ☐ CPU Functionality
        - ☐ CPU Utilisation
        - ☐ Number of Cores
        - ☐ Clock Speed
        - ☐ Type
        - ☐ Compute Pressure
    - ☐ GPU Functionality
    - ☐ Memory Related Functionality
        - ☐ Total Used / Memory
        - ☐ Speed
        - ☐ Type
    - ☐ Disk Related Functionality
    - ☐ Security / Cryptographic Related Functionality
    - ☐ Video/Audio/Image Encoding Functionality
    - ☐ Managing Processes (Spawn/Terminate, etc.)

- ☐ **4.7. Telecommunications**
    - ☐ Baseband/Radio/5G/4G Related Hardware Functionality
    - ☐ Telephony Related Functionality (Receiving/Making Calls)
    - ☐ SMS/MMS Related Functionality
    - ☐ Wake from Sleep
    - ☐ Ability to swap about the default phone call application
    - ☐ WIFI (including ability to connect, disconnect, hotspot etc)

- ☐ **4.8. Device Management (iCloud Replacement)**
  - ☐ Device Backup
  - ☐ Device Restore
  - ☐ App Data Backup
  - ☐ App Data Restore
  - ☐ **Device Fleet Management**
    Managing fleets of devices, with the ability to apply settings/policies remotely to those devices.
  - ☐ **Locate Devices**
    The ability to run an interoperable device finding and remote management service (i.e. Find your Android/iOS/Windows/Linux/mac devices from the same service)
  - ☐ Remote Lock
  - ☐ Remote Alarm

- ☐ **4.9. Filesystem Access**

- ☐ **4.10. Software Access Equality**

  - ☐ **4.10.1 Media Player/Music Library**
    The ability to interact with or **replace** the system default media player including on the lock screen (i.e. Apple Music).

  - ☐ **4.10.2.Payment Services**

    - ☐ **4.10.2. Replacement Wallet**
      The ability for browsers to integrate or replace with the default payment service (i.e. Apple Pay).

    - ☐ **4.10.3 Third-Party Payment Handlers**
      The ability for Native and Web Apps to register as third party payment handlers for the Web Payment APIs

  - ☐ **4.10.3. Voice Assistants**
    The ability to interact/integrate and replace voice assistants.

  - ☐ **4.10.4.Time Tracking/Focus Management**
    The ability to track application use and manage application time limits and other restrictions.

  - ☐ **4.10.5.Widgets**
    The ability to create and place home screen widgets

  - ☐ **4.10.6. Control Center (iOS)**

The ability to add icons and widgets to control center

- ☐ **4.10.7. Privacy and Access Monitoring (App Privacy Report)**
  The ability to monitor application access to OS functions like network requests, bluetooth, contacts etc

☐ **4.11. AirDrop (iOS and Android Equivalent)**
Although not covered by browsers or Web Apps, OWA believes consumers and competition would benefit from an **interoperable standard** for Apple's **AirDrop,** and its Android equivalents.

☐ **4.12.AirPlay (iOS and Android Equivalent)**
As above but for Apple's **AirPlay** and its Android equivalents..

☐ **4.14. Private Relay**
Private Relay must apply equally to traffic from Native Apps as it does to installed Web Apps. OWA's strong preference would be that all Native Apps also must run through Private Relay, rather than removing Private Relay functionality from installed Web Apps. Note that while Private Relay only applies to Web Apps and Browser Traffic it will push developers to Native Apps as it enables them to circumvent the privacy protections that Private Relay offers namely cross-site tracking.

☐ **4.15 Direct Download via App Stores**
Third party browsers which include their engines must be directly downloadable via Apple's AppStore

☐ **4.16 Authentication / Multi-Factor Authentication / Keychain / Password Management**

The system for storing and managing passwords / passkeys, tokens across all apps must be swappable with equivalent integration for a competitor's product. Apple's password/passkey management ecosystem is only available on Apple devices and so will not work on Windows or Android devices.

- ☐ **5. App Store Support for Web Apps (iOS / Android)**

  - ☐ **5.1 External Web Based Service**
    The app store should accept Web App Submission via a Public Web Based Service with a goal of ensuring specific hardware or operating systems are not required and to enable automated deployment tools that can submit to multiple app stores from different providers reducing the effort and cost for developers.

  - ☐ **5.2 No Specific Operating System or Hardware Required**
    Developers should not be required to use any specific operating system to submit the Web App to the app stores.

    - ☐ No Mac required.
    - ☐ No XCode required.

  - ☐ **5.3 Minimal Steps**
    The steps and complexity of submitting a Web App are reduced to the minimum required. Note that signing up for a developer account on an app store can be a separate process.

- ☐ **6.Safari Core Functionality (iOS)**
  - ☐ **6.1 Install Prompts (Installability)**
    - ☐ **6.1.1 Minimum Bar is Equivalent to Native**
      The install procedure for Web Apps is at least as easy and requires equal or less steps as the procedure for installing Native.

    - ☐ **6.1.2.No Regression in Installation Speed**
      Any changes should not noticeably increase the speed it takes to install a Web App

    - ☐ **6.1.3 In-App Browser Installation**
      All In-App Browsers must support easy and seamless installation of Web Apps

    - ☐ **6.1.4 Developer Control over Native/Web Install Prompt**
      When neither a Web App or Native App is installed, the developer should be able to indicate to the browser whether or not they wish to display an install banner.

      Developers should have explicit control over when/if to display a native app banner/prompt OR a Web App banner/prompt

- ☐ **6.2 Notifications**
  (Same Requirements as 2.2)

- ☐ **6.3. Fullscreen**
  <canvas>For <canvas> and other non-video content.

- ☐ **6.4. Badging**

- ☐ **6.5. Deep Links**

- ☐ **6.6. Screen Orientation Lock**

- ☐ **6.7. Web Payment APIs**
  Full support for the various Web Payment APIs along with third-party payment handlers that can register via a Web App or a Native App

☐ **7.No Chrome Preferencing (Android)**

- ☐ **7.1 "Google Search App" Must Respect Default Browser.**

- ☐ **7.2 Mobile Application Distribution Agreement (MADA) should not require Original Equipment Manufacturers (OEMs) to prefer Chrome**

☐ **8. Website Transparency Obligations**

- ☐ **8.1 Gatekeepers must have a compat.txt with detailed explanation for browsers with a market share of greater than 2% that the website does not support including:**

  - ☐ **Bugs**
    With links to bug tickets

  - ☐ **Functionality**
    With links to missing functionality

- ☐ **8.2. "Try Anyway"** button must always be available

- ☐ **8.3. "Switch to a browser"** must describe the benefits to the user including in terms of speed, performance or functionality

- ☐ **8.4. Consider applying all large companies**

9. Web Standards

Web Browsers implement designs that are developed and specified in many Standards Development Organisations ("SDOs") including, but not limited to:

- IETF (The Internet Engineering Task Force)
- W3C (The World Wide Web Consortium)
- WhatWG (Web Hypertext Application Technology Working Group)
- ECMA(The European Computer Manufacturer's Association)
- The FIDO Alliance
- AOM (the Alliance for Open Media)
- The Khronos Group
- ISO (The International Organization for Standardization)
- The Unicode Consortium
- IANA (the Internet Assigned Numbers Authority)

Each of these organisations contain a part of the "web standards community", but feature differing processes, publication gates, and formal mechanisms for including designs into official standards documents. Critically, all of these SDOs create *voluntary standards*. Companies contribute their intellectual property to the commons of voluntary web standards for complex reasons, but competition has historically driven this process.

The existing patchwork of standards groups, along with the social and legal background of voluntary standards, makes proposals to bar browsers from implementing features ahead of formal standardisation deeply problematic.

Our analysis of the situation regarding browsers suggests that users and developers do not suffer from too much divergence of views about how to solve leading-edge problems, but rather a lack of engagement and investment in addressing those challenges. We believe this lack of investment stems from reduced competition that browser makers experience on iOS (and to a lesser extent, Android).

In a healthy environment, Web Standards evolve quickly, spurred on by competing browser makers working with developers to solve important problems. This involves collaboration in standards bodies to improve compatibility, however if each vendor had to wait until there was consensus among every vendor regarding every design, it would be possible for a vendor (e.g. Apple) to game these processes. There is also significant risk that well-funded third-parties could infiltrate standards organisations in order to block/stall development or functionality in a manner that is not in the best interests of the user and/or competition.

Browser vendors enjoy outsized influence in development of web standards, and providing them with a veto over all progress will only serve to reward the slowest mover by preventing competitors from taking market share. In the *existing* structure, it is enough for a vendor to withhold engagement to prevent functionality from being standardised.

Typically, cutting edge features are deployed by browser makers in their own engines first, then, using real world feedback over several years, eventual standards are created. No feature starts out as a web standard:

> *"Web Standards are voluntary. The force that most powerfully compels their adoption is competition, rather than regulation. This is an inherent property of modern browsers. Vendors participate in standards processes not because they need anyone else to tell them what to do, and not because they are somehow subject to the dictates of standards bodies, but rather to learn from developers and find agreement with competitors in a problem space where compatibility returns outsized gains"*

Alex Russell - Microsoft Edge

No one can predict what web technologies will be important in the future, and disagreements between browser makers on the exact path forward are reasonable and expected. It is very difficult, if not impossible for regulators to predict which standards will be the most important and what their exact definition will end up being. It's a subtle and complex topic, and one that would require significant staffing, over a wide swath of technical areas, for any regulator to credibly participate in.

The suggestion that individual browser vendors or groups of browser vendors should be able to block all other browsers from producing functionality (that these browsers have no intention of implementing) would further stifle competition.

Requiring consensus before implementation would block the ability of users to signal their discontent by switching browsers to ones that offer the functionality they need. It would also remove any pressure browser vendors might feel to implement popular features if they can block all competitors from providing it. Subtly, it would also reduce the quality of features delivered, because it is only through market-oriented mechanisms like Origin Trials that leading engine teams ensure their designs are fit for purpose.

Apple has held back the Web on iOS (and mobile in general, thanks to network effects) for more than a decade via their veto on features for all browsers on iOS. Handing Apple explicit power to veto features for all browsers would be a disaster. One of the key aims of our remedies is to break Apple's ability to prevent iOS users from accessing useful Web functionality that competes with either their own Apps/Services or their App Store.

Browsers (and their engines) need to be free to lead and experiment on the competitive frontier. They need the freedom to be wrong but also to win users through unique features and leading edge capabilities. It is competition, developers and user choice that determines which features will be successful. Standiziation between similar features in browsers is desirable, but it is an outcome that comes at the end of the development and competitive process. It should not be placed as a gate at the beginning. Further, the potential for blocking browsers ability to differentiate themselves and compete would be catastrophic to competition. Our primary concern is not that Apple will have a different vision for how particular features will function, but rather that Apple has already sought to delay features that allow the web to compete with its own, proprietary, Native Apps platform. By trying to ensure that competing features are not available, Apple removes pricing pressure on its Native App ecosystem and extends power over developers.

Rather than allow the excellent work of the W3C and other standards organisations to be weaponized to block innovation and competition, a better approach is regulating to **enable effective competition** on the gatekeepers operating systems both between rival browsers and between Web Apps and Native Apps. Then allow market forces to push forward the changes (new web features) most beneficial to end users.

Both developers and users gain significant advantages from interoperable implementations of browser functionality, however it's essential that any interventions ensure that the competitive aspects of browsers are protected including the pressure to significantly invest in new and cutting edge technologies.

Standards should be used to:

1. Increase interoperability between Browsers
2. To allow multiple browser vendors from different companies an ability to be meaningfully involved and provide constructive feedback including but not limited to privacy and security concerns.
3. Provide implementation documentation for Browser Vendors

Standards should not be used to:

1. Reduce or block the level of investment OR pressure on vendors to invest
2. Reduce or block implementation of cutting edge functionality
3. Designate functionality as being exclusive to native (proprietary) apps
4. Enable third-parties to undermine features or functionality required by developers or users (i.e. Telecommunication or Advertising Companies attempting to undermine privacy rules)

10. Brighter Future for Mobile Ecosystems

We would like to thank the CMA again for considering these important issues.

Mobile ecosystem app stores wield a lot of power and lock-in over consumers. This allows them to both block competitors and charge consumers additional fees to install Apps from competitors.

Web Apps could be an interoperable source of competition for app stores however they are not currently viable on iOS for a number of reasons we have outlined in our answers and our paper.

We have proposed a number of remedies that we believe are vital to restore competition and user choice to the mobile ecosystem.

We believe that gatekeepers of mobile hardware/operating systems should compete to offer additional services and software to customers on merit and user choice, not by blocking applications that compete with their own or by charging consumers additional fees on applications/services from competitors.

The CMA has a chance to blunt the anti-competitive power of incumbent gatekeepers and create a level and competitive playing field for the future of mobile App development.

These changes will lead to the following benefits to consumers:

- More competition
- Interoperable Apps
- Higher quality Apps
- Lower cost Apps
- Less lock-in into particular mobile operating systems

Competition not walled gardens leads to the best outcomes for consumers

11. References

Definition of Web Apps
https://developer.mozilla.org/en-US/docs/Web/Progressive_web_apps

W3C patent policy
https://www.w3.org/Consortium/Patent-Policy-20200915/

Commercial users now spend greater than 60% of their time within a browser
https://www.microsoft.com/en-us/microsoft-365/blog/2020/08/04/revisit-idea-modern-browser/

Apple claim Web Apps are a viable alternative to the App Store
https://www.accc.gov.au/system/files/Apple%20Pty%20Limited%20%2810%20February%202021%29.pdf

Tim Cook making a similar claim to congress
https://www.youtube.com/watch?v=H6eYLCxxQdA&t=306s

Apple's lawyers making a similar claim in a court case in Australia
https://9to5mac.com/2021/03/25/bypass-the-app-store-says-apple/

51% of mobile phones in the UK use iOS
https://gs.statcounter.com/os-market-share/mobile/united-kingdom

65% of web traffic in Japan is from iOS
https://gs.statcounter.com/os-market-share/mobile/japan

The latest revision of the EU's Digital Markets Acts
https://www.consilium.europa.eu/media/56086/st08722-xx22.pdf

UKs Competition and Market Authority - Mobile Ecosystems Market Study Interim Report
https://www.gov.uk/government/publications/mobile-ecosystems-market-study-interim-report/interim-report

Apple collected $72.3 billion USD in App Store fees in 2020
https://appleinsider.com/articles/21/01/05/app-store-earns-723-billion-in-2020-almost-double-google-play-revenues

iOS App Store has an estimated nearly 80% profit margin
https://www.marketwatch.com/story/how-profitable-is-apples-app-store-even-a-landmark-antitrust-trial-couldnt-tell-us-11622224506

Industries with healthy competition feature leading firms with profit margins between 5 and 20 percent
https://www.brex.com/blog/what-is-a-good-profit-margin

iOS App Store Guidelines
https://developer.apple.com/app-store/review/guidelines/

FossPatents on Apple's Third Party Browser Engine Ban
http://www.fosspatents.com/2022/06/stats-suggest-apples-browser-engine.html#:~:text=Apple%2D%2Dand%20all%20sorts%20of%20people%20beholden%20to%20it%2D%2Dstress%20that%20it%27s%20about%20national%20security

iOS - Intro to mobile device management
https://support.apple.com/en-gb/guide/deployment/depc0aadd3fe/web

Discussion of iOS Webkit Security
https://infrequently.org/2021/08/webkit-ios-deep-dive/#:~:text=Some%20engines%20go,other%20than%20Safari.

Chromium's Site Isolation
https://www.chromium.org/Home/chromium-security/site-isolation/#current-status

Microsoft Edge's Enhanced Security Mode
https://microsoftedge.github.io/edgevr/posts/Introducing-Enhanced-Security-for-Microsoft-Edge/

iOS - Lockdown Mode
https://www.pocket-lint.com/phones/news/apple/161776-what-is-apple-lockdown-mode-and-how-to-turn-it-on-iphone-ipad-mac

iOS - Rapid Security Response
https://www.engadget.com/apple-security-fixes-ios-16-rapid-security-response-082551143.html

Microsoft Edge allowed disabling JIT compilation
https://www.bleepingcomputer.com/news/microsoft/microsoft-edge-adds-super-duper-secure-mode-to-stable-channel/

US Executive Order 14028
https://www.federalregister.gov/documents/2021/05/17/2021-10460/improving-the-nations-cybersecurity

NIST - Definition of critical software
https://www.nist.gov/system/files/documents/2021/10/13/EO%20Critical%20FINAL.pdf

NIST - Software development security requirements
https://www.nist.gov/itl/executive-order-improving-nations-cybersecurity/security-measures-eo-critical-software-use

NIST - Minimum Standards
https://nvlpubs.nist.gov/nistpubs/ir/2021/NIST.IR.8397.pdf

Discussion of definition of a browser
https://infrequently.org/2021/07/hobsons-browser/#starting-points

iOS - Browser Entitlement
https://developer.apple.com/documentation/bundleresources/entitlements

J's Entitlement Database
http://newosxbook.com/ent.jl?osVer=iOS15.2

IETF (The Internet Engineering Task Force)
https://www.ietf.org/

W3C (The World Wide Web Consortium)
https://www.w3.org/

WhatWG (Web Hypertext Application Technology Working Group)
https://whatwg.org/

ECMA(The European Computer Manufacturer's Association)
https://www.ecma-international.org/

The FIDO Alliance
https://fidoalliance.org/

AOM (the Alliance for Open Media)
https://aomedia.org/

The Khronos Group
https://www.khronos.org/

ISO (The International Organization for Standardization)
https://www.iso.org/home.html

The Unicode Consortium
https://home.unicode.org/

IANA (the Internet Assigned Numbers Authority)
https://www.iana.org/

Alex Russell - On Web Standards
https://infrequently.org/2020/07/why-ui-isnt-specified/

Origin Trials
https://developer.chrome.com/docs/web-platform/origin-trials/

Google Search App - Undermining Browser Choice
https://infrequently.org/2021/07/hobsons-browser/

12. Open Web Advocacy

Open Web Advocacy is a loose group of software engineers from all over the world, who work for many different companies who have come together to fight for the future of the open web by providing regulators, legislators and policy makers the intricate technical details that they need to understand the major anti-competitive issues in our industry and potential ways to solve them.

It should be noted that all the authors and reviewers of this document are software engineers and not economists, lawyers or regulatory experts. The aim is to explain the current situation, outline the specific problems, how this affects consumers and suggest potential regulatory remedies.

This is a grassroots effort by software engineers as individuals and not on behalf of their employers or any of the browser vendors.

We are available to regulators, legislators and policy makers for presentations/Q&A and we can provide expert technical analysis on topics in this area.

For those who would like to help or join us in fighting for a free and open future for the web, please contact us at:

Email	contactus@open-web-advocacy.org
Twitter	@OpenWebAdvocacy
Web	https://open-web-advocacy.org