# Progressive Webapp

**Definition:**

Any application that delivers app-like experience in the browser and can be developed progressively (Can be developed in layers that are independent of each other)

**Characteristics:**

1. Instant loading 7. Linkable
2. Discoverable 8. Re-engageable(Push notification)
3. N/W independent 9. Works everywhere
4. Responsive 10. Fast
5. Installable
6. Secure

**Progressive Enhancement: Adding features incrementally in layers.**

1. It’s a website
2. Adds native installation
3. Adds Web push notifications
4. Add hardware & platform access

**PWAs and Hybrids:**

1. Big differences
2. Store distribution – Need to pay for Hybrid apps and need to package them
3. No packaging and signing for PWAs
4. No native plugins for PWA. This is where Hybrids take the lead**.**

**PWAs compatibility:** <http://gs.statcounter.com/>(Site for getting stats for browser usage)

1. Chrome on Android
2. Opera Mobile
3. Samsung Internet browser (Chromium based)
4. Firefox developer edition
5. Edge (future, IE +16)
6. Limited APIs available for iOS, so limited support. iOS 11 may be better performing in PWAs
7. Chrome iOS is not chrome. Check user-agent string for chrome.
8. Each instance of the PWA depends on the browser engine it has been installed. Eg: If we installed the same app from 2 different browsers, then we would have 2 separate instances of the app running independently. This is a problem to be addressed.
9. We could save the credentials of the user if there is only one engine that the app is being installed. Currently this is supported only in Android. But there is a credentials management API in the pipeline that helps the OS manage the user session.
10. PWA is ideal for the apps that are single page applications.
11. PWA supports web-sockets as the browsers supporting the PWAs do that.
12. PWA is great asset to the Physical web. (https://google.github.io/physical-web/)

**Limitations:**

1. Multiplatform support, eg: camera might different in different platform.
2. Responsive design – Taking into account network, UI rendering and other cases for better design.
3. PWAs do not get the privileges of First class citizen like the native apps.
4. Intents
5. Distribution, PWAs are not present in the store.
6. Social networking sites cannot be acting as a discovery mechanism for PWAs. The reason is that a lot of the time in the mobile devices the social networking apps open their own internal browser. These do not support PWAs.
7. Web app installers also could be ineffective as the user has to visit the website twice for getting the PWA install option.

Web App manifest: Eg: <https://www.flipkart.com/manifest.json> (Change to device mode in chrome debug tool)

* Should be based on W3C spec.
* It is a JSON file.
* Should be served as of type application/manifest.json
* name: The name shown on the splash screen.
* Start\_name: The name shown under the icon on the mobile home page.
* Start\_url : The url from where the app should or opened.
* Orientation : any, portrait [-primary | -secondary], landscape [-primary | -secondary], natural(Lets the device decide what is best suitable.)
* Display:
  + Browser : Not actually PWA
  + minimal-ui : Lite version of the app. Browser UI but with little options to edit the options say URL etc. Currently similar as standalone
  + **standalone**: Most web-apps use this for mobile app experience.
  + fullscreen: Mostly games where we miss the battery status, top bar of phone.
* background\_color:
* theme\_color: The color of the theme for the app.
* Icon: array of the icons with props
  + Src
  + Sizes: size of the icons for diff platforms.
  + type

Notes for iOS:

* <meta name="apple-mobile-web-app-capable" content="yes"> add this for iOS devices.
* In iOS the PWAs are not safari processes. They are stored as process with extension “app”. iOS 11 would have better support.
* <meta name="apple-mobile-web-app-title" content="Fluent"> to add a name for the icon.
* <meta name="apple-mobile-web-app-status-bar-style" content="black-translucent"> along with the header tag and css styles can be used to emulate the theme property of manifest.json file in iOS
* In iOS the PWAs are reloaded every time we open the app.
* <link rel="apple-touch-icon" href="icon\_ios.png"> to add the icon for PWA in iOS devices like iPhone and iPad.