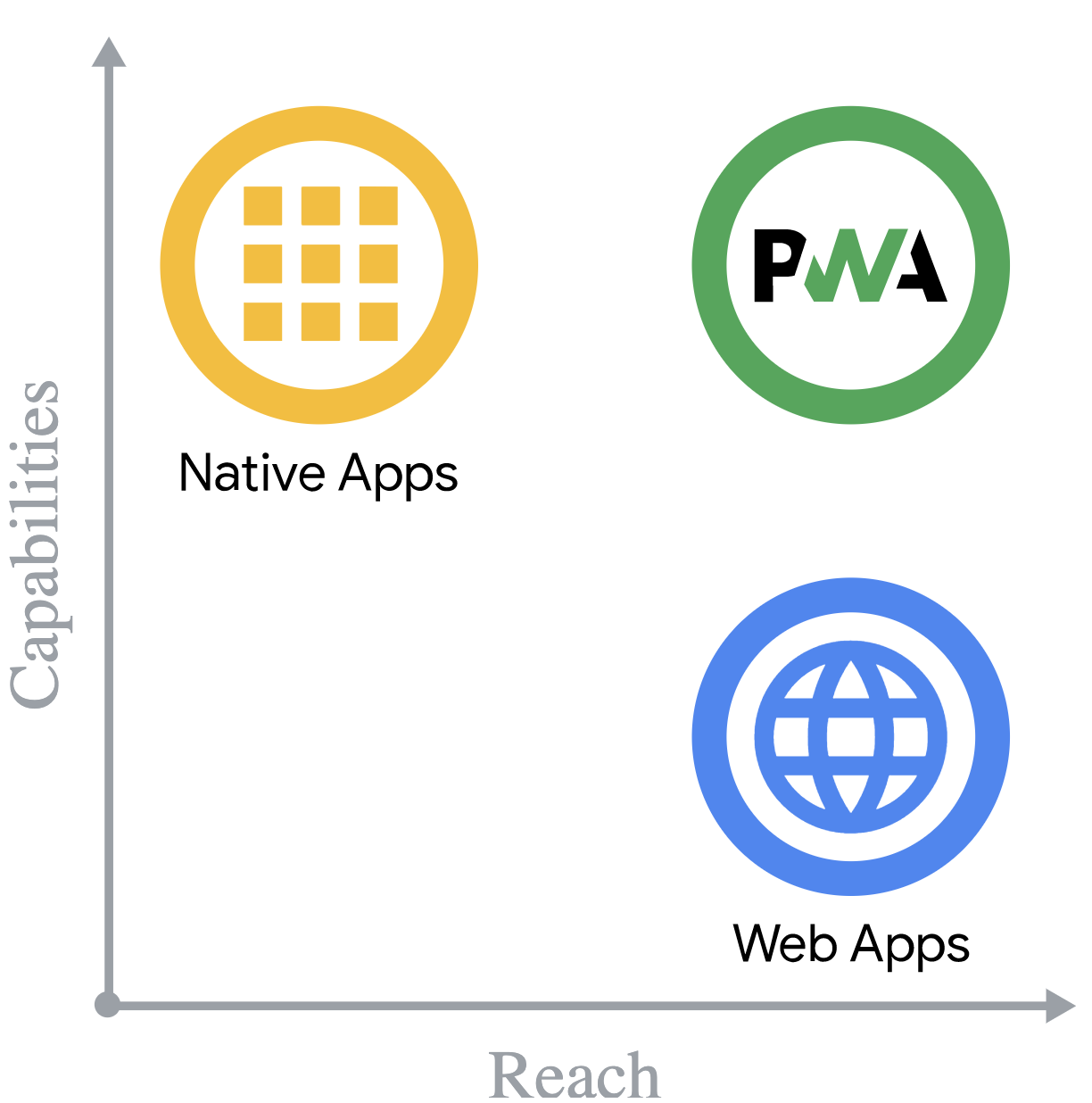
## Wat is een PWA

Het web is gegroeid tot een platform waar applicaties kunnen gepubliceerd worden zonder afhankelijk te zijn van een overkoepelend bedrijf. Voor een website is er slecht 1 codebase en telkens en de laatste versie is steeds beschikbaar voor de gebruiker.   
Dit allemaal zorgt ervoor dat een webapplicatie iedereen, overal kan bereiken en dit op elk mogelijk toestel.

Native applicaties zijn gekend om betrouwbaar te zijn en een heel goede gebruikers ervaring te kunnen beiden. Ze starten op als een alleenstaande ervaring en ze kunnen uitgebreid gebruik maken van het besturingssysteem: ze kunnen bestanden lezen en schrijven, gebruik maken van usb-connecties en bluetooth, ze hebben toegang tot de contacten en de kalender en nog veel meer. Native applicaties voelen aan alsof ze deel uitmaken van het toestel waarop ze werken.

We kunnen dus concluderen dat webapplicatie de bovenhand hebben in bereik maar dat native applicaties de bovenhand hebben als op functies aankomt.

Een Progressive web application (PWA) is een webapplicatie die gebruik maakt van moderne web API’s om een functies aan te bieden die voordien enkel beschikbaar waren voor native applicaties. De bedoeling van PWA’s is om de sterktes van webapplicaties (het bereik) en native applicaties te combineren (de functionaliteit).

­­ 

Volgens Google Developers zijn de drie pijlers van PWA’s dan ook

* Functionaliteit (capable)
* Betrouwbaarheid
* Installeerbaarheid

Deze aspecten worden in dit hoofdstuk verder uitgelegd.

<https://web.dev/what-are-pwas/> – 6 maart - 2020

### Service workers

#### Wat is een service worker

De service worker is een script dat veel functionaliteiten beschikbaar maakt die voordien enkel beschikbaar waren voor native applicatie. In dit hoofdstuk wordt er bekeken naar welke functionaliteiten de service worker juist beschikbaar maakt en hoe dit gebeurt.

Een service worker is een web worker die tussen het netwerk en de applicatie wordt geplaatst. Deze service worker wordt tussen de backend en de frontend geplaatst. Dit zorgt ervoor dat de service worker inkomende en uitgaande netwerkverzoeken kan controleren en eventueel manipuleren.

<https://developer.mozilla.org/en-US/docs/Web/Progressive_web_apps/Offline_Service_workers> - 6 Maart - 2020

Een web worker is een script dat in de achtergrond van een applicatie werkt en die onafhankelijk is van de andere scripts. Web workers hebben dus geen impact op de prestaties van op de webapplicatie die er gebruik van maakt. Web workers hebben geen toegang tot de DOM van een webapplicatie, ze kunnen de inhoud van een website dus niet rechtsreeks manipuleren.

<https://upcommons.upc.edu/bitstream/handle/2117/99263/Web_Workers-Camera_Ready-1.pdf>

(hiltunen, 2018)

//nog iets schrijven over volgend artikel

<https://www.researchgate.net/profile/Ivano_Malavolta/publication/318326808_Assessing_the_Impact_of_Service_Workers_on_the_Energy_Efficiency_of_Progressive_Web_Apps/links/59d7404a0f7e9b42a6b069fb/Assessing-the-Impact-of-Service-Workers-on-the-Energy-Efficiency-of-Progressive-Web-Apps.pdf>

#### Functionaliteiten die een serivce worker mogelijk maakt

De service worker werkt onafhankelijk van de applicatie. Dit houdt in dat een service worker wel nog kan werken terwijl de applicatie afgesloten is, hierdoor zijn volgende functies mogelijk binnen een webapplicatie:

**Offline gebruik**

De service worker kan netwerk verzoeken onderscheppen, als er een netwerk verzoek wordt verstuurd en er is geen internetverbinding zal de service worker de client antwoorden met een gepaste boodschap. Zonder service worker zou het systeem crashen.

Met service workers kunnen bestanden ook gecacht worden. Als een pagina geladen wordt kunnen alle elementen opgeslagen worden op het toestel. Als deze pagina later opnieuw bezocht wordt hoeft deze niet meer aan de server gevraagd te worden. Hierdoor wordt de applicatie sneller en minder afhankelijk van de netwerkverbinding.

De twee mechanismes die gebruikt worden om data offline beschikbaar te maken zijn ‘indexedDB’ en de ‘cache API’.

<https://developers.google.com/web/fundamentals/instant-and-offline/web-storage/offline-for-pwa> - 6 Maart - 2020

<https://developer.mozilla.org/en-US/docs/Web/API/Service_Worker_API/Using_Service_Workers> - 6 Maart - 2020

De cache API wordt gebruikt om data die verkregen werd van netwerkverzoeken op te slaan. Zowel de ‘request’ als de ‘response’ van een netwerkverzoek kunnen in de cache API opgeslage worden

<https://developers.google.com/web/fundamentals/instant-and-offline/web-storage/cache-api> - 6 Maart - 2020

IndexedDB is een mechanisme dat gebruikt wordt om lokaal gestructureerde data op te slaan. Het kan vergeleken worden met traditionele relationele databasemanagementsystemen, er wordt echter geen gebruik gemaakt van kolommen maar van javascript objecten.   
Een IndexedDb maakt gebruik van indexen, dit heeft als voordeel dat het uitlezen van data heel snel gaat.

<https://developer.mozilla.org/en-US/docs/Web/API/IndexedDB_API> - 6 Maart - 2020

**Push notifications**

Er zijn twee soorten notificaties lokale notificaties en push notificaties.   
Het verschil tussen beide is dat lokale applicaties geactiveerd worden vanuit de code van de applicatie, deze notificaties kunnen dus enkel geactiveerd worden als de gebruiker de applicatie aan het gebruiken is, lokale notificaties hebben dus niet zoveel use-cases. Dit is mogelijk zonder service worker.   
Push notificaties worden niet geactiveerd binnen de applicatie maar worden geactiveerd door een server.

Om push notificaties te gebruiken moet er gebruikt gemaakt worden van twee webAPI’s.

* Notifications API

Dit is een API die het uiterlijk en het gedrag van een notificatie zal bepalen. Deze API wordt zowel gebruikt voor lokale als voor push notifications.  
Om gebruik te maken van deze API moet de gebruiker expliciet toegang geven aan de applicatie.

Een voorbeeld van een notificatie kan er als volgend uitzien

function displayNotification() {  
  if (Notification.permission == 'granted') {  
    navigator.serviceWorker.getRegistration().then(function(reg) {  
      var options = {  
        body: 'Here is a notification body!',  
        icon: 'images/example.png',  
        vibrate: [100, 50, 100],  
        data: {  
          userId: “383209489398274”  
        },  
        actions: [  
          {action: 'explore', title: 'Explore this new world',  
            icon: 'images/checkmark.png'},  
          {action: 'close', title: 'Close notification',  
            icon: 'images/xmark.png'},  
        ]  
      };  
      reg.showNotification('Hello world!', options);  
    });  
  }  
}

Om een notificatie weer te geven wordt er een object verwacht waar de inhoud van de melding wordt vastgelegd. Volgende keys kunnen meegegeven worden:

|  |  |
| --- | --- |
| Body | De boodschap die in de melding staat komen. |
| Icon | Het icoontje dat in de notificatie wordt getoond |
| Vibrate | Het vibratiepatroon dat melding zal maken in miliseconden |
| Data | Data is een object dat gebruikt kan worden als de gebruiker om de notificatie klikt. Dit object zal dan ontvangen worden in de applicatie. Hier zal vaak het id van de gebruiker teruggevonden worden. |
| Actions | Er kunnen ook acties toegevoegd worden aan de melding. Elk object in deze array zal een knop worden op de melding met een andere functie. Het gedrag van de knoppen wordt bepaald in de applicatie aan de hand van de ‘ation’ |

<https://developers.google.com/web/ilt/pwa/introduction-to-push-notifications> - 6 Maart - 2020

<https://developer.mozilla.org/en-US/docs/Web/API/Notifications_API> - 6 Maart - 2020

* Push API

De push API wordt gebruikt door de service worker. Als de server een notificatie verstuurd wordt deze opgevangen door deze push API. Deze push API zal dan gebruik maken van de notifications API om een melding op het toestel van de eindgebruiker te tonen.

<https://developer.mozilla.org/en-US/docs/Web/API/Push_API> - 6 Maart - 2020

<https://web-push-book.gauntface.com/>- 6 Maart - 2020

**Background synchronisatie**

Een PWA kan gebruik maken van de background sync API om achtergrondsynchronisatie toe te passen.

Achtergrondsynchronisatie kan toegepast worden als een trage of geen netwerkverbinding is.

Achtergrondsynchronisatie is het proces waarbij een netwerkverzoek, dat uitgevoerd werd als er geen of een te zwakke internetverbinding is, wordt opgeslagen in de service worker en wordt uitgevoerd als er wel een internetconnectie is.

Een voorbeeld hiervan is het verzenden van een bericht via een sociaal media platform. Als het bericht verzonden wordt als de gebruiker offline is, zal er geen fout getoond worden maar zal dit bericht verzenden vanaf er internet is.

Ook google chrome op android maakt hier gebruikt van. Als er een website bezocht wordt als er geen internetverbinding is krijgt de gebruiker de melding ‘chrome laat je weten wanneer de pagina klaar is’. Vanaf het toestel de pagina heeft kunnen downloaden krijgt de gebruiker een melding dat de pagina nu bekeken kan worden.

**A screenshot of a cell phone

Description automatically generatedA screen shot of a computer

Description automatically generated**

<https://developers.google.com/web/updates/2015/12/background-sync>

**Performantie**

// to doooo

#### Service worker lifecycle

Om een service worker te installeren moet deze geregistreerd worden in de javascript van het de webapplicatie. Als een service worker wordt geïnstalleerd worden de opgegeven statische bestanden (foto’s, css bestanden, javascript bestanden,) gedownload. Als dit slaagt, is de service worker geactiveerd, als dit niet slaagt zal dit proces zich herhalen tot het slaagt.

Tijdens de activatiefase wordt er bekeken welke gecachte gegevens upgedate moeten worden en welke niet.

Als het oude cach geheugen is upgedate waar nodig zal de service worker overgaan naar een ‘rust’ toestand, hierbij wacht de service worker op netwerk verzoeken van bestanden die binnen zijn scope vallen.

Als er een netwerk verzoek wordt verstuurd zal de service worker deze verzoeken afhandelen. Na een bepaalde tijd zal de service worker worden afgesloten om geheugen te sparen.

In beide gevallen zal de service worker terug naar de rust staat gaan om nieuwe verzoeken af te handelen.

A picture containing text

Description automatically generated

<https://developers.google.com/web/fundamentals/primers/service-workers>

### A2HS

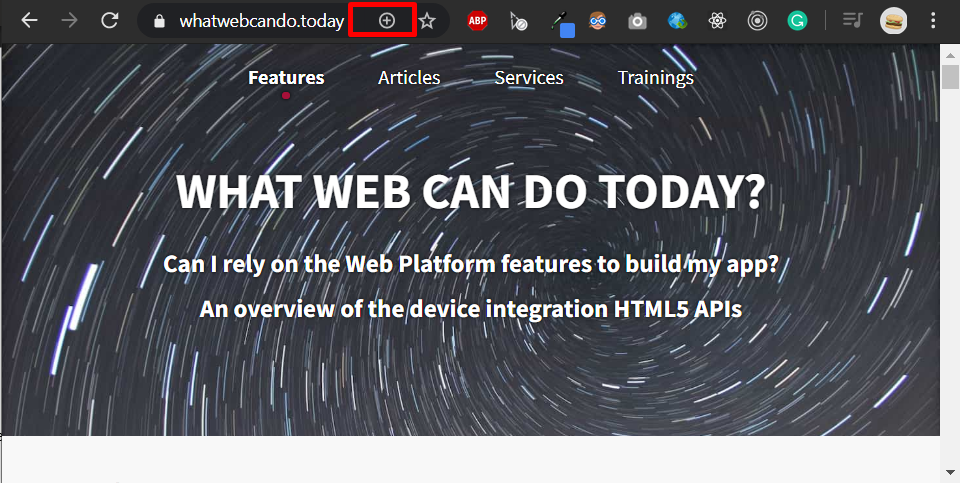
Als een applicatie voldoet aan bepaalde criteria kan deze geïnstalleerd worden op het toestel van de gebruiker. Deze functie is beschikbaar voor op verschillende besturingssystemen: Windows, Mac OS, Android, IOS.

Een website moet voldoen aan volgende criteria

* Nog niet geïnstalleerd zijn
* Een HTTPS connectie hebben
* Een manifest.json bestand hebben
* Een service worker registreren.

Als een website aan alle criteria voldoet zal er een beforeinstallprompt event gestart worden. Elke browser gaat hier anders met om.

Google chrome desktop:

A screenshot of a cell phone

Description automatically generated

Op Apple toestellen (iphone, Mac) heeft het beforeinstallprompt geen effect. De gerbuiker moet zelf op zoek gaan in het menu om de applicatie te installeren, maar dit is wel mogelijk.

Het beforeinstallprompt kan wel in de code opgevangen worden, De applicatie kan dan de gebruiker informeren dat de applicatie geïnstalleerd kan worden en hoe hij dit moet doen.

### Application shell

// to do

### Progressive enhancement

Dit is een principe waarbij een web applicatie gebouwd moet worden die op elke browser (ook verouderd) moet werken. Deze basis website kan dan uitgebreid worden met functies die ondersteund worden door moderne browsers. Maar de website mag niet afhankelijk zijn van deze moderne functies.

Als dit principe succesvol wordt geïmplementeerd dan is de website op alle platformen bruikbaar en is het bereik van de applicatie maximaal.

<https://aaltodoc.aalto.fi/bitstream/handle/123456789/29096/master_Vanhala_Janne_2017.pdf?sequence=1&isAllowed=y>

### App manifests

het web app manifests is een JSON bestand dat informatie bevat over de applicatie. Deze informatie is nodig voor het installeren van een PWA op een toestel.

Voorbeeld van een minimum app manifest voor de google maps PWA.



|  |  |
| --- | --- |
| short\_name | Naam van de applicatie die op het startscherm gebruikt zal worden. Deze mag maximaal 12 karakters lang zijn. |
| Name | Naam die op alle andere plekken gebruikt zal worden: vb bij de vraag als de app geïnstalleerd mag worden. Deze mag maximaal 45 karakters lang zij. |
| Icons | Een object die het icoon bepaald dat de applicatie zal gebruiken.  Dit object heeft volgende eigenschappen   * Src * Type * size |
| Start\_url | De url naar waar de PWA moet gaan als de applicatie gestart wordt vanaf het startscherm van een toestel |
| Background\_color | Hier wordt een kleur gedefinieerd, dit kleur zal gebruikt worden voor het opstart scherm |
| Display | Bepaald in wat voor webview de PWA getoond zal worden. Mogelijkheden zijn:   * fullscreen – opent de browser zonder UI-elementen (adresbalk, terug knop, ….) * standalone – opent de applicatie als een native applicatie los van de browser. Er worden geen UI elementen van de browser getoond. * minimal-ui – open de applicatie in de browser maar toont slecht beperkte ui elementen van de browser. De adresbalk is weg maar te terugknop is er nog. * Browser – opent de PWA in een normaal browser tapblad. |
| Scope | De scope bepaalt alle links die binnen de PWA vallen. |
| Theme\_color | De kleur die de adresbalk zal innemen. |