|  |
| --- |
| Karel de Grote Hogeschool |
| Vergelijkende studie Xamarin - Phonegap |
| Web Research 3 |

|  |
| --- |
| Seppe Goossens  15-11-2016 |

# Inleiding

Het doel van het vak Web Research in de eerste periode is het zelfstandig leren van een nieuwe technologie, in zijn ruimste interpretatie, en het vergelijken van twee technologieën. Aangezien ik als nieuwe technologie Xamarin neem, een framework dat gebruikt wordt om mobile apps te maken, zal ik het ook gebruiken om de vergelijkende studie mee uit te voeren. Er zijn momenteel heel wat verschillende mogelijkheden om mobile apps te ontwikkelen, gaande van HTML5, over Phonegap, Cordova en varianten, langs Xamarin tot native, met elk hun voor en nadelen. Hieruit kies ik Phonegap, een framework waar de voorbije jaren al gewerkt is in de les, om deze vergelijking aan te vullen.

Welk van beide is het best voor het ontwikkelen van apps, is de vraag die gesteld wordt. Aan de hand van grote voordelen van het ene framework over het andere, en specifieke doelen van beide zal hier een antwoord op gegeven worden, alsook een suggestie van situaties waarin respectievelijk het ene of de andere aangeraden wordt.

Eerst worden kort beide omschreven met hun specifieke eigenschappen en de manier waarom zij appontwikkeling aanpakken, waarna enkele van hun voornaamste voor- en nadelen op een rijtje gezet worden. Tot slot worden de pro’s en con’s van beide naast elkaar gezet waaruit een conclusie getrokken zal worden.

# Beide aan het woord

## Phonegap

### Korte omschrijving

Elk platform heeft z’n eigen specificaties, vereisten, frameworks en taal, maar iedereen wil graag op zo veel mogelijk te bereiken zijn, anders mis je een groot deel van je doelgroep, welke deze ook mag zijn. Een app maken voor allemaal is dus een tijdrovende en/of kostelijke zaak. Wat meestal wel al gegeven is, is een website. Hier komt Phonegap in het spel. Wat als die website, al dan niet mits kleine aanpassingen, nu gebruikt kon worden als app?

Phonegap zet applicaties, geschreven in HTML om zodat ze als native applicaties gebruikt kunnen worden. Om dit te bekomen wordt de app in een native shell gestoken waardoor hij gewoon herkent wordt door de platformen als native. Verder zorgt Phonegap er ook voor dat je kunt werken met de native API, welke toegang geeft tot de camera, contactpersonen, etc. doormiddel van plugins. Hier kan echter niet altijd vanuit gegaan worden, want niet alle plugins bestaan voor alle platformen, en niet alle platformen hebben dezelfde functionaliteiten.

HTML, CSS en Javascript worden geschreven zoals voor een website, rekening houdende met de grootte van mobile schermen, wat de leercurve voor de meeste developers zeer laag houdt. Ook is er een zeer grote community, wat problemen en errors makkelijker en sneller verhelpt. Ook zorgt het schrijven van de app zelf in HTML, CSS en Javascript ervoor dat je heel die community ook kan gebruiken, voor vragen of problemen op dat vlak. Er zijn echter geen voorgemaakte UI widgets, transitions, standaard knoppen, etc., wat het ontwikkelingsproces van de app aanzienlijk kan verlengen, als deze een native look and feel moet hebben. Er bestaan wel tools met voorgemaakte interface elementen, maar aanpassingen aan de specifieke app kost nog steeds tijd. Verder is het gebruik van toetsenborden niet altijd even evident, aangezien verschillende platformen hier anders mee om gaan, wat ten koste gaat van de gebruiksvriendelijkheid van de app.

Phonegap is dus relatief eenvoudig om te gebruiken en om van een website een app te maken, maar dit heeft als gevolg dat de app minder performant is, zeker grafisch zware apps. Ook als er nood is aan veel rekenkracht zal de performantie aanzienlijk lager liggen aangezien er geen directe toegang is tot native graphics acceleration of dergelijke. Hier bestaan echter wel workarounds voor, maar die zullen nog steeds niet kunnen typen aan volwaardig native apps. Ook het gebrek aan multi-threading beperkt de performantie en het gebruiksgemak.

### Voor- en nadelen

#### Voordelen

Phonegap heeft een lage leercurve, wat zeker niet te onderschatten valt, en kan op vrij korte tijd resultaten leveren. Ook is er toegang tot heel wat Javascript libraries en plugins en een grote community. Toegang tot basis native hardware is altijd mogelijk.

#### Nadelen

Cross platform is niet helemaal cross platform, zeker niet aangezien de verschillende platformen verschillende browsers gebruiken. Ook al zijn er heel wat plugins in de aanbieding, toch is toegang tot de hardware niet zo eenvoudig of niet altijd mogelijk, wat best wel een grote beperking is. Grotere apps worden vrij snel problematisch met betrekking tot cache en geheugen beheer, en performantie. Ook is het debuggen en testen niet altijd even eenvoudig. Er is verder geen standaard architectuur, wat het werken aan verschillende projecten met verschillende mensen ingewikkeld kan maken. Tot slot zijn games niet haalbaar door de slechte performantie en de beperkingen van HTML5 Canvas.

## Xamarin

### Korte omschrijving

Wederom, een web app op zo veel mogelijk platformen met zo weinig mogelijk moeite, en bijgevolg kost, is een ideaal uitgangspunt. Xamarin doet dit echter niet met web technologie, maar met een framework dat gebaseerd is op Mono. Geschreven in C#, ligt de leercurve aanzienlijk hoger voor de gemiddelde web developer.

De UI wordt in XML, een markup language, of varianten geschreven, die niet bij iedereen in de vingers zit. Een voordeel hiervan echter is het feit dat deze interfaces native zijn, en bijgevolg veel beter aanvoelen en reageren voor de gebruiker. Nadelig is dan weer dat er voor elk platform een apart interface gemaakt moet worden, aangezien platformen niet allemaal dezelfde functionaliteiten hebben en de interfaces niet allemaal op dezelfde manier gebouwd zijn. Xamarin werkt sinds 2014 aan Xamarin.Forms, dat dit probleem op moet lossen. Het laat toe UI’s te bouwen, gebruik makende van C# of XAML. De controls in deze interfaces zijn gelinkt aan de respectievelijke native varianten, wat in se toelaat dit ene interface te gebruiken op alle platformen, wat de app er uniform laat uitzien, maar toch telkens binnen de stijl van dat specifieke platform. Xamarin.Forms staat echter nog niet helemaal op punt, maar is dus wel iets waar in de toekomst op gerekend zal kunnen worden.

Zoals hierboven vermeld, gebruikt Xamarin C# als taal, door velen gezien als een van de betere programmeertalen die er momenteel bestaan en een van de top vijf gebruikte op dit moment. .NET en Mono hebben een grote community en code base, wat het vinden van voorbeelden en oplossingen van bugs aanzienlijk kan versnellen. Ook kan NuGet gebruikt worden voor packages te installeren, wat een wereld van herbruikbare en voorgemaakte functionaliteiten openstelt. Verder hebben apps ook een duidelijkere structuur, MVC of MVVM, wat duidelijkheid van de code en werken in team ten goede komt.

Een van de grote troeven van Xamarin in deelbare code over de verschillende platformen. Native apps worden meestal per platform geschreven, wat driemaal het zelfde werk met zich meebrengt, als er uit gegaan wordt van ondersteuning voor Android, iOS en Windows. Dit werk kan meestal niet door dezelfde mensen worden uitgevoerd, aangezien veel developers niet zowel native Android als iOS en Windows apps kunnen bouwen. Xamarin zorgt ervoor dat grote delen van de code voor alle drie de apps bruikbaar zijn, wat het multipliceren van code met dezelfde functionaliteit bijna tot nul kan reduceren. Logica is bijna in zijn geheel deelbaar, en met Xamarin.Forms’s UI wordt heet aandeel deelbare code nog vergroot. Blijft er vooral nog toegang tot de native hardware over, wat voor elk platform wat verschilt, en dus ook afzonderlijk afgehandeld moet worden. Dit is echter slechts een klein deel, en er kan dus uit gegaan worden van minimum vijfenzeventig procent deelbare code, wat in sommige gevallen zelfs bijna de honderd procent kan bereiken.

Aangezien er voor elk platform een native app uit het proces komt, ligt de performantie zeer hoog. Ook multi-threading is beschikbaar, wat dit zeker in de hand werkt. Zoals eerder gezegd voelen native apps beter aan voor de gebruiker, wat deels te danken is aan deze hogere performantie.

Tot slot biedt Xamarin met zijn Test Cloud een zeer goede omgeving om apps uitgebreid uit te testen. Het geeft toegang tot een cloud met enkele honderden apparaten, waarom de app getest kan worden, waarna een visueel duidelijke analyse de vinger legt op knelpunten.

### Voor- en nadelen

#### Voordelen

Voor heel wat developers is werken met C# in Visual Studio een hele opluchting. Goede ondersteuning tijdens het developen en een grote code base zijn zeker voordelen. Ook bevordert werken binnen een MVC of MVVM design patroon teamwerk. Verder is een echte, volwaardige native app als eindresultaat met betere performantie en gebruiksgemak zeker niet te onderschatten in waarde. De interfaces zullen in de toekomst allemaal grotendeels kunnen komen van dezelfde code, eens Xamarin.Forms op punt staat, wat het proces van verschillende native apps en hun respectievelijke UI bijna volledig zal kunnen vervangen. Ook de deelbaarheid van logica is een zeer grote tijdsbespaarder. Tot slot is er de Test Cloud, die voor betere apps bij launch zorgt.

#### Nadelen

Hoewel voor een deel van de developers C#, Mono en Visual Studio gebruiken een godsgeschenk zal zijn, kan dit de leercurve aanzienlijk verhogen. Verder is het een Mono framework, geen echte .NET, wat links en rechts voor problemen kan zorgen. Ook is er nog heel wat plek voor verbetering betreffende de documentatie van Xamarin zelf. Tot slot staat een van hun grootste troeven, de deelbaarheid van code, nog niet helemaal op punt. Er is een groot deel van de logica deelbaar, maar op het interface gedeelte van Xamarin.Forms kan nog niet volledig gerekend worden, aangezien het nog beperkt is qua omvang.

# Conclusie

De vraag die beantwoord dient te worden is: welk van beide is het beste voor app ontwikkeling? Hierboven werden zowel Phonegap als Xamarin uit de doeken gedaan met hun voor- en nadelen. Hieronder zullen deze tegenover elkaar gezet worden en conclusies getrokken worden.

Ten eerste is er de leercurve van beide, waar Xamarin moet onderdoen voor de eenvoud en algemene bekendheid van HTML, CSS en Javascript. Beide hebben een grote comunity, veel plugins en libraries, maar waar ze bij Phonegap niet altijd even eenvoudig te gebruiken zijn of niet beschikbaar voor alle platformen zijn, zijn ze bij Xamarin gemaakt voor .NET en niet Mono, wat beide in een minder dan ideale situatie brengt. Xamarin heeft dan weer wel een zeer goed ondersteunde toegang tot native hardware, waar er bij Phonegap slechts een basis gegarandeerd kan worden en verdere toegang via niet altijd beschikbare of gebruiksvriendelijke plugins moet gaan. Ook het gebrek aan een vaste structuur, waar zijn tegenstander MVC of MVVM gebruikt, is nadelig voor Phonegap. Tot slot is er het volledig nateve zijn van Xamarin apps een grote troef op zowat alle vlakken.

Hieruit kan besloten worden dat, globaal gezien, Xamarin aan te raden is boven Phonegap. Dit moet echter genuanceerd worden, aangezien Phonegap het go-to framework is in situaties zoals gebrek aan kennis van Mono en gebrek aan fondsen om een Mono developer in te huren of tijd om het te leren. Hier is Phonegap zeker op zijn plaats, zolang de app klein is en geen toegang nodig heeft tot geavanceerde native hardware. Op lange termijn is het zeker de moeite om met Xamarin te leren werken indien er regelmatig apps van de band moeten rollen.

# Bronnen

Stackoverflow, Xamarin Fora, Phonegap Fora, Blogs (o.a. van Xamarin en Phonegap developers)