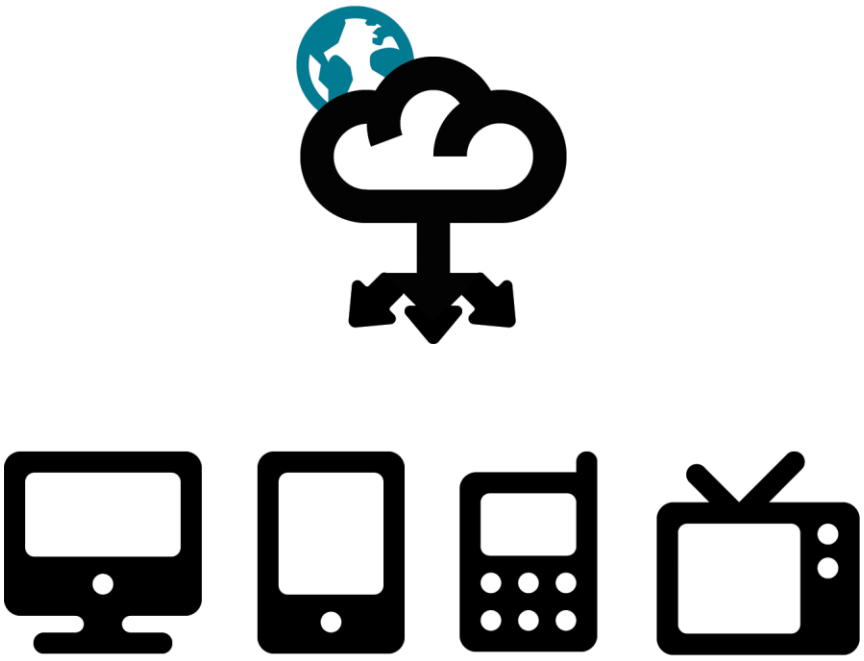
multi-device website



Jacob van Zijp

in opdracht van evident

Titel	Multi-device website
Ondertitel	Evident
Opleiding	Digitale Communicatie
Bedrijfsbegeleider	Dhr. Marc Bruisten
Schoolbegeleider	Dhr. Guus Koning
Afstudeerperiode	30 januari 2012 – 15 juni 2012
Opdrachtgever	Evident BV
Telefoon	0636164651
Door	Jacob van Zijp

Samenvatting

Opdracht
Deze scriptie is geschreven als onderdeel van een afstudeerstage bij Evident vanuit de Hogeschool Utrecht. Het onderzoek dat is uitgevoerd richtte zich op de techniek achter het opzetten van een website voor verschillende internet capabele apparaten (een multi-device website). De hoofdvraag luidt dan ook:

Hoe moet een website technisch opgezet worden om alle devices te kunnen bedienen?

Werkwijze
De scriptie is in eerste instantie opgebouwd uit 4 afzonderlijke onderzoeken. Twee maal een literatuuronderzoek, interviews en een expertonderzoek. Later bleek een additioneel hoofdstuk nodig. De noodzaak voor hoofdstuk 'Realisatie' bleek tijdens de uitvoering van Literatuuronderzoek II. Hierin werd duidelijk dat voor twee struikelblokken uit literatuuronderzoek I geen passende mogelijkheden waren. Daarom zijn er voor de punten: 'scripts conditioneel inladen' en 'media optimalisatie' twee op maat oplossingen ontwikkeld:

- Conditionele Javascript lader
- Media Server

Tijdens het eerste literatuuronderzoek is er onderzoek gedaan naar de struikelblokken die zich voordoen bij het ontwikkelen van een multi-device website bij Evident. De interviews haken hierop in. In literatuuronderzoek II zijn de mogelijkheden en technieken onderzocht. Hierna zijn deze mogelijkheden en technieken beoordeeld in het expertonderzoek.

Conclusie
Uit de scriptie is te concluderen dat het goed mogelijk is een multi-device website op te zetten, gezien vanuit het technisch oogpunt. Door de op maat oplossingen welke ontwikkeld zijn tijdens het schrijven van deze scriptie en de gevonden mogelijkheden zijn de technische middelen aanwezig om dit op een professionele wijze mogelijk te maken.

Voorwoord

Deze scriptie is geschreven voor de opleiding digitale communicatie aan de Hogeschool Utrecht in opdracht van Evident BV te Utrecht.

Deze scriptie heb ik geschreven om mijn groeiende passie voor programmeren en mobiel te kunnen combineren. Met deze opdracht hoopte ik van beide vlakken veel te leren en uiteindelijk ook daadwerkelijk te kunnen bijdragen op technisch vlak bij de ontwikkeling van multi-device websites.

Ik wil graag mijn afstudeerbegeleider vanuit de hogeschool, Guus Koning, bedanken voor zijn inzet en waardevolle advies die ik heb mogen ontvangen tijdens het schrijven van deze scriptie. Hiernaast wil ik ook mijn afstudeerbegeleider vanuit Evident bedanken. Marc Bruisten heeft mij tijdens de ontwikkeling van deze scriptie alle benodigde ruimte gegeven en waar nodig ondersteund en gestuurd.

1. Inleiding

In het moderne web landschap volgen de ontwikkelingen en de veranderingen elkaar snel op. Innoveren blijkt noodzakelijk om mee te kunnen blijven doen in dit bewegelijke wereldje.

Zo breekt er nu voor Evident een moment aan dat er aan een omschakeling gewerkt moet worden. Deze omschakeling heeft te maken met toenemende diversiteit van apparaten die met het internet verbonden zijn. En dan wel de apparaten die websites kunnen bezoeken. We zijn van alleen een desktop binnen een paar jaar overgesprongen op een heel arsenaal aan verschillende apparaten. Een mooie illustratie die deze verschuiving weergeeft:

Figuur 1: het web toen en nu: (Frost, 2012)

Deze verschuiving brengt naast nieuwe kansen en mogelijkheden ook een aantal complicaties met zich mee. Websites die ontworpen en ontwikkeld zijn voor een computer beeldscherm blijken vaak veel complicaties te hebben op mobiele apparaten. Een aantal voorbeelden van dit soort complicaties zijn:

- Een trage laadtijd van de website
- De website is slecht te lezen
- De website werkt helemaal niet
- Er moet te veel ingezoomd worden
- De website is onoverzichtelijk

Deze complicaties hebben een noodzaak van innovatie met zich meegebracht. De prestatie van websites is belangrijker dan ooit en met de komst van deze nieuwe apparaten wordt het steeds lastiger om aan deze behoefte te voldoen.

Evident heeft gemerkt dat de groei in het gebruik van mobiele apparaten sterke invloed heeft op de vraag naar én de verwachting van geoptimaliseerde websites voor mobiele apparaten.

De aanleiding tot dit onderzoek ligt bij de verschillende complicaties die Evident ondervind bij het ontwikkelen van een website die geschikt moet zijn voor meerdere apparaten.

Met apparaten en devices wordt hoofdzakelijk de volgende soorten bedoeld:

- Desktop computers / Laptops
- Mobiele telefoons
- Tablets

Dit zijn dan ook de drie soorten devices die in de scope van dit onderzoek worden opgenomen en van elkaar onderscheiden worden.

De complexiteit die is ontstaan door deze nieuwe internet capabele apparaten ligt voornamelijk bij de uiteenlopende schermgroottes en de hardware-matige verschillen van deze apparaten. Deze verschillen zijn onderling te groot om de apparaten een exact dezelfde website aan te bieden.

Naast de complicaties bieden de draagbare apparaten ook juist nieuwe mogelijkheden waarop ingespeeld kan worden. Met deze insteek kan de ervaring van een mobiele gebruiker sterk verbeterd worden doordat er context specifieke functies aan de beleving toegevoegd kunnen worden. Om een beter beeld te schetsen, volgt hier een voorbeeld van zo een context specifieke functie:

> *Persoon A zit in de trein op weg naar zijn werk en zit op zijn iPad op naar verschillende huizen te kijken. Bij het zoeken wordt er automatisch een regio ingevuld waarmee in combinatie met een geselecteerde straal gezocht kan worden.*

In dit voorbeeld wordt er door middel van GPS (locatie voorziening) automatisch een stad of locatie ingevuld zodat de gebruiker dit niet hoeft te doen. Dit scheelt de gebruiker tijd, moeite en mogelijke frustratie. Dit is een van de vele mogelijkheden waarmee de ervaring van een mobiele gebruiker verreikt kan worden door context specifieke eigenschappen, technieken of mogelijkheden te benutten.

Tot nu toe heeft Evident dit aangepakt door een geheel aparte mobiele website te produceren die voor dit doel diende. Door de beschikbare techniek van nu is er veel meer mogelijk waardoor een aparte mobiele website niet meer nodig is. Hiernaast brengt een multi-device website nog andere voordelen met zich mee. Al met al is er een behoefte ontstaan voor een aangepaste website voor mobiele apparaten.

De probleemstelling luidt als volgt:

Hoe moet een website technisch opgezet worden om alle devices te kunnen bedienen?

Deze probleemstelling wordt onderverdeeld in een aantal deelvragen waarvan de antwoorden gezamenlijk de probleemstelling gaan beantwoorden. Deze deelvragen luiden als volgt:

- Wat zijn de complicaties bij het opzetten van een website welke zowel mobiele als niet mobiele apparaten moet kunnen bedienen?
- Welke mogelijkheden zijn er om deze complicaties aan te pakken?
- Welke mogelijkheden zijn toepasbaar binnen Evident?
- Hoe verhouden deze technieken zich tot elkaar?

Wanneer deze deelvragen beantwoord zijn, wordt het resultaat samengevat in een conclusie. Vervolgens wordt er op basis van deze conclusie een voorbeeld website gebouwd of aangepast voor alle devices.

2. Onderzoeksopzet

Het onderzoek is opgedeeld in twee afzonderlijke onderzoeksfases. De eerste onderzoeksfase betreft een literatuuronderzoek. In dit literatuuronderzoek zullen de verschillende knelpunten waar Evident tegen aan loopt bij het ontwikkelen van een multi-device website aan het licht worden gebracht. Na het literatuuronderzoek vinden er interviews plaats. Deze interviews dienen als aanvulling op het literatuuronderzoek waarin de verschillende knelpunten verder worden onderzocht en additionele knelpunten worden vastgesteld.

Vervolgens zal er een tweede literatuuronderzoek plaats vinden waarin gezocht zal worden naar mogelijke oplossingen voor de knelpunten die aan het licht zijn gekomen tijdens de interviews en het eerste literatuuronderzoek.

Als laatste zal er een expertonderzoek worden uitgevoerd waarin de (in de voorgaande onderzoeken gewonnen) informatie geanalyseerd, getest en beoordeeld zal worden.

Een overzicht van de verschillende onderzoeken:

Literatuuronderzoek I: Identificeren knelpunten (deelvraag 1)

Interviews: Identificeren knelpunten (deelvraag 1)

Literatuuronderzoek II: Identificeren mogelijkheden (deelvraag 2)

Expertonderzoek: Mogelijkheden testen en beoordelen (deelvraag 3 & 4)

2.1 Literatuuronderzoek I

In de eerste stap naar een multi-device website is er gekozen voor een literatuuronderzoek. Het doel van dit onderzoek is het aan het licht brengen van de verschillende knelpunten die zich voordoen bij het ontwikkelen van een ‘multi-device website’.

Het literatuuronderzoek zal voornamelijk via het medium internet verlopen. Ook zal er gebruik worden gemaakt van verschillende boeken en magazines. De reden dat er voornamelijk gebruik zal worden gemaakt van het medium internet is, omdat het internet over het algemeen de meest recente informatie bevat.

De resultaten die voortkomen uit dit literatuuronderzoek zullen worden gecategoriseerd en per onderwerp worden uitgewerkt. De gevonden knelpunten zullen in een later stadium geanalyseerd worden en vervolgens zal er worden onderzocht of er mogelijke oplossingen zijn.

2.2 Interviews

Dit vervolgonderzoek haakt in op literatuuronderzoek I. Ook in dit onderzoek is het doel het aan het licht brengen van de verschillende knelpunten die zich voordoen bij het ontwikkelen van een 'multi-device website'.

De interviews zal worden uitgevoerd volgens de richtlijnen van een half gestructureerd interview. Dit houdt in dat er aan de hand van een 'topiclist' (of onderwerpenlijst) (zie bijlage I) een open interview wordt gehouden waar de vragen niet van te voren zijn vastgesteld.

Er is voor de half gestructureerd interview methode gekozen omdat deze methode meer speling bied binnen het interview. Dit zorgt ervoor dat er doorgevraagd kan worden en dat op andere onderwerpen ingegaan kan worden die niet van tevoren voorspeld zijn. Met deze methode wordt de kans op een kwalitatief beter onderzoek vergroot.

Voorafgaand aan de daadwerkelijke interviews wordt een topiclist samengesteld die betrekking heeft op het onderwerp van het interview. Het onderwerp van het interview wordt bepaald door de functie van de geïnterviewde werknemer van Evident.

In totaal zullen er vier interviews plaats vinden. Voor deze interviews is gekozen voor vier personen met verschillende expertises. Er is gekozen voor een mobiele interactie designer, designer, front-end developer en een reguliere interactie designer. Er is voor deze diversiteit gekozen omdat door de verschillende perspectieven een goed beeld ontstaat van alle verschillende knelpunten.

2.3 Literatuuronderzoek II

Het literatuuronderzoek II dient als tweede stap na de interviews en het eerste literatuuronderzoek . De kennis welke in de vorige stappen is verzameld, wordt gebruikt om dit literatuuronderzoek te vormen. In de praktijk zal dit betekenen dat de in de vorige stappen geïdentificeerde knelpunten geanalyseerd zullen worden, waarbij vervolgens naar een bijpassende oplossing wordt gezocht met behulp van dit tweede literatuuronderzoek.

Deze knelpunten zullen als eerst gecategoriseerd en geprioriteerd worden. Op deze manier kan er gericht naar een oplossing gezocht worden en wordt aan de meest belangrijke knelpunten voorrang verleent. Voorafgaand aan het onderzoek is nog niet vast te stellen op welke punten het tweede literatuuronderzoek in zal gaan.

2.4 Expertonderzoek

In het expertonderzoek zullen alle mogelijkheden en technieken (verzameld tijdens het literatuuronderzoek) geanalyseerd en getest worden. Uit dit proces moet een selectie van één of meerdere mogelijkheden of technieken voortkomen. Tijdens deze analyse en testfase worden de technieken aan de hand van een aantal voorwaarden getoetst. De voorwaarden zijn in onderstaand schema weergegeven:

Voorwaarden	Beschrijving
Future Friendly	De oplossing moeten voldoen aan de richtlijnen van de Future Friendly filosofie. *(Wroblewski, Future Friendly, 2011)*
Toepasbaar binnen de systemen	De oplossing dient volledig toepasbaar te zijn binnen de systemen van Evident. Dit houdt in dat er geen gebruik gemaakt kan worden van technieken die niet worden ondersteund binnen Evident
Cross-browser	De oplossing dient in alle browsers goed te werken. De minimale eisen hiervan zijn. IE7+ FF2+ Chrome 5+ Opera 10+
Schaalbaarheid	De oplossing dient schaalbaar te zijn. En website van grootte omvang of met veel bezoekers dient geen complicaties op te leveren
Snelheid	Een website mag er niet langzamer op worden door het gebruik van de oplossing
Stabiliteit	De oplossing dient stabiel en onbetrouwbaar te zijn

3. Onderzoeksresultaten

3.1 Literatuuronderzoek I

In het eerste literatuuronderzoek zijn de technische struikelblokken aan het licht gebracht, welke zich voordoen bij het ontwikkelen van een multi-device website. Deze knelpunten zijn onderverdeeld in de volgende onderdelen:

- Device capabiliteit detectie
- Laadtijd optimalisatie
- Media optimalisatie

Het laatste punt: Media optimalisatie zal later in deze scriptie (hoofdstuk realisatie) besproken worden.

3.1.1 Device capabiliteit detectie

Om een website te bouwen welke goed te gebruiken is op elk soort device, moet er een vorm van apparaat detectie aanwezig zijn. De apparaat detectie zorgt ervoor dat er vast gesteld kan worden over welke eigenschappen of mogelijkheden de devices van de bezoekers beschikken. Er moet een detectie worden uitgevoerd waaruit blijkt over welke schermgrootte het device beschikt. Ook is het interessant om te weten over welke capaciteiten het device beschikt. De verschillende eigenschappen zullen hier besproken worden:

Schermgrootte

De schermgrootte is een interessante eigenschap van het device van de bezoeker, omdat deze de layout en de presentatie van de website voor een groot deel kan bepalen. Dit heeft te maken met de uiteenlopende schermgroottes van alle verschillende devices. Deze variëren van 240 bij 320 tot wel 2560 bij 1600 pixels. In onderstaand tabel is af te lezen welke veel voorkomende schermgroottes er zijn.

	Lage resolutie **(120 dpi), *ldpi***	Middel resolutie **(160 dpi), *mdpi***	Hoge resolutie **(240 dpi), *hdpi***	Extra hoge resolutie **(320 dpi), *xhdpi***
Klein scherm	*QVGA* 240x320		480x640	
Normaal scherm	*WQVGA400* 240x400	*HVGA* 320x480	*WVGA800* 480x800 *WVGA854* 480x854 600x1024	640x960
Groot scherm	*WVGA800* 480x800 *WVGA854* 480x854	*WVGA800* 480x800 600x1024		
Extra groot scherm	1024x600	*WXGA* 1280x800 1024x768 1280x768	1536x1152 1920x1152 1920x1200	2048x1536 2560x1536 2560x1600

Tabel 1: (Android, 2012)

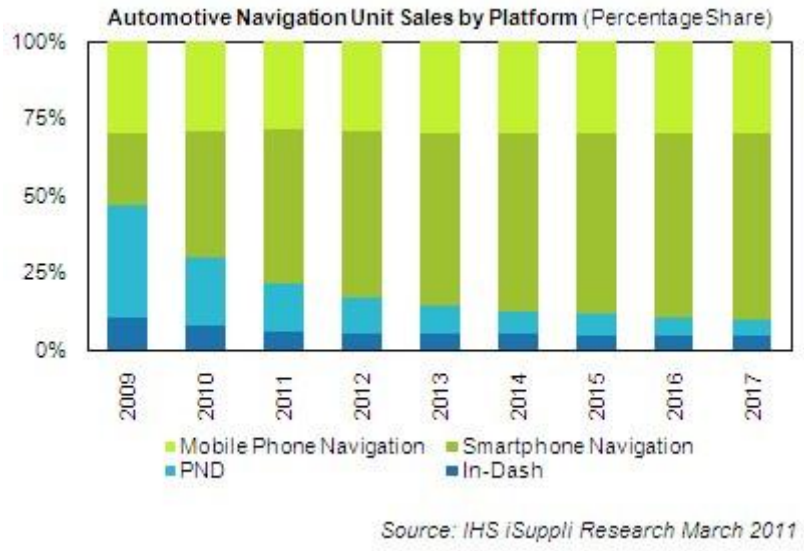
Door de grote variatie aan beschikbare schermruimte moeten er keuzes worden gemaakt hoe een website reageert en gebruik maakt van deze ruimte. Deze keuzes kunnen pas gemaakt worden wanneer er bekend is om welke afmetingen het gaat.

Touch screen

De nieuwste generaties smartphones en tablets beschikken veelal over een touchscreen. Het gebruik van een touchscreen biedt een gehele andere gebruikerservaring dan een bediening met behulp van "hardware-knoppen". Uit onderzoek is gebleken dat een gebruiker een andere gebruikerservaring ervaart wanneer er van een touchscreen gebruik wordt gemaakt. Uit dit onderzoek is bijvoorbeeld gebleken dat het aantal fouten dat wordt gemaakt bij en aanraken/aanklikken van kleine elementen vele malen hoger ligt bij touchscreen devices dan bij devices zonder touchscreen. *(Bender, 1999)* en *(Sesto, 2011)*

GPS

Veel moderne devices beschikken over een GPS chip. Met de GPS functionaliteit kan de locatie van de gebruiker interactief gebruikt worden in een website. Volgens een onderzoek van iSupply bestaat de huidige verkoop van GPS capabele apparaten al voor 80% uit telefoons en smartphones. Er wordt verwacht dat dit alleen nog maar zal stijgen *(Oh, 2011)*. Dit is te zien in onderstaand diagram:



Figuur 2: Auto navigatie verkoop per platform (Oh, 2011)

Met behulp van GPS kan de gebruikerservaring verreikt worden. Een voorbeeld hiervan kan zijn:

> *Op een mobiele website van een vacature website kan gezocht worden binnen een straal vanaf je huidige locatie. Deze locatie kan automatisch ingevuld worden door middel van de GPS chip. Het gevolg hiervan is dat de gebruiker geen locatie hoeft in te typen. Dit bevorderd de gebruikerservaring doordat de zoek procedure wordt versimpeld en hiermee versneld.*

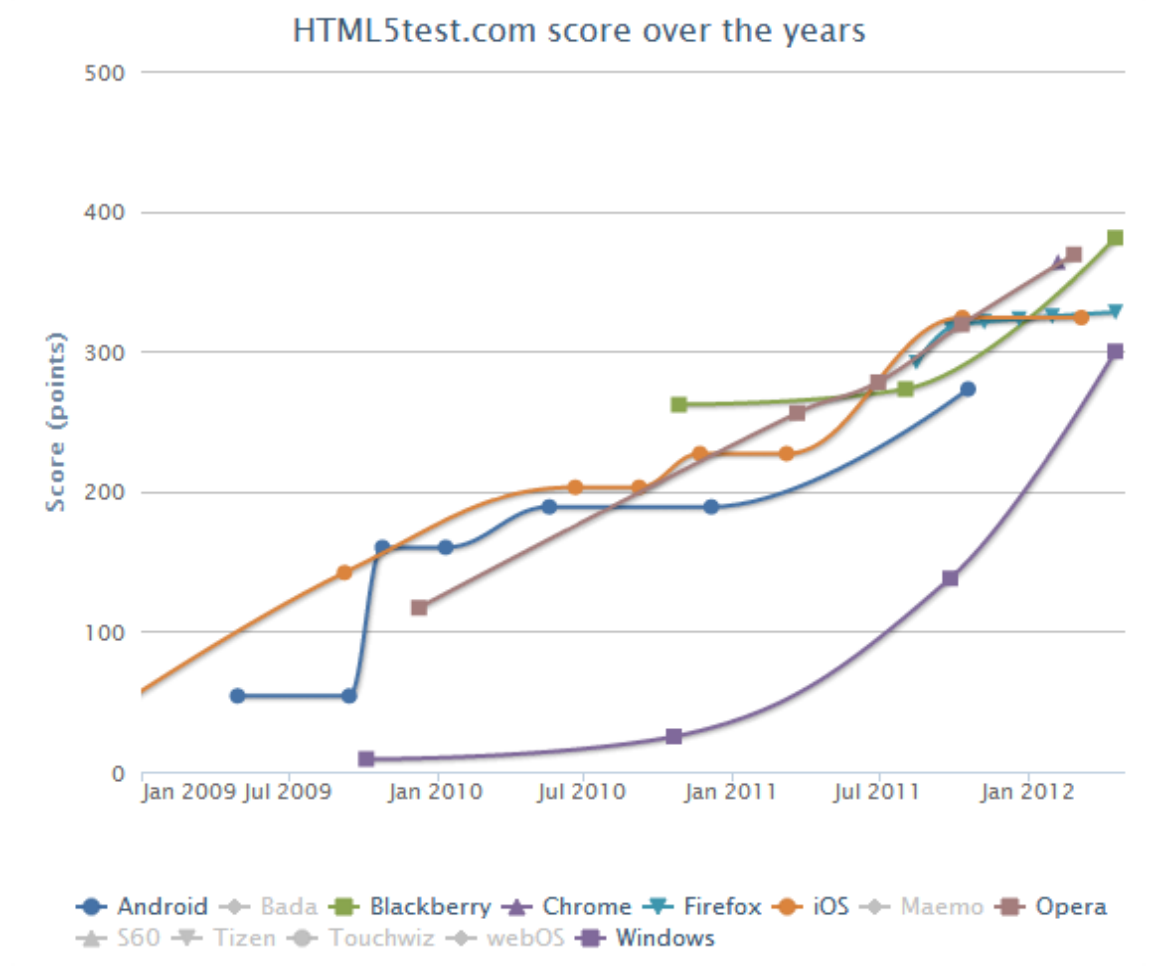
Internet snelheid

De hoogte van de internet snelheid is variabel. Bij desktop computers is het veelal een hogere snelheid maar bij mobiele devices loopt dit erg uiteen. Dit heeft te maken met de verschillende soorten internet connectiemogelijkheden. De device kan verbonden zijn door middel van Wi-Fi (vaak een hogere snelheid) maar ook in de vorm van mobiel internet. Connectievormen van mobiel internet zijn:

2G GSM :	9,6 kbps of veelvouden hiervan met HSCSD
2.5G GPRS:	tot 52 kbps
2.75G EDGE:	tot 384 kbps
3G UMTS:	tot 384 kbps
3.5G HSDPA:	tot 14 Mbps
3.9G LTE:	tot 326 Mbps
4G LTE Advanced:	tot 1 Gbps

De laatste twee vormen (LTE & LTE Advanced) zijn op het moment nog niet van toepassing in Nederland. Op dit moment is de meest voorkomende internet connectie vorm GPRS en HSDPA.

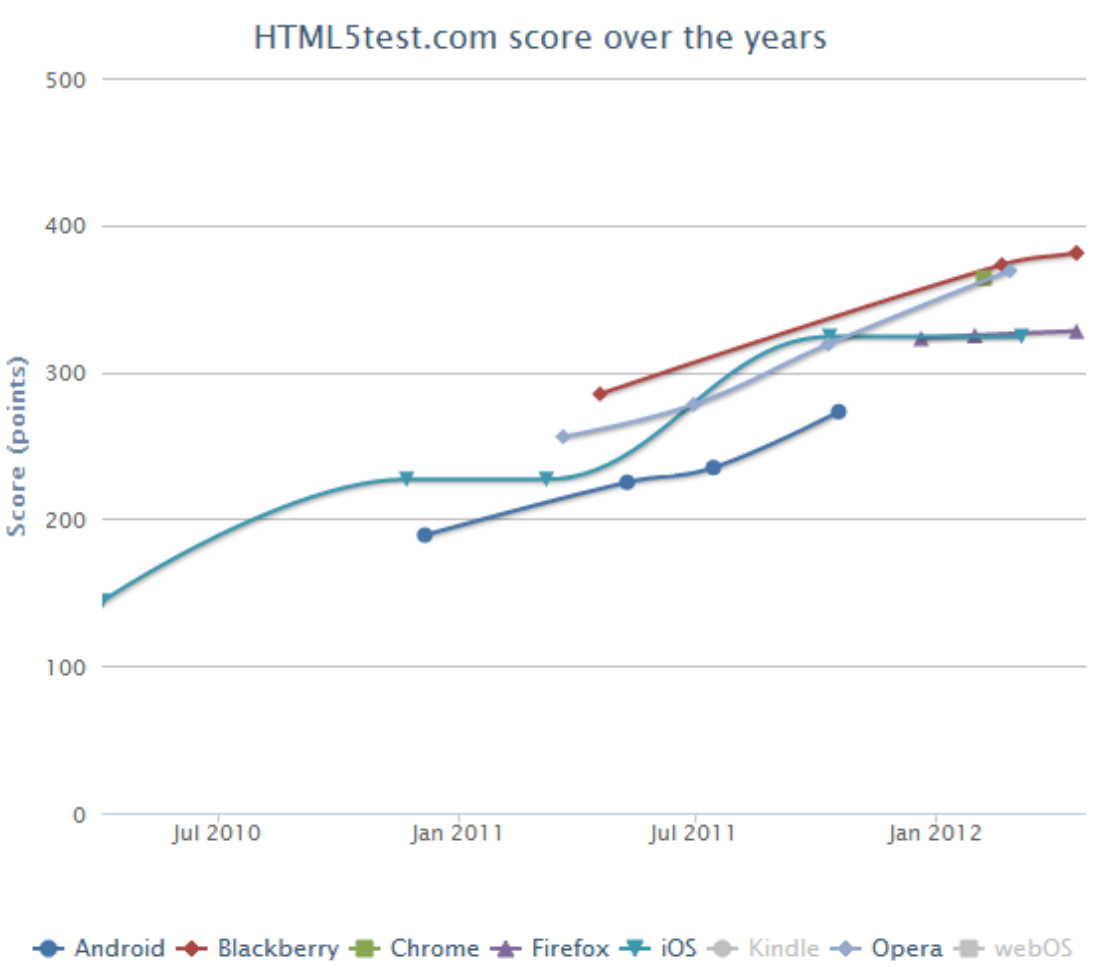
HTML5 browser ondersteuning



Figuur 3: Browser score mobiele telefoons (Sights, 2012)

Het detecteren van HTML5 capabiliteit is een knelpunt om mee rekening te houden. Dit komt doordat niet alle browsers van alle devices dezelfde ondersteuning bieden qua HTML5 mogelijkheden. Wanneer hier geen rekening mee wordt gehouden kan dit zorgen voor grote inconsistentie of foutmeldingen. Er dient op deze gewenste capabiliteit getest te worden waardoor er een 'fallback' (terugval, alternatieve functie of effect) kan worden gebruikt om deze missende functionaliteit op te vangen. Deze terugval moet gaan volgens het principe 'progressive enhancement' *(Gustafson, 2008).*

In onderstaand figuur 3 is de vergelijking te zien tussen de verschillende browsers voor mobiele telefoons. De vergelijking wordt gemaakt op basis van een score. Deze score wordt berekend aan de hand van de HTML5 ondersteuning van de browser. Deze score is de maatstaf van de website HTML5test.com om verschillende browsers met elkaar te vergelijken. Deze maatstaf is door HTML5test zelf ontwikkeld.



Figuur 4: Browser score tablets (Sights, 2012)

In dit figuur 4 heeft dezelfde insteek als het vorige alleen specifiek gericht op tablets.

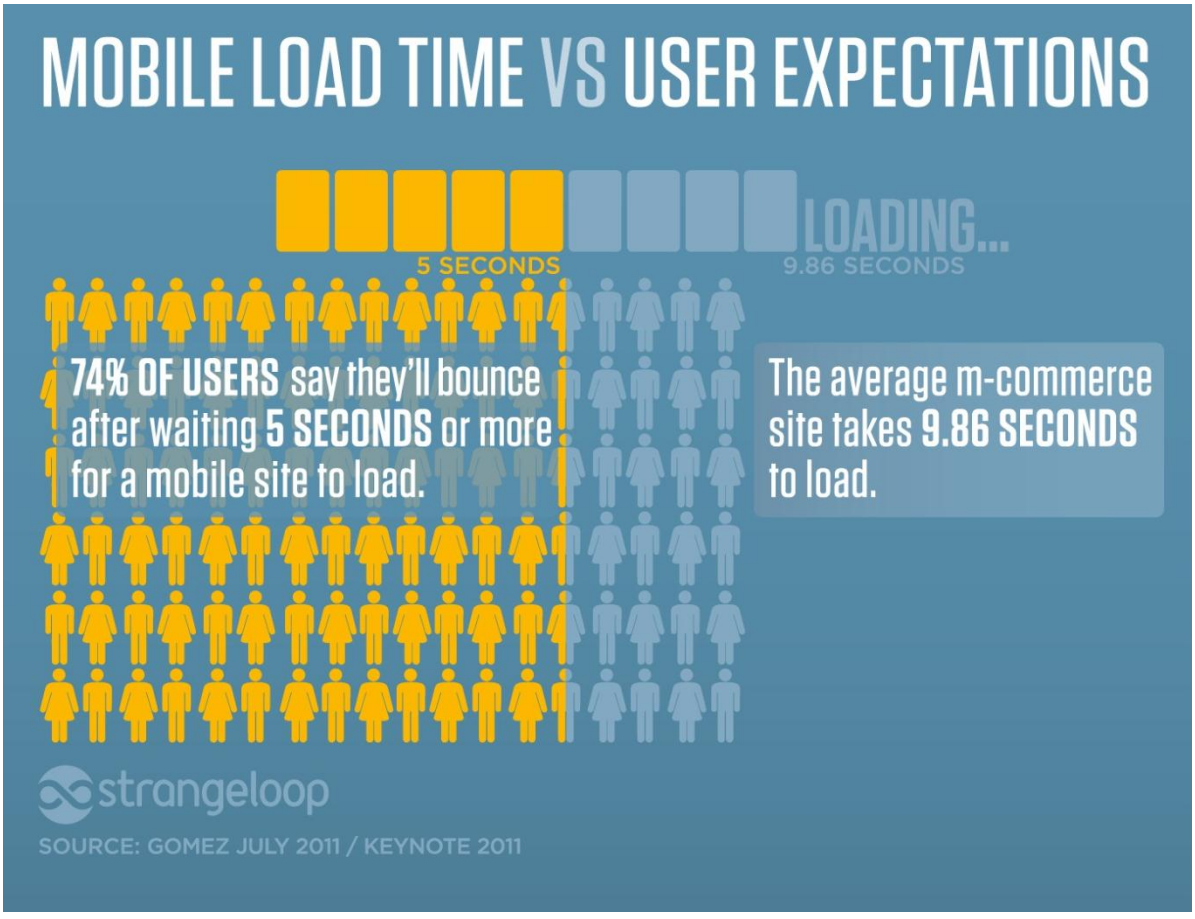
Uit deze figuren is af te leiden dat score in een korte tijd snel toeneemt. Dit heeft als gevolg dat er veel verschillende apparaten met veel verschillende mogelijkheden in gebruik zijn. Dit houd in dat oudere devices veel minder HTML5 mogelijkheden bevatten dan nieuwere devices. Hierdoor kan er lastig van moderne technieken gebruik gemaakt worden zonder compatibiliteit problemen te veroorzaken bij oudere apparaten.

CSS3 capabiliteit

Net als met HTML5 geld hetzelfde voor CSS3. Met deze laatste CSS specificatie zijn er een aantal nieuwe technieken en mogelijkheden toegevoegd. Niet al deze nieuwe specificaties zijn (volledig) geïmplementeerd in alle browsers. Hierdoor is er net als bij HTML5 een verschil aan mogelijkheden qua CSS3. Dus ook bij CSS3 is het lastig van nieuwe technieken gebruik gemaakt worden zonder compatibiliteit problemen te veroorzaken bij oudere apparaten.

3.1.2 Laadtijd optimalisatie

De laadtijd wordt steeds belangrijker voor alle verschillende apparaten. Dit heeft te maken met een aantal zaken. Als eerst verwachten de gebruikers relatief snelle laadtijden *(Gomez, 2011)*. In de figuur hieronder is af te lezen dat de gemiddelde laadtijd van een website ver boven de maximaal aanvaarde wachttijd (gemiddeld vijf seconden) zit. Dit zorgt voor veel afhakers onder de mobiele bezoekers.



Figuur 5: Mobiele laadtijd VS gebruikersverwachting gebaseerd op: (Gomez, 2011)

Om de laadtijd onder het maximum te houden dient deze goed geoptimaliseerd te worden. Deze laadtijd kan op verschillende manier geoptimaliseerd worden.

De verschillende verbeterpunten opgesomd:

- Script optimalisatie
- Scripts conditioneel inladen
- Stylesheet optimalisatie
- HTTP requests minimaliseren
- Media optimalisatie
- Caching

Script optimalisatie
Script optimalisatie is te verdelen in twee soorten. Deze zijn script minimalisatie en script compressie.

- **Script minimalisatie**
  Script minimalisatie houd in dat het script zo klein mogelijk wordt gemaakt. Dit wordt gedaan door het script van alle witruimte te ontdoen. Wanneer dit gebeurd is, kan het script soms vele malen kleiner worden in omvang dan voorheen.

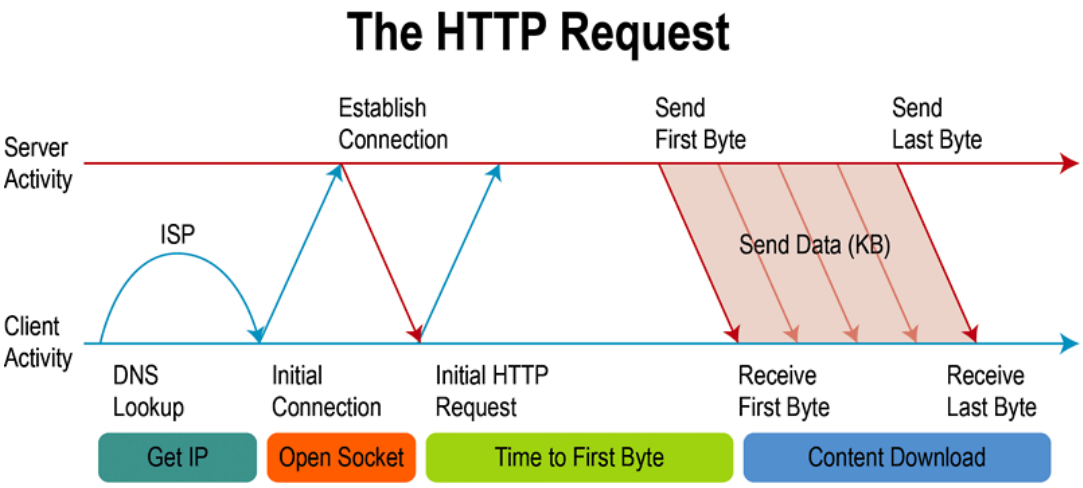
- **Script compressie**
  Naast script minimalisatie is er ook script compressie. Dit houdt in dat een script wordt herschreven om de prestaties te verhogen zonder de functionaliteit aan te passen. De prestatie kan worden verhoogt doordat er meerdere manieren zijn hetzelfde resultaat te krijgen bij het programmeren. Sommige presteren beter dan de anderen. De handeling kan sneller gaan of minder processorkracht kosten. Door een bewuste keuze te maken bij het schrijven van code kan de prestatie en laadtijd van het script sterk verbeterd worden. Vooral op mobiele devices kan dit een grote winst opleveren in verband met de relatief lage verbindingssnelheid en lage processorkracht.

Scripts conditioneel inladen
Doordat de verschillende devices erg van elkaar verschillen, is niet iedere functionaliteit beschikbaar voor ieder apparaat. Hierdoor wordt er vaak onnodig bestanden ingeladen waar vervolgens niets mee gedaan wordt. Dit dient voorkomen te worden door deze bestanden niet in te laden wanneer deze niet nodig zijn. Dit heet het conditioneel inladen van bestanden. Wanneer de capaciteiten van het device van de bezoeker bekend zijn, wordt aan de hand van deze gegevens gekeken welke bestanden benodigd zijn. Op deze manier wordt alleen ingeladen wat nodig is en wordt er geen laadtijd verspild.

Stylesheet optimalisatie
Ook stylesheets dienen net als scripts geminimaliseerd te worden om de prestatie te verbeteren. De bestandsgrootte zal hierdoor aanzienlijk verkleind worden waardoor deze sneller gedownload kan worden door de browser van de bezoeker. Door deze snelheidsverbetering zal de website dan ook sneller getoond worden aan de gebruiker.

HTTP requests minimaliseren

Door HTTP requests te minimaliseren zal de laadtijd van een website aanzienlijk verbeterd worden. HTTP requests zorgen voornamelijk voor langzaam ladende pagina's. In onderstaand schema is te zien welke activiteiten er worden uitgevoerd bij iedere aanvraag.



Figuur 6: Overzicht http Request (King, 2002)

Kort samengevat gebeurd het volgende:

1. De browser van de gebruiker maakt een aanvraag naar een url
2. Deze URL wordt word door de DNS server omgezet in een IP adres met het juiste pad
3. Connectie wordt gemaakt
4. Uiteindelijke aanvraag wordt verstuurd
5. Server stuurt bestand
6. Browser ontvangt bestand

Wanneer deze hele reeks doorlopen moet worden voor een groot aantal aanvragen kan dit voor onnodige laadtijden zorgen. Deze laadtijden zijn vaak niet nodig omdat er in de meeste gevallen aanvragen gecombineerd kunnen worden. In de praktijk betekend dit dat css, javascript en afbeeldingen gecombineerd kunnen worden in één enkel bestand per soort.

Media optimalisatie

Door media te optimaliseren of helemaal niet in te laden voor specifieke devices van de bezoeker wordt de laadtijd geoptimaliseerd. Zo kan een afbeelding voor bijvoorbeeld een iPhone 3GS geschaald worden naar een breedte van 320 pixels. De oorspronkelijke afbeelding kan in sommige gevallen wel 3 maal zo groot zijn. Alle media zouden geoptimaliseerd moeten worden om de bestandsgrootte te beperken. Op deze manier kan er zonder kwaliteitsverlies een grote snelheidswinst geboekt worden. Meer over Media optimalisatie in de paragraaf "Media optimalisatie".

Caching

Caching is het fenomeen dat er voor zorgt dat een webpagina sneller geladen kan worden. In de praktijk bestaan er twee soorten caching:

- **HTTP caching**
  HTTP caching zorgt ervoor dat de gerendeerde pagina opgeslagen wordt door de caching module. Dit zorgt ervoor dat wanneer dezelfde pagina nogmaals aangevraagd wordt deze niet iedere keer gecompileerd hoeft te worden. Dit zorgt er in de praktijk voor dat dynamische websites veel sneller geladen kunnen worden, omdat er geen backend code aan te pas hoeft te komen.

- **Query caching**
  Query caching is het opslaan van queries in de database. Deze soort van caching heeft alleen invloed op de prestaties van de database. Dit heeft als voordeel dat flexibeler gewerkt kan worden en dat het vaak minder complicaties oplevert. Het nadeel van query caching is dat het niet zo effectief als HTTP caching is.

3.1.3 Media optimalisatie

Zoals eerder besproken is media optimalisatie belangrijk voor het verbeteren van de laadtijd van een website. In deze paragraaf wordt er dieper ingegaan op de verschillende soorten van media optimalisatie. Als eerst wordt er ingegaan op device afhankelijke afbeeldingsafmetingen. Vervolgens wordt er ingegaan op andere mogelijkheden om de afbeeldingsgrootte te minimaliseren.

Afbeelding grootte
De afbeeldinggrootte is de belangrijkste factor voor de afname in bestandsgrootte. De bestandsgrootte dient dus zo ver mogelijk teruggedrongen te worden om zo de afbeelding zo klein mogelijk te krijgen. De afbeelding dient geschaald te worden aan de hand van de beeldschermbreedte van de gebruiker. Zo hoeft een afbeelding op een iPhone 3GS maar maximaal 320 pixels breed te zijn omdat het scherm van dit device deze breedte bezit (wanneer de device in landschapsmodus wordt gehouden wordt dit 480 pixels).

Afbeelding optimalisatie
De afbeelding dient maximaal geoptimaliseerd te worden zodat er zonder kwaliteitsverlies een zo klein mogelijke bestandsgrootte gerealiseerd kan worden. De optimalisatie wordt bereikt door onnodige informatie uit het bestand te strippen waardoor er alleen de data overblijft welke nodig is om de afbeelding op het beeldscherm weer te geven.

Afbeelding comprimeren
Door afbeeldingen te comprimeren neemt de bestandsgrootte aanzienlijk af. Wat hier tegenover staat is dat er kwaliteitsverlies optreed. Wanneer de afbeelding met een kleine hoeveelheid (bijvoorbeeld 10 of 20 procent) wordt gecomprimeerd is het kwaliteitsverlies nauwelijks op te merken en kan de bestandsgrootte met wel 30 tot 50 procent afnemen.

3.2 Interviews

In dit hoofdstuk worden de resultaten uit de interviews gepresenteerd en vervolgens wordt een conclusie getrokken. Er hebben vier interviews plaats gevonden met verschillende personen uit verschillende disciplines.

Hieronder een overzicht van de interviews:

Interview 1: Linda Stuurman, Front-end developer(afgestudeerd op mobile interaction design)

Interview 2: Sven Dinslage, Designer

Interview 3: Marc Bruisten, Front-end developer

Interview 4: Joost Verweij, Interaction designer

3.2.1 Interview met Linda Stuurman

In dit half-gestructureerde interview gehouden met Linda Stuurman is door middel van een 'topiclist' een interview tot stand gekomen die de knelpunten weerspiegelen die zich voordoen bij het proces van een mobiele website opzetten. De samenvatting van deze knelpunten worden hieronder weergegeven. Deze knelpunten zijn hierin onderverdeeld in twee categorieën, namelijk:

- Aparte mobiele website
- Mobiele website door middel van responsive design

Vaak wordt er voor een aparte mobiele website gekozen omdat het al te laat is om een responsive website te realiseren. Volgens mevr. Stuurman is het lastig om een responsive website te realiseren met terugwerkende kracht. Het knelpunt ligt dan bij het feit dat in het allereerste stadium van de ontwikkeling van een website (meestal het interactie ontwerp) rekening gehouden dient te worden met responsive elementen. Dit is nodig omdat het anders lastig wordt het volledige potentieel dat mobiele apparaten bieden te benutten. Hiernaast ontstaat er een kans dat het gerealiseerde resultaat van een responsive website (die met terugwerkende kracht ontwikkeld is) minder bruikbaar is dan de oorspronkelijke variant.

Er zijn naast een aantal technische redenen ook een paar praktische redenen waarom er voor een aparte mobiele website gekozen wordt. Een van deze redenen is het feit dat er gebrek aan kennis op het gebied van responsive technieken. Dit gebrek zorgt ervoor dat het gemakkelijker is om voor een bekende manier van werken te kiezen.

Een andere reden is mogelijk dat er meer verdiend kan worden met een afzonderlijke mobiele website.

Mevr. Stuurman vertelt dat er ook juist situaties voorkomen waarin er bewust wordt gekozen voor het ontwikkelen van een aparte mobiele website. Dit heeft dan vaak te maken met de opzet en de omvang van de mobiele website. Vaak is het met grotere en geavanceerdere websites niet praktisch om van responsive technieken gebruik te maken. Hierdoor kan volgens Mevr. Stuurman de website traag en onoverzichtelijk worden.

3.2.2 Interview met Sven Dinslage

In het interview met Dhr. Dinslage zijn er zeer verschillende onderwerpen besproken en hierbij zijn er een aantal zaken boven komen drijven. Ten eerste werd het duidelijk dat Evident als full service internet bureau nog relatief weinig ervaring heeft met het ontwikkelen van mobiele websites. Hieronder vallen zowel de aparte mobiele websites als de responsive websites.

Een verschijnsel dat hier aan bijdraagt, is het feit dat de ontwikkeling van een mobiele website nog niet standaard is. Met standaard wordt bedoeld dat het gegeven 'mobiel' geen vanzelfsprekende vereiste voor een nieuwe website is. Steeds vaker wordt uiteindelijk wel een mobiele website ontwikkeld, maar dit vindt vaak pas plaats nadat de ontwikkeling van de desktop website al is afgerond. Dit is om meerdere redenen nadelig. Hieronder een aantal voordelen van het ontwikkelen van een mobiele website gelijktijdig met een desktop website:

- Mogelijkheid tot het gebruik van een responsive layout wordt sterk vergroot
- Mobile First filosofie kan worden toegepast waardoor gemakkelijker wordt een multi-device website op te zetten

Een andere invalshoek die Dinslage opperde is dat de ontwikkeling van een responsive website niet hoeft te beginnen bij concept of design. Omdat de mogelijkheden en de onmogelijkheden voornamelijk kennis is die 'front-end developers' bezitten, kan dit het beste toegepast worden wanneer de afdeling front-end met het design aan de slag gaat. Op deze manier loop je niet tegen gemiste mogelijkheden en vervelende onmogelijkheden aan.

3.2.3 Interview met Marc Bruisten

In het interview met Marc Bruisten worden een aantal zaken onderstreept die al in de voorgaande interviews besproken zijn. Zo benadrukt Bruisten dat het op het moment nog niet zo ver is dat een mobiele website als vanzelfsprekend beschouwd wordt binnen Evident. Dit is wel iets waar Evident naar toe wil groeien. Het plan is om aankomende projecten standaard te voorzien van 'responsiveness'. Hiermee wordt bedoeld dat iedere nieuwe website zo opgebouwd wordt dat deze zich automatisch aanpast aan de schermgrootte. De Mobile First filosofie wordt door Bruisten wel gewaardeerd, maar hij voorspeld dat deze manier van werken niet zo gemakkelijk is te implementeren binnen Evident. De reden hierachter is dat voor het implementeren van deze werkwijze alle processen drastisch veranderd dienen te worden. Deze omslag zal op sommige afdelingen binnen de organisatie en mogelijk voor de klant lastig te realiseren zijn.

Een van de struikelblokken die Evident op het moment ondervind bij het opzetten van responsive websites is dat niet iedere layout zich even gemakkelijk leent om omgezet te worden naar een responsive website. Bruisten geeft hierbij wel aan dat dit zich alleen voordoet bij websites waar de desktop website al voor gerealiseerd is waardoor in een later stadium de responsiviteit toegevoegd

moet worden. Hierbij is er dus geen rekening gehouden met mobiele apparaten in de concept en design fase waardoor het lastiger wordt een responsive website te bouwen. Hieraan kan je zien dat het succesvol implementeren van een responsive website al begint bij het concept/interactie-ontwerp.

Een ander struikelblok volgens Bruisten is de laadtijd van de website. Omdat er eenzelfde website wordt gebruikt bij het bekijken van een responsive website op een mobiel apparaat worden dezelfde bronnen gebruikt als bij een desktop website (een uitzondering hierop zijn achtergrondafbeeldingen). Aangezien mobiele apparaten vaak via een tragere verbinding verbonden zijn met het internet kan het soms lang duren voordat een webpagina geladen is als deze niet geoptimaliseerd is voor mobiele apparaten. Afbeeldingen zijn de grootste boosdoeners in verband met de relatief grote bestandsomvang.

Bruisten geeft aan dat de reden voor de migratie van een aparte mobiele website naar een responsive website te maken heeft met de term 'One Web'. Met One web wordt het volgende bedoeld:

> *Making, as far as is reasonable, the same information and services available to users irrespective of the device they are using.*

Vrij vertaald wordt hiermee bedoeld dat tot zo ver het redelijk is dezelfde informatie beschikbaar moet zijn onafhankelijk van in welke context de website opgevraagd wordt.

3.2.4 Interview met Joost Verweij

In het interview met Joost Verweij is het perspectief verschoven naar functioneel ontwerper. Verweij is een groot voorstander van het principe 'Mobile First', maar hij maakt duidelijk dat het soms ook alleen bij het principe blijft. Soms kan de theorie niet in zijn geheel opgaan door de situatie. Hiermee wordt bedoeld dat alles vanuit 'mobile' geredeneerd moet worden met als gevolg dat alle besluiten zich door vertalen naar grotere schermen. Dit heeft als gevolg dat alle content gelijk is over alle verschillende devices. Dit verschijnsel is niet altijd goed toepasbaar bij ieder project.

Verweij geeft twee redenen waarom er tot nu toe voor een aparte mobiele website gekozen wordt:

- Er is in eerste instantie geen budget voor een mobiele website waardoor dit achteraf gebeurd

- Door de techniek van het veel gebruikte CMS Sitecore is dit ontstaan. In het standaard pakket van het CMS Sitecore is het mogelijk een mobiele website aan te maken welke gebruik maakt van dezelfde content als de 'normale website'. Hierdoor is het erg gemakkelijk een aparte mobiele website te bouwen.

Verweij is ervan overtuigd dat er genoeg kennis en techniek is om responsive webdesign te gaan gebruiken als een standaard binnen Evident. Dit moet gebeuren omdat de verpakking verkocht wordt en niet het systeem. Het moet als standaard ingevoerd worden anders hoeft de klant het waarschijnlijk niet, door het prijskaartje.

3.2.5 Conclusie

Uit deze interviews zijn een aantal conclusies te trekken waarmee rekening gehouden dient te worden bij dit onderzoek. Een van de eerste bevindingen die duidelijk wordt uit deze interviews is dat het in een later stadium implementeren van responsive elementen in een website niet ideaal is. Dit zorgt voor veel complicaties en wordt dan ook als niet ideaal beschouwd. Liever zal er in de eerste fases van de ontwikkeling van een website hier al rekening mee gehouden moeten worden.

Een ander punt dat naar voren is gekomen, is de technische kant. Omdat in het verleden websites nooit op dit soort apparaten heeft hoeven werken, zijn er vele technische struikelblokken waar rekening mee gehouden dient te worden. Deze technische complicaties zullen in het tweede literatuuronderzoek worden onderzocht en geïnventariseerd zodat deze in een later stadium in het onderzoek kunnen worden onderzocht en mogelijk opgelost kunnen worden.

3. Literatuuronderzoek II

In het tweede literatuuronderzoek wordt er naar oplossingen gezocht voor de vastgestelde knelpunten die in de voorgaande onderzoeken aan het licht zijn gekomen. Deze knelpunten zijn onderverdeeld in de volgende onderdelen:

- Device capabiliteit detectie
- Laadtijd optimalisatie
- Media optimalisatie

3.3.1 Device capabiliteit detectie

Device capabiliteit detectie is het detecteren van de verschillende eigenschappen en mogelijkheden van het betreffende device. In dit onderdeel zullen de volgende mogelijkheden besproken worden:

- Media Queries
- Javascript resolutie berekenen
- Javascript cookie
- Server side UA sniffing

Media Queries

Inleiding

De meest geaccepteerde en veelbelovende techniek is het gebruik van media queries. Media queries is een conditionele techniek die toegepast wordt in cascading style sheets (CSS). Met deze techniek kan aan de hand van de context van de gebruiker bepaald worden welke stylesheet gebruikt moet worden tijdens het renderen van de pagina.

Geschiedenis

Het eerste voorstel van deze techniek dateert terug naar begin april 2001. In dit voorstel wordt duidelijk dat dit als uitbreiding bedoeld is op de HTML4 en CSS2 specificatie “media-dependent” (of media afhankelijke) stylesheets. De twee media die hierin werden onderscheden waren ‘screen’ en ‘print’. In de praktijk kwam dit neer op een stylesheet voor beeldschermen en een stylesheet voor printers.

width & height	De exacte waarde van de viewport
max-width & min-width:	De minimale of maximale waarde van de viewport
device-width & device-height	De exacte afmetingen van het scherm van het apparaat
orientation	De oriëntatie van het apparaat: “portrait” of “landscape”

De huidige specificatie is van 27 juli 2010 *(Håkon Wium Lie, 2010)*, deze is voorlopig de definitieve welke in moderne browsers geïmplementeerd is. Het betreft de zogeheten 'Candidate Recommendation'.

Specificaties

De volledige specificatie bevat vele mogelijkheden welke niet interessant zijn in de huidige context. Om deze reden wordt hier de meest relevante specificaties weergegeven. De specificaties zijn opgebouwd op basis van conditionele logica. Dit houdt in dat er voorwaarden gesteld kunnen worden waar aan voldaan moet worden. Zodra aan alle voorwaarden voldaan wordt zal de gespecificeerde actie uitgevoerd worden. Voorbeelden hiervan zijn te vinden onder het kopje 'Mogelijkheden'.

Mogelijkheden

De techniek biedt zeer bruikbare mogelijkheden voor het optimaal weergeven van een website op verschillende apparaten. Zo kan er afhankelijk van de schermgrootte een aparte layout gespecificeerd worden.

Zo kan er bijvoorbeeld voor een mobiel apparaat van minimaal 240 pixels en maximaal 479 pixels breed de stylesheet "mobile.css" toegewezen krijgen terwijl een tablet van minimaal 480 pixels en maximaal 959 pixels breed de stylesheet "tablet.css" toegewezen krijgt. Voor apparaten die een grotere horizontale breedte hebben wordt de desktop stylesheet getoond "desktop.css".

Dit voorbeeld kan gerealiseerd worden door middel van deze code:

```
<link rel="stylesheet" media="screen and (min-width: 240px) and (max-width:479px)" href="mobile.css" />

<link rel="stylesheet" media="screen and (min-width: 480px) and (max-width:959px)" href="tablet.css" />

<link rel="stylesheet" media="screen and (min-width: 960px) " href="desktop.css" />
```

Javascript resolutie berekenen

Het berekenen van de resolutie is alleen client-side mogelijk, dit komt doordat de server weinig informatie te weten krijgt over de bezoeker voordat de pagina gegenereerd wordt. Desondanks kan het berekenen van de schermresolutie via javascript voor vormgeving gerelateerde zaken interessant zijn.

Met behulp van dit stukje javascript code kan de breedte en de hoogte van het beeldscherm berekend worden:

```
<script>

    var width =         screen.width;
    var height =        screen.height;
    var orientation =   window.orientation;

</script>
```

Naast de breedte en de hoogte kan ook de oriëntatie van het beeldscherm bepaald worden. Dit is portret of landschap. Het bekijken van de oriëntatie is van belang omdat de fysieke afmetingen van het scherm niet veranderen op het moment dat het apparaat gekanteld wordt, dit heeft als gevolg dat in de praktijk de hoogte van het scherm in de breedte wordt weergegeven en vice versa. Om hiermee rekening te kunnen houden is de waarde van de oriëntatie interessant.

Met behulp van deze waardes kan er in de javascript code condities worden opgesteld waarmee er besloten kan worden wanneer er wel en wanneer er niet bepaalde acties worden uitgevoerd.

Javascript cookie

Deze techniek is op zichzelf geen techniek die kan bijdragen aan een responsive website. Met deze techniek wordt er namelijk een klein tekst bestand op apparaat van de bezoeker opgeslagen met hierin bepaalde data die verkregen kan worden op een van de mogelijkheden die hiervoor zijn besproken.

```
<script>

        var width =         screen.width;
        var height =        screen.height;
        var orientation =   window.orientation;

        document.cookie = "device_dimensions=" + width + "/" + height + "/" + orientation;

</script>
```

Deze javascript mogelijkheden kunnen dus gebruikt worden om data te verzamelen en deze in een cookie op te slaan. Het nut van een cookie is dat deze op andere momenten en op andere wijzen benaderd en gebruikt kan worden. Hiermee wordt bedoelt dat wanneer een cookie is aangemaakt door middel van javascript er een dag later door middel van een backend programmeertaal als php de informatie uit deze cookie ingelezen kan worden.

Een mogelijke scenario waarin dit is toe te passen is als volgt:

> *Bij het laden van een pagina wordt er een cookie aangemaakt met hierin de schermgrootte en de schermbreedte van het apparaat van de bezoeker. Op het moment dat de pagina de afbeeldingen wil gaan laden die zich in de pagina bevinden wordt er aan de hand van de informatie in de cookie bepaald welke grootte de afbeelding moet hebben.*

In het voorbeeld wordt een cookie aangemaakt met de naam “device_dimensions”. De cookie wordt gevuld met de breedte en hoogte informatie. Dit is een voorbeeld van de aangemaakte cookie:

> *device_dimensions = “320/480/landscape”;*

Server-side UA sniffing
Bij het gebruik van deze techniek wordt de “User Agent String” die bij het laden door de browser van de bezoeker naar de webserver wordt gestuurd geanalyseerd. Deze analyse moet uitwijzen of het een mobiel apparaat is of niet.

Hieronder een voorbeeld van een user agent string van een iPhone 4 met iOS5 geïnstalleerd:

```
Mozilla/5.0 (iPhone; CPU iPhone OS 5_0_1 like Mac OS X) AppleWebKit/534.46 (KHTML, like
Gecko) Version/5.1 Mobile/9A405 Safari/7534.48.3
```

In dit voorbeeld kan er op het fragment ‘iPhone’ en ‘Mobile’ gefilterd worden waarmee wordt vastgesteld dat de bezoeker met deze UA string de mobiele website krijgt aangeboden.

Om zo accuraat mogelijk te werken bestaan er verschillende verzamelingen van fragmenten van UA strings waarmee mobiele apparaten zoals mobiele telefoons en tablets herkend kunnen worden.

Een uitgebreide open source verzameling is beschikbaar op http://detectmobilebrowsers.com/. Deze verzameling is te gebruiken in 16 verschillende programmeertalen of webservers.

Een groot voordeel om server-side de UA string uit te lezen is dat het mogelijk wordt de bezoeker te identificeren voordat de pagina opgebouwd wordt door de back-end code. Server-side UI sniffing kan op twee manieren uitgevoerd worden:

1. Via de webserver
2. Via backend programmeercode
3. Via Combinatie webserver en programmeercode

Webserver
Via de webserver kan de useragent string uitgelezen worden waarna er een actie ondernomen kan worden. Er kan bijvoorbeeld een redirect worden uitgevoerd, een rewrite worden uitgevoerd of een variabele aangemaakt worden. Deze drie voorbeelden zijn de meest interessante bij het gebruik van de useragent string om devices te detecteren.

- **Redirect**
  Een veel gebruikte techniek is om na het detecteren van het device via de user agent string een redirect uit te voeren naar een subdomein (bijvoorbeeld: van http://website.nl naar -> http://**m**.website.nl/). Dit zorgt ervoor dat er twee aparte websites moeten worden gemaakt. Hierdoor is deze methode niet geschikt.

- **Rewrite**
  Een rewrite is het herschijven van een URL. Dit betekend dat de URL http://website.nl/cat/pagina in werkelijkheid er als volgt uit ziet: http://website.nl/?cat=cat&p=pagina. De eerste URL is fictief en wordt gekoppeld aan de werkelijke URL. Deze herschrijving kan gemanipuleerd worden op basis van de useragent string detectie. Wanneer de UA string uitwijst dat de bezoeker via een mobiel device de website bezoekt kan de eerste URL gekoppeld worden aan een andere URL. Bijvoorbeeld: http://website.nl/?cat=cat&p=pagina**&mobile=true**

Backend programmeercode
Ook via backend programmeercode kan de UA string worden uitgelezen. Deze methode werkt op nagenoeg dezelfde manier als via de webserver. Het grote verschil zet hem erin dat programmeercode meer vrijheid biedt dan en webserver. De webserver biedt beperkte mogelijkheden maar werkt daarentegen wel sneller en effectiever.

Via programmeercode kan er bijvoorbeeld specifiek per onderdeel van de website keuzes worden gemaakt aan de hand van de UA string. Zo kunnen er bijvoorbeeld bepaalde onderdelen anders opgebouwd of zelfs weggelaten worden voor bepaalde devices zodat dit de gebruikerservaring ten goede komt.

Combinatie webserver en programmeercode
Een mooie oplossing om het beste van beide werelden te ervaren is door de technieken te combineren. Via de webserver zal een variabele kunnen aangemaakt worden welke te gebruiken is via de programmeercode. Hierdoor kan er via de webserver snel worden bepaald om wat voor een device het gaat, er wordt dan een waarde aan een variabele toegekend welke vervolgens in de programmeercode ingelezen kan worden.

3.3.2 Laadtijd optimalisatie
In dit onderdeel kijken we naar de laadtijd optimalisatie. Het doel van dit onderdeel is er voor te zorgen dat de website zo snel mogelijk geladen wordt en zichtbaar is voor de bezoeker. Dit onderdeel bestaat uit twee technieken:

- Script optimalisatie & compressie
- Bestanden combineren

Script optimalisatie & compressie
Script optimalisatie en compressie kan met verschillende tools worden uitgevoerd.

Script optimalisatie
Een van de meest gebruikte en efficiënte script optimalisatie tools is closure. Closure is ontwikkeld door Google om javascript te optimaliseren. Via de URL: http://closure-compiler.appspot.com/home

is de webversie van closure te vinden. Via deze webinterface kan er direct code of javascript bestanden inladen worden. Vervolgens kunnen de bestanden of code er in verschillende maten geoptimaliseerd en / of herschreven worden. In theorie optimaliseert deze service de code zo dat deze sneller uitgevoerd kan worden door de browser.

Script compressie
Script compressie kan wederom met verschillende tools worden uitgevoerd. De meest bekende en betrouwbare is de compressor van Yahoo: YUI compressor. Deze compressor is te vinden op: http://yui.2clics.net/. Via deze website is het mogelijk javascript bestanden up te loaden en de gecomprimeerde bestanden te downloaden wanneer deze verwerkt zijn.

Bestanden combineren
Om de prestaties verder te verbeteren dienen alle mogelijke scripts, stylesheets en afbeeldingen samengevoegd te worden in één enkel bestand per type. Dit betekend dat alle scripts in één bestand samengevoegd moeten worden.

Scripts
Met de eerder genoemde tools is het mogelijk om bestanden te combineren tot één enkel bestand.

Stylesheets
Ook stylesheets kunnen en dienen gecombineerd te worden tot één bestand.

Afbeeldingen
Voor afbeeldingen heeft dit een speciale naam, namelijk 'sprites'. Een sprite is dus een samenvoeging van meerdere afbeeldingen. Door middel van CSS kan er per element een andere achtergrond toegekend worden door hetzelfde bestand te gebruiken maar hierin een andere positie in te gebruiken.

Dit zorgt ervoor dat er maar één aanvraag aan de server hoeft worden gedaan waardoor dit een grootte snelheidswinst. In de totale bestandsomvang zal het combineren van afbeeldingen ook een gunstig resultaat bieden.

Een voorbeeld van een spite is hiernaast te vinden. Deze sprite is afkomstig van google.nl. In deze sprite zijn verschillende icoontjes, buttons en logo's te vinden die op verschillende pagina's van de website hergebruikt worden.

Figuur 7: Google sprite (Google, 2012)

3.4 Expertonderzoek

In het expertonderzoek zullen de gevonden technieken beoordeeld worden. Hiernaast zal er een selectie gemaakt worden van technieken en mogelijkheden welke geschikt zijn voor gebruik binnen Evident.

3.4.1 Device capabiliteit detectie

Media Queries

Hier wordt de techniek media queries kritisch beoordeeld. De voor en de nadelen zullen worden besproken en later in dit hoofdstuk zal bepaald worden of deze techniek voldoet aan de eisen en interessant is voor Evident om te gebruiken.

Voordelen

- Layout kan gemakkelijk aangepast worden aan het apparaat:
  Omdat de techniek geen beperkingen legt op het gebruik van media queries kan er ieder denkbaar formaat apparaat een aangepaste stijl gemaakt worden. Dit zorgt ervoor dat het mogelijk wordt praktisch iedere apparaat een optimale beleving te bieden.

- Snelheid:
  Doordat de techniek via CSS werkt, kan het snel uitgevoerd worden door de browser waardoor er geen vertraging optreed bij het laden van de website.

- Browser compatibiliteit:
  De browser compatibiliteit is zeer goed wanneer er gebruik wordt gemaakt van een polyfill. Een polyfill is een script dat de missende functionaliteit, in dit geval media queries, toevoegd aan de browser. Met deze polyfill wordt de compatibiliteit uitgebreid met IE6 t/m 8. Uiteindelijk is de ondersteuning dus: IE6+, FF2+, Chrome 5+, Opera 10+

- Responsive achtergrond afbeeldingen:
  Doordat deze techniek via CSS werkt wordt het mogelijk bepaalde achtergrond afbeeldingen niet te laden of te vervangen door geoptimaliseerde afbeeldingen.

Nadelen

- Alleen schermgrootte wordt bekeken:
  Omdat alleen de schermgrootte kan worden ontvangen geeft dit niet altijd de accurate situatie weer. Op het moment van schrijven is dit nog niet van toepassing, maar het is waarschijnlijk dat er in de toekomst tablets op de markt komen welke een grotere schermgrootte hebben dan de kleinere monitoren. Hierdoor kan de tablet niet onderscheden worden van een desktop met een klein scherm bij het gebruik van minimale en maximale schermgrootte-waardes.

  Dit probleem zou mogelijk wel opgelost kunnen worden wanneer er gebruik wordt gemaakt

van exacte device-width waardes die specifiek gericht zijn op mobiele apparaten met een hoge resolutie.

Javascript resolutie berekenen
Hier wordt de techniek javascript resolutie berekenen kritisch beoordeeld. De voor en de nadelen zullen worden besproken en later in dit hoofdstuk zal bepaald worden of deze techniek voldoet aan de eisen en interessant is voor Evident om te gebruiken.

Voordelen

- Exacte afmetingen:
  De exacte afmetingen van het betreffende scherm kan worden uitgelezen. Dit heeft als voordeel dat er precieze beslissingen kunnen worden genomen waarmee er een optimaal resultaat behaald kan worden. Een voorbeeld hiervan zijn afbeeldingen. Afbeeldingen kunnen tot op de pixel nauwkeurig verkleind worden waardoor de bestandsgrootte en hiermee de laadtijd maximaal verlaagd kan worden.

Nadelen

- Geen kennis over soort apparaat:
  Door het berekenen van de schermgrootte kan er op basis van deze informatie niet vastgesteld worden welk of van welk soort het apparaat is. Dit is handig om te weten omdat het niet altijd af te lezen is aan de schermgrootte of het om een mobiel apparaat gaat.

  In sommige gevallen is er wel een goede inschatting te maken. Denk hierbij bijvoorbeeld aan een schermgrootte van 320 bij 480 pixels, in dit geval kan er van uitgaan worden dat dit geen desktop computer betreft maar een mobiele telefoon. Nog een stap verder zou zijn dat er een grote kans is dat dit apparaat een iPhone betreft, dit is te herleiden aan de exacte afmetingen die aan deze telefoon gelijk is.

Javascript cookie
Hier wordt de techniek javascript cookie kritisch beoordeeld. De voor en de nadelen zullen worden besproken en later in dit hoofdstuk zal bepaald worden of deze techniek voldoet aan de eisen en interessant is voor Evident om te gebruiken.

Voordelen

- Resultaat uit client-side code beschikbaar:
  Het resultaat dat voortkomt uit client-side code wordt weggeschreven in een cookie waardoor de server-side code hier gebruik van kan maken. Dit zorgt ervoor dat er veel meer informatie over de bezoeker kan worden vastgesteld waarvan gebruik kan worden gemaakt door de server.

Nadelen

- Door de recente cookie-wet wordt het gebruik van zogenaamde ‘third party cookies’ erg lastig gemaakt *(Haes, 2012)*. Third party cookies zijn cookies die door een derde partij aangemaakt zijn. In de praktijk betekend dit dat de cookies die doormiddel van de javascript cookie techniek worden aangemaakt alleen door dezelfde website mag worden gebruikt. De cookie techniek gebruiken in combinatie met bijvoorbeeld een mediaserver is dus niet mogelijk.

Server-side UA sniffing

Hier wordt de techniek media queries kritisch beoordeeld. De voor en de nadelen zullen worden besproken en later in dit hoofdstuk zal bepaald worden of deze techniek voldoet aan de eisen en interessant is voor Evident om te gebruiken.

Voordelen

- Het uitlezen van de user agent string via de server heeft als voordeel dat dit kan gebeuren voordat de pagina laadt. Hierdoor kan er bijvoorbeeld aan de hand van de UA string bepaald worden welke javascript bestanden er ingeladen moeten worden.

- Lichtgewicht apparaat detectie:
  Doordat deze techniek zoekt naar een gedeeltelijke match in de user agent string is het een snelle en lichte methode.

- Futureproof:
  Doordat er gedeeltes worden getoetst ten opzichte van de user agent string en niet de gehele string wordt de kans sterk vergroot dat toekomstige apparaten ook goed herkend worden.

- Weinig onderhoud nodig:
  Doordat de techniek future proof is heeft dit tot gevolg dat de lijst maar af en toe geüpdatet hoeft te worden.

Nadelen

- Onderhoud is noodzakelijk:
  Ondanks dat er weinig onderhoud nodig is, moet het op zijn tijd wel gebeuren. Hiermee is deze techniek niet future friendly *(Wroblewski, Future Friendly, 2011)*.

3.4.2 Laadtijd optimalisatie

Script optimalisatie & compressie
Hier wordt de techniek script optimalisatie & compressie kritisch beoordeeld. De voor en de nadelen zullen worden besproken en later in dit hoofdstuk zal bepaald worden of deze techniek voldoet aan de eisen en interessant is voor Evident om te gebruiken.

Voordelen
- Sneller:
  Door de kleinere bestandsgrootte kan na het toepassen van deze techniek het bestand sneller ingeladen worden door de gebruiker.

- Dit kan server-side geautomatiseerd worden waardoor hier geen inspanning voor geleverd hoeft te worden.

Nadelen
- Geen agressieve optimalisatie:
  Door agressieve optimalisatie en compressie kan het voorkomen dat de gegenereerde bestanden fouten bevatten waardoor er onderdelen van de website minder goed of niet meer werken.

Bestanden combineren
Hier wordt de techniek bestanden combineren kritisch beoordeeld. De voor en de nadelen zullen worden besproken en later in dit hoofdstuk zal bepaald worden of deze techniek voldoet aan de eisen en interessant is voor Evident om te gebruiken.

Voordelen
- Verbeterde laadtijd:
  Net als het bestanden comprimeren en optimaliseren draagt het combineren van soortgelijke bestanden bij aan het verbeteren van de laadtijd voor de bezoeker. Met deze techniek wordt de bestandsgrootte niet kleiner maar het aantal bestanden neemt af.

  Dit is bevorderlijk voor de laadtijd aangezien het aantal aanvragen aan de server afneemt. Dit heeft twee redenen:

  - Deze aanvragen nemen onafhankelijk van de bestandsgrootte een vaste connectietijd in. Als er vele bestanden gedownload moeten worden kan deze vaste connectietijd hoog oplopen. Door het samenvoegen tot één bestand wordt deze vaste connectietijd teruggedrongen.

  - Iedere browser heeft een maximum aantal connecties. In onderstaand tabel is een overzicht te vinden van verschillende browsers en het maximum aantal connecties.

Browser	HTTP/1.1	HTTP/1.0
IE 6,7	2	4
IE 8	6	6
Firefox 2	2	8
Firefox 3	6	6
Safari 3,4	4	4
Chrome 1,2	6	?
Chrome 3	4	4
Chrome 4+	6	?
iPhone 2	4	?
iPhone 3	6	?
iPhone 4	4	?
Opera 9.63,10.00alpha	4	4
Opera 10.51+	8	?

Figuur 8: maximum parallelle connecties (Sounders, 2008)

Dit betekend dat er een maximum is aan het gelijktijdig downloads waardoor er een wachtrij kan ontstaan bij meerdere bestanden. Dit zorgt voor een langzaam ladende pagina.

- Schonere html code:
  De HTML code wordt schoner doordat er minder bestanden vanuit deze code aangeroepen hoeft te worden. Dit zorgt voor een kleinere pagina wat hierop voor een kleine snelheidswinst zorgt.

Nadelen

- Kans op fouten:
  Bij het combineren van javascript bestanden bestaat er de mogelijkheid dat er compatibiliteitsfouten optreden.

Conclusie

Om duidelijk weer te geven welke technieken interessant zijn om te gebruiken is het volgende schema opgezet (zie hieronder). Dit schema geeft weer welke technieken wel, niet of misschien gebruikt mogen worden.

	Wel	Niet	Misschien?	
Media Queries	X			
Javascript resolutie	X			
Javascript cookie			X	Alleen gebruiken wanneer noodzakelijk
Server-side UA		X		
Script optimalisatie	X			
Bestanden combineren	X			

Figuur 9: overzicht mogelijkheden

De verklaring per techniek zal hier worden toegelicht:

- Media Queries:
  Media queries zijn tot op zekere hoogte een goede manier om apparaten van verschillende schermgroottes te onderscheiden en te bedienen. Hierbij zit de beperking in het feit dat de enige manier om een mobiel apparaat te herkennen op basis van schermgrootte wordt uitgevoerd. Dit gaat voor mobiele telefoons tot nu toe nog goed maar voor tablets gaat dit samenlopen met desktop beeldschermen.

- Javascript resolutie:
  De exacte afmetingen van het scherm van de bezoeker berekenen is op het moment alleen mogelijk via Javascript. Dit maakt het een erg sterk hulpmiddel op elementen in de layout van een webpagina zo flexibel mogelijk te kunnen maken. Verder voldoet deze techniek aan de gestelde voorwaarden waarmee deze techniek als geschikt beschouwd kan worden.

  Deze techniek zal in combinatie met andere technieken voor interessante mogelijkheden kunnen zorgen wat betreft layout flexibiliteit en media optimalisatie.

- Javascript cookie:
  Wanneer het van pas komt, zou deze techniek prima oplossing kunnen zijn. Anders zou er gebruik gemaakt moeten worden van een omweg waarbij de cookie niet aangemaakt wordt door de javascript zelf maar door de betreffende server doormiddel van een aanvraag naar deze server in de vorm van een afbeelding of javascript bestand. Deze techniek is op een vergelijkbare manier toegepast in de ontwikkelde media server *(Zijp, Media Server, 2012)*.

- Server-side UA sniffing:
  Ondanks het feit dat de techniek waarschijnlijk nog enige tijd goed zou kunnen blijven werken is deze niet future friendly. Hierdoor is deze techniek niet geschikt als oplossing voor device detectie in dit onderzoek.

- Script optimalisatie & compressie:
  Doordat er met relatief weinig inspanning een compressie met een bestandsgrootteverlies van gemiddeld 21% *(Yahoo, 2011)* behaald kan worden is dit een geschikte techniek de laadtijd van een multi-device website te verhogen. De enige voorwaarde waardoor de techniek betrouwbaar blijft is dat er geen vorm van agressieve compressie of optimalisatie wordt toegepast.

- Bestanden combineren:
  Ook deze methode kan goed en gemakkelijk bedragen aan een snelheidswinst op laadtijd gebied. De voorwaarde die hier aan zit is dat het grondig getest moet worden voordat deze techniek in een live omgeving geïmplementeerd kan worden.

4. Realisatie

Uit literatuuronderzoek I zijn er nog een tweetal zaken waarvoor nog geen oplossing is gevonden. Deze struikelblokken zijn:

- Media optimalisatie
- Conditioneel laden

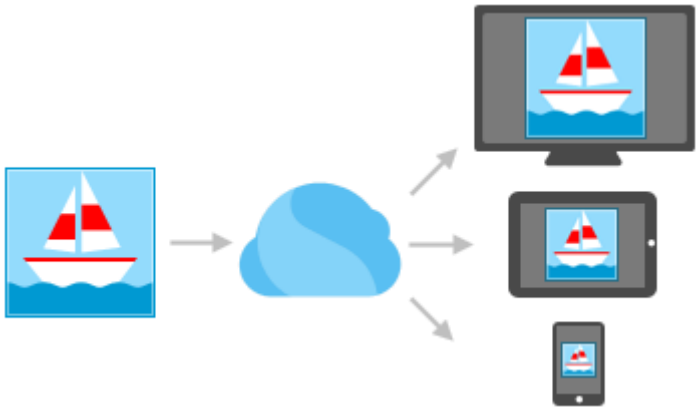
Voor deze twee struikelblokken zijn twee maatwerk oplossingen ontwikkeld. Voor de media optimalisatie is een media server ontwikkeld in PHP en javascript en voor het conditioneel laden is een conditional script loader geschreven.

Hiernaast is er een prototype opgesteld waarin de geschikte technieken en mogelijkheden worden weergegeven. Dit prototype dient als een demonstratie voor Evident.

4.1 Media optimalisatie

Er is een raamwerk ontwikkeld dat als media optimalisatie server ingezet kan worden. Deze server kan dan voor meerdere websites afbeeldingen schalen, optimaliseren en device afhankelijk opleveren aan de gebruiker.

Hoe het in zijn werkt gaat, wordt door onderstaand figuur goed weergegeven.



Figuur 10: Werking media optimalisatie server

Er wordt een afbeelding aangeleverd door de betreffende website. De media server bekijkt per bezoek om wat voor een device het gaat. Aan de hand van deze informatie wordt er een gepaste afbeelding aan de bezoeker aangeboden.

Het raamwerk kan op twee manieren besluiten welke afbeelding opgeleverd moet worden. Als eerste wordt er geprobeerd om een cookie uit te lezen en aan de hand van deze informatie een afbeelding op te leveren. Deze eerste methode is de meest betrouwbare methode op het gebied van nauwkeurigheid. De mogelijkheid bestaat echter dat de cookie niet of niet op tijd wordt aangemaakt waardoor deze methode niet werkt. Als terugval wordt er op dat moment gebruik gemaakt van methode twee. Dit is apparaat detectie. Via de User-Agent String wordt bepaald op wat voor een device het gaat (desktop, tablet, mobile). Als deze bepaald is wordt er een geschikte afbeelding opgeleverd.

In onderstaand tabel wordt weergegeven welke formaten er wanneer en hoe worden aangeleverd:

	Cookie Detectie	UA String Detectie
Mobile	320, 480, 768 of 900 pixels	320 pixels breed
Tablet	320, 480, 768 of 900 pixels	480 pixels breed
Desktop	Alleen afbeelding optimalisatie	Alleen afbeelding optimalisatie

Figuur 11: Overzicht formaten (Zijp, Media Server, 2012)

Cookie Detectie
De cookie detectie wordt aangemaakt door middel van een javascript bestand. Dit bestand berekend de scherm grootte, breedte, en oriëntatie. Vervolgens wordt er aan de hand hiervan een keuze gemaakt in hoe breed de opgeleverde afbeelding moet zijn. Deze kan 320, 480, 768 of 900 pixels zijn. Wanneer de breedte van het scherm groter is dan 900 pixels wordt de afbeelding niet verkleind maar wel geoptimaliseerd.

Om veiligheidsredenen is het niet mogelijk de cookie vanuit dit script te creëren. Daarom wordt er door het script een aanvraag gedaan naar de mediaserver welke vervolgens die cookie aanmaakt met hierin de berekende breedte.

User Agent String Detectie
De user agent string detectie gaat direct via de media server. Deze bekijkt of er een match wordt gevonden met een van de bekende devices. Als dit het geval is wordt er vanuit gegaan dat het om een mobiel device gaat (mobiel en tablet worden onderscheiden). Als er geen match kan worden gemaakt wordt er vanuit gegaan dat het om een desktop computer gaat.

Media Server
De media server gaat vervolgens bekijken of er een cookie aanwezig is. Wanneer dit het geval is wordt de cookie gebruikt om de passende afbeelding te creëren. Zo niet dan wordt dit gedaan door middel van de User Agent String. De afbeelding wordt dan geoptimaliseerd en mogelijk verkleind naar de gewenste afmetingen.

Optimalisatie
Naast het schalen van de afbeelding vindt er dus standaard een optimalisatie van de afbeelding plaats. Deze optimalisatie wordt verzorgt door de GD library *(Joye, 2011)*. Dit stukje software is een verzameling van functies die te gebruiken zijn om afbeeldingen mee te maken, manipuleren en optimaliseren.

Net als de manipulatie die plaats vindt bij het verkleinen van de afbeelding wordt door deze software de optimalisatie verzorgt. De bestandsgrootte wordt maximaal verkleint door alle onnodige bytes uit het bestand te verwijderen. De kwaliteit van de afbeelding blijft dus hetzelfde terwijl de download snelheid verlaagt zal worden.

Voorbeeld
In onderstaand voorbeeld is een afbeelding te zien welke alleen geoptimaliseerd. Dit is gedaan om de vergelijking te maken tussen een niet en een wel geoptimaliseerde afbeelding.

Figuur 12: geoptimaliseerde afbeelding: http://jlab.nl/http://www.evident.nl/~/media/Images/Evident/Carrousel/Utrecht.ashx

In onderstaande tabel is te zien dat de bestandsgrootte en de uiteindelijke laadtijd van de geoptimaliseerde afbeelding duidelijk verbeterd is. Zo is de bestandsgrootte vier maal kleiner en is de laadtijd twee maal sneller. Dit zal voor de mobiele bezoeker bevorderlijk zijn in de vorm van besparing in dataverkeer en in een snellere gebruikerservaring.

	Originele afbeelding	Geoptimaliseerde afbeelding
Bestandsgrootte	343 KB	**82 KB**
Connectietijd	**35 ms**	48 ms
Downloadtijd	267 ms	**103 ms**
Totale laadtijd	302 ms	**151 ms**

Figuur 13: vergelijking media server (Zijp, Media Server, 2012)

Gebruik
Hier volgt een voorbeeld van het gebruik van de mediaserver in een simpele html pagina.

```
<img src =
"http://jlab.nl/http://www.evident.nl/~/media/Images/Evident/Carrousel/Utrecht.ashx" alt =
"Utrecht" />
```

Let vooral op de dikgedrukte gedeelte voor de URL. De media server draait op de domeinnaam ‘http://jlab.nl’. De originele afbeelding is beschikbaar onder ‘http://www.evident.nl/~...trecht.ashx’. Deze URL voor voorgevoegd door ‘http://jlab.nl/’ waardoor van de media server gebruik kan worden gemaakt.

Voordelen

- Een perfecte afbeelding:
  De afbeelding wordt tot een perfect passende afmeting geschaald. Dit zorgt voor een maximale afname in bestandsgrootte.

- Afbeelding optimalisatie:
  Naast het schalen van de afbeelding wordt deze dus ook geoptimaliseerd. Hierdoor blijft de kwaliteit gelijk terwijl de bestandsgrootte nog verder afneemt.

- Alles geautomatiseerd:
  Voor al deze voordelen hoeft na de implementatie van deze techniek geen extra inspanning geleverd te worden. Alles wordt geautomatiseerd uitgevoerd. Er hoeven dus geen verschillende versies van afmetingen gemaakt te worden.

- Schone HTML code:
  De techniek maakt geen gebruik van aangepaste html maar enkel van het voorvoegen van de afbeelding url. Dit zorgt voor een schone html code.

- Future Friendly:
  Hoewel er gewerkt wordt aan een responsive afbeeldingen oplossing *(WhatWG, 2012)* blijft deze techniek future friendly. Dit heeft te maken met de mogelijkheid om specifieke afmetingen te specificeren. Hierdoor kan er op de volgende manier gewerkt worden wanneer een responsive afbeeldingen specificatie definitief is:

  ```
  <h1><img alt="afbeelding"
          src="jlab.nl/http://test.nl/afbeelding.jpg"
          srcset=" jlab.nl/320/http://test.nl/afbeelding.jpg 320w,
                   jlab.nl/640/http://test.nl/afbeelding.jpg 640w,
                   jlab.nl/1280/http://test.nl/afbeelding.jpg 1280w">
  </h1>
  ```

  Zo kan de ‘src’ tag dienen als terugval voor oudere browsers en de ‘src-set’ tag kan gebruikt worden voor nieuwe browsers waarin een vaste waarde gebruikt kan worden. Zo kan deze techniek in de verre toekomst nog steeds hulp bieden.

Nadelen

- Afbeeldingen onder andere URL:
  De afbeeldingen zijn door het voorvoegen van de afbeelding url beschikbaar geworden onder een ander domein. Dit geeft per definitie geen problemen. Er dient wel een additionele 'DNS Lookup' gedaan te worden. Een DNS lookup is een handeling die de browser uitvoert om te bepalen van welk IP adres de bestanden gehaald moeten worden. Dit gebeurd door een DNS server welke een domeinnaam omzet naar een IP adres. Deze extra DNS lookup zorgt voor een langere laadtijd van de afbeelding.

Conclusie

De media server blijkt een goede techniek op media geoptimaliseerd aan te leveren. Het enige nadeel aan deze techniek kan opgelost worden door middel van een DNS wijziging waarin een lokale URL gebruikt kan worden voor de media server.

De media server zou een goede prestatie bevordering kunnen bieden voor devices met kleinere beeldschermen en/of devices met een slechtere internetverbinding.

4.2 Conditionele Javascript lader

Door javascript conditioneel in te laden kan er voor gezorgd worden dat er geen overbodige bestanden worden geladen door de bezoeker. Deze bestanden worden onnodig gedownload en hiermee wordt de laadtijd van de pagina onnodig verlengt. Een goede oplossing is dus om javascript conditioneel in te laden. Met conditioneel wordt bedoeld: afhankelijk van welke conditie de bezoeker zich bevind wordt besloten welke javascript bestanden er worden ingeladen. Deze conditie is de schermgrootte van het device van de bezoeker.

Met behulp van een bestaand script Head JS is er een conditionele lader gebouwd welke op basis van de schermgrootte bepaald welke bestanden er ingeladen moeten worden *(Piirainen, 2011)*. Er is rekening gehouden met de mogelijkheid dat bepaalde bestanden altijd ingeladen moeten worden en in een bepaalde volgorde.

Hieronder bevindt zich de javascript code welke de conditionele lader vormt:

```
<script src="head.load.min.js"></script>
<script>
    var width = screen.width, height = screen.height, orientation = window.orientation,
    mobile = void 0 != orientation, allsources, basesources, aftersources, sources;

    if(90 == orientation || -90 == orientation) {
      width = -(height = (width += height) - height) + width
    }

    basesources = ["//ajax.googleapis.com/ajax/libs/jquery/1.7.2/jquery.min.js"];
    aftersources = ["scripts.js"];

    mobile ? mobile && 900 >= width && (
        document.writeln("mobile sources loaded"),
        sources = [] //MOBILE ONLY SOURCES
    ) : (
        document.writeln("desktop sources loaded"),
        sources = [] //DESKTOP ONLY SOURCES
    );

    allsources = basesources.concat(sources, aftersources);
    head.js.apply(this || window, allsources);

    console.log(allsources); //LOGGING ALL SOURCES
</script>
```

Figuur 14: Javascript code conditionele lader (Zijp, Load, 2012)

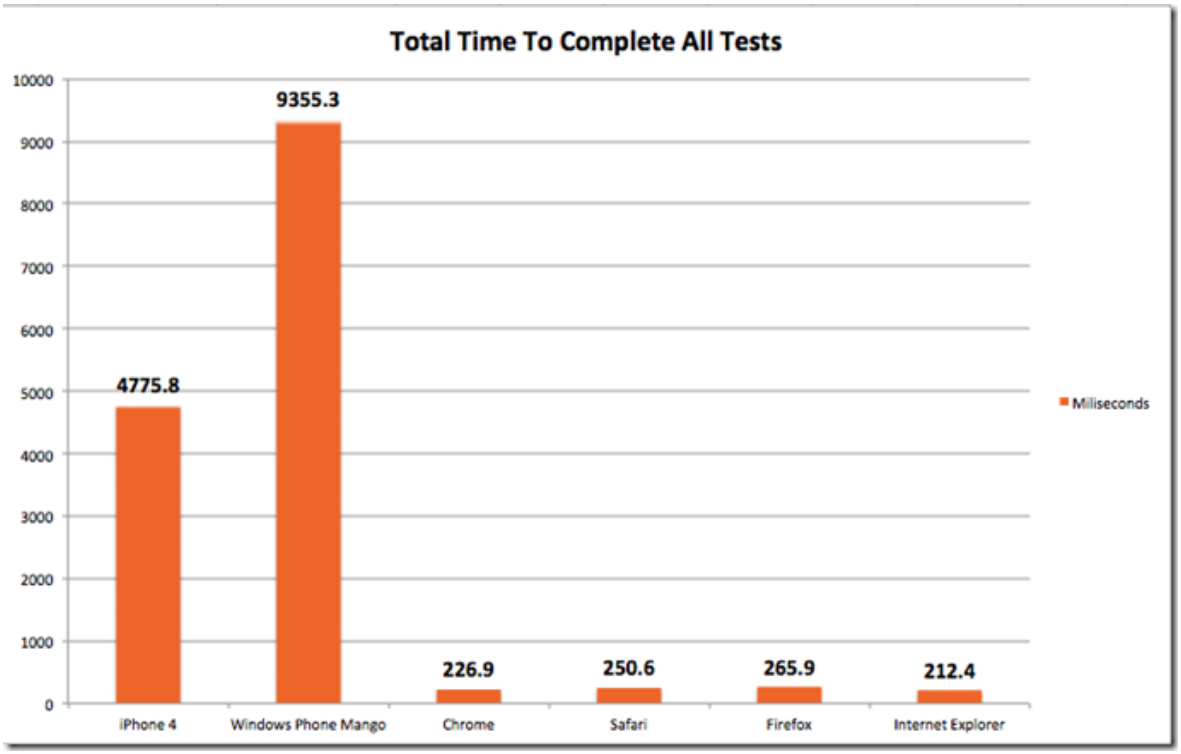
Het volgende vindt plaats:

- Head JS wordt ingeladen
- Alle benodigde variabele worden aangemaakt
- Als er een horizontale oriëntatie aanwezig is worden de hoogte en de breedte omgedraaid
- Standaard bestanden die ongeacht het device ingeladen dienen te worden, opgesplitst in eerder en later uitvoeren.
- Als de breedte kleiner is dan 900 pixels dan worden de mobiele bestanden aan de sources array toegevoegd, anders de desktop bestanden.
- Alle arrays worden samengevoegd
- De uiteindelijke bestanden worden ingeladen door Head JS
- De lijst met de bestanden wordt gelogd

Een online voorbeeld is te vinden via: http://jlab.nl/load *(Zijp, Load, 2012)*

Voordelen

- Sneller geladen:
  Door het conditioneel inladen van javascript bestanden worden alleen de nodige bestanden gedownload en gebruikt. Dit heeft als voordeel van het hardware-matig gezien geoptimaliseerd wordt. Veel onnodige javascript code kan namelijk voor een tragere ervaring zorgen. In onderstaand figuur is te zien hoe lang de verschillende browsers doen over het uitvoeren van de javascript tests.



Figuur 15: Javascript prestatie per browser (KendoUI, 2011)

De Windows Phone doet er wel meer dan 40 maal zo lang over dan Google's browser Chrome. Overbodige javascript code kan hierdoor dus voor onnodige prestatie vermindering zorgen.

- Sneller download:
  Doormiddel van het conditioneel inladen zullen er minder bestanden nodig zijn voor het weergeven van een pagina. Hierdoor zal de downloadsnelheid van de pagina verbeterd worden.

- Combineren van Javascript is niet nodig:
  Het combineren van javascript bestanden is doormiddel van deze techniek niet nodig. Dit heeft te maken met de manier van inladen welke gebruikt wordt. Deze manier van inladen wordt parallel inladen genoemd. Dit zorgt ervoor dat er geen wachtrij ontstaat bij het inladen van javascript bestanden. Deze bestanden kunnen onafhankelijk van het aantal parallel ingeladen worden.

Nadelen

- FUBC is mogelijk:
  FUBC (flash of unbehaviored content) kan mogelijk optreden. Een FUBC is het verschijnsel dat er een flits optreed van een gebroken webpagina welke binnen een paar milliseconde veranderd naar de werkelijke pagina. Dit verschijnsel ontstaat doordat de javascript bestanden iets later in het proces worden gedownload dan volgens de traditionele manier.

Conclusie

Hoewel er een mogelijkheid is dat er een FUBC ontstaat biedt deze techniek een zeer interessante mogelijkheid om effectief en met snelheidswinst script bestanden in te laden. Hiernaast zijn er mogelijkheden om de FUBC te verminderen of zelfs te verwijderen. Dit kan bijvoorbeeld gedaan worden door bepaalde elementen door middel van CSS te verbergen in plaats van met javascript.

4.3 Prototype

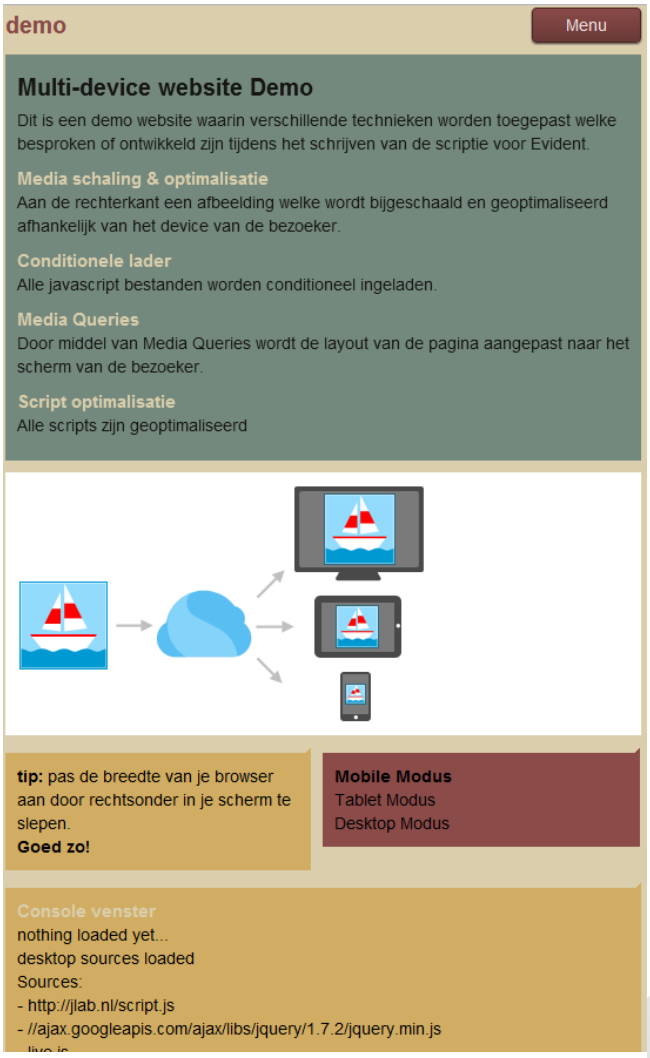
In het prototype zijn een aantal technieken toegepast welke in de vorige hoofdstukken staan beschreven. Het prototype is te bezoeken via: http://jlab.nl/demo *(Zijp, Multi-device website Demo, 2012)*

De volgende twee afbeeldingen zijn schermafdrukken van de demo website, genomen op verschillende resoluties.

De eerste afbeelding is een screenshot op desktop resolutie en de tweede op telefoon resolutie.

In de demo zijn vier technieken toegepast:

- Media schaling & optimalisatie
- Conditionele lader
- Media Queries
- Script optimalisatie



Figuur 16: screenshots demo (Zijp, Multi-device website Demo, 2012)

5. Conclusie

In de voorgaande onderzoeken zijn de volgende stappen ondernomen:

- Identificeren struikelblokken
- Identificeren mogelijkheden
- Vergelijken mogelijkheden
- Ontwikkeling van nieuwe mogelijkheden

Deze vier stappen zijn ondernomen om de eerder geformuleerde deelvragen te beantwoorden. Deze deelvragen worden hieronder nogmaals genoemd en beantwoord.

Deelvragen

- *Wat zijn de complicaties bij het opzetten van een website welke zowel mobiele als niet mobiele apparaten moet kunnen bedienen?*
  De complicaties zijn op te splitsen in drie categorieën.

  - De eerste is device capabiliteit detectie. Dit houdt in dat het detecteren van de mogelijkheden van een device mogelijk zou moeten zijn.

  - De tweede complicatie is laadtijd optimalisatie. Dit is het optimaliseren van de website waardoor de tijd voor het laden van een pagina geminimaliseerd wordt.

  - De derde is media optimalisatie. Dit is de complicatie waarbij de noodzaak voor device geoptimaliseerde media aanwezig is.

- *Welke mogelijkheden zijn er om deze complicaties aan te pakken?*
  Voor de categorieën 'device capabiliteit detectie' en 'laadtijd optimalisatie' zijn er verschillende technieken en mogelijkheden opgesteld. In onderstaand tabel is een overzicht hiervan:

Device capabiliteit detectie	Laadtijd optimalisatie
Media Queries	Script optimalisatie
Javascript resolutie berekenen	Script compressie
Javascript cookie	Bestanden combineren
Server-side UA sniffing	

Figuur 17: overzicht mogelijkheden

Tijdens literatuuronderzoek II werd duidelijk dat er voor twee struikelblokken geen gepaste oplossing gevonden kon worden. Dit waren, conditioneel scripts inladen en media optimalisatie. Om deze reden zijn er voor deze twee struikelblokken twee op maat oplossingen ontwikkeld:

- Media optimalisatie server
- Conditionele javascript lader

- *Welke mogelijkheden zijn toepasbaar binnen Evident? & Hoe verhouden deze technieken zich tot elkaar?*
  Deze twee deelvragen zijn gezamenlijk in het expertonderzoek en gedeeltelijk in het hoofdstuk realisatie behandeld. In onderstaand tabel bevind zich een overzicht van de mogelijkheden met de bijhorende beoordeling.

	Altijd	Nooit	Misschien?
Media Queries	X		
Javascript resolutie	X		
Javascript cookie			X Alleen gebruiken wanneer noodzakelijk
Server-side UA		X	
Script optimalisatie	X		
Bestanden combineren			X Alleen wanneer de conditionele lader niet wordt gebruikt
Javascript conditioneel laden	X		
Media Server	X		

Figuur 18: overzicht beoordeling mogelijkheden

Hoofdvraag
Hier wordt de hoofdvraag beantwoord. De hoofdvraag luidt:

Hoe moet een website technisch opgezet worden om alle devices te kunnen bedienen?

De eindconclusie en het antwoord op de hoofdvraag is uit de uitkomsten van de verschillende onderzoeken te concluderen. De uitkomst van de eerste twee onderzoeken (literatuuronderzoek I en de interviews) was dat er een aantal struikelblokken zijn waarvoor een gepaste oplossing gevonden moest worden. Uit de onderzoeken die hierop volgde bleek er voor bijna ieder struikelblok een geschikte oplossing te zijn. Voor de overige punten is een op maat oplossing ontwikkeld. Door de volledige verzameling van oplossingen voor de struikelblokken is de hoofdvraag als volgt te beantwoorden:

Een multi-device website kan technisch gezien succesvol worden opgezet door de richtlijnen en de technieken toe te passen, welke geformuleerd en ontwikkeld zijn voor deze scriptie.

6. Begrippenlijst

CSS:	Afkorting voor Cascading Style Sheets. Dit is een bestand dat aan een webpagina gekoppeld wordt. In dit bestand kan de opmaak van de webpagina worden beinvloed.
Devices	De verschillende mobiele apparaten en computers. Onder dit begrip vallen zowel mobiele telefoons en tablets als deskstop computers en laptops.
Mobiele devices	Alleen de mobiele varianten van de devices worden bedoeld. Laptops worden hier *niet* in meegenomen.
Multi-device website	Een website welke geoptimaliseerd is om op verschillende devices[1] goed te werken.
GPS	Afgekort voor "Global Positioning System": dit is een techniek wat in moderne mobiele devices is ingebouwd wat gebruikt kan worden om locatievoorzieningen toe te voegen aan een website.
HTTP requests	Een aanvraag van de browser aan de server voor een bepaald bestand. Bijvoorbeeld: afbeelding, css, javascript, pdf, doc.
Mobiele gebruiker	De gebruiker die gebruik maakt van een mobiele device.
Responsive	Responsive of Adaptive webdesign is een term die verwijst naar het automatisch aanpassen van de website aan de schermgrootte van de bezoeker.
Adaptive	Zie responsive.
Tablet	Een mobiele computer die in de hand kan worden gebruikt, over het algemeen is de tablet uitgerust met een touch screen. De tablet kan als een grote, krachtigere pda, of als een kleine laptop worden gezien.
Mobile First	De filosofie (als eerste beschreven door Luke Wroblewski) waarin een nieuwe aanpak van websites ontwikkelen wordt geïdealiseerd. Het denken vanuit een statisch ontwerp verschuift naar een mobile gecentreerde benadering. Meer over mobile first: http://www.lukew.com/resources/mobile_first.asp *(Wroblewski, Mobile First, 2011)*
One Web	Het principe om tot zo ver het redelijk is dezelfde informatie beschikbaar maken onafhankelijk van in welke context de website opgevraagd wordt. Meer informatie: http://www.w3.org/TR/mobile-bp/#OneWeb *(Rabin, 2008)*
Viewport	Het zichtbare gedeelte van de website in de browser.

Future Friendly	Los vertaald toekomst vriendelijk, hiermee wordt bedoeld dat de techniek geen tijdelijke oplossing is maar gericht op de toekomst. http://futurefriend.ly/ *(Wroblewski, Future Friendly, 2011)*
Redirect	De gebruiker wordt doorgestuurd naar een andere pagina.

7. Bibliografie

Android. (2012, 05 24). *Screen support*. Retrieved 05 29, 2012, from Android Developers: http://developer.android.com/guide/practices/screens_support.html

Bender, G. T. (1999). *Touch Screen Performance as a Function of the Duration of Auditory Feedback and Target Size*. Retrieved 04 13, 2012, from thisoldtractor: http://www.thisoldtractor.com/gtbender/papers/dissertation.pdf

Bruce, R. (2012, 03 28). *The Myth of Mobile Content Marketing*. Retrieved 03 29, 2012, from Copyblogger: http://www.copyblogger.com/mobile-content-marketing/

Coyier, C. (2009, 01 28). *Browser Detection is Bad*. Retrieved 04 12, 2012, from CSS-Tricks: http://css-tricks.com/browser-detection-is-bad/

Frost, B. (2012, 03 19). *responsive web design: missing the point*. Retrieved 04 20, 2012, from Brad Frost Web: http://bradfrostweb.com/blog/web/responsive-web-design-missing-the-point/

Gomez. (2011, 07 01). *What Users Want from Mobile*. Retrieved 04 02, 2012, from Gomez.com: http://www.strangeloopnetworks.com/resources/infographics/mobile-infographics/mobile-load-time-vs-user-expectations/

Gustafson, A. (2008, 10 07). *Understanding Progressive Enhancement*. Retrieved 05 04, 2012, from A List Apart: http://www.alistapart.com/articles/understandingprogressiveenhancement/

Haes, A. U. (2012, 05 08). *Finale beslissing over netneutraliteit en cookiewet*. Opgeroepen op 05 16, 2012, van Webwereld: http://webwereld.nl/analyse/110434/finale-beslissing-over-netneutraliteit-en-cookiewet.html

Håkon Wium Lie, T. Ç. (2010, 07 27). *Media queries*. Retrieved 02 14, 2012, from w3.org: http://www.w3.org/TR/css3-mediaqueries/

Joye, P. (2011, 06 17). *gf-libgd*. Retrieved 03 03, 2012, from Bit Bucket: https://bitbucket.org/pierrejoye/gd-libgd

KendoUI. (2011, 10 07). *JavaScript Performance On Mobile Devices*. Retrieved 05 30, 2012, from KendoUI: http://www.kendoui.com/blogs/teamblog/posts/11-10-07/javascript_performance_on_mobile_devices.aspx

King, A. (2002, 04 04). *Anatomy of an HTTP request and correlation to Pagetest legend*. Retrieved 03 18, 2012, from Website Optimization: http://www.websiteoptimization.com/secrets/metrics/10-21-http-request.html

Marquis, M. (2012, 01 31). *Responsive Images: How they Almost Worked and What We Need*. Retrieved 03 05, 2012, from A List Apart: http://www.alistapart.com/articles/responsive-images-how-they-almost-worked-and-what-we-need/

Oh, S. S.-J. (2011, 03 10). *Smart Phones to Generate Half of Total Navigation Sales in 2011* . Retrieved 04 13, 2012, from iSuppli: http://www.isuppli.com/Automotive-Infotainment-and-

Telematics/MarketWatch/Pages/Smart-Phones-to-Generate-Half-of-Total-Navigation-Sales-in-2011.aspx

Piirainen, T. (2011, 05 18). *Head JS*. Retrieved 04 20, 2012, from Head JS: http://headjs.com/

Rabin, J. (2008, 07 29). *Mobile Web Best Practices 1.0*. Retrieved 02 12, 2012, from W3C: http://www.w3.org/TR/mobile-bp/#OneWeb

Sesto, M. E. (2011, 11 22). *Effect of Touch Screen Button Size and Spacing on Touch Characteristics of Users With and Without Disabilities*. Retrieved 23 05, 2012, from Human Factors: http://hfs.sagepub.com/content/54/3/425.abstract

Sights. (2012, 04 01). *HTML5Test.com*. Retrieved 04 08, 2012, from HTML5Test.com : HTML5test.com

Sounders, S. (2008, 03 20). *Roundup on Parallel Connections*. Retrieved 05 29, 2012, from stevesounders.com: http://www.stevesouders.com/blog/2008/03/20/roundup-on-parallel-connections/

WhatWG. (2012, 05 15). *Embedded content*. Retrieved 05 30, 2012, from WhatWG: http://www.whatwg.org/specs/web-apps/current-work/multipage/embedded-content-1.html#attr-img-srcset

Wroblewski, L. (2011, 04 01). *Future Friendly*. Retrieved 04 11, 2012, from Future Friendly: http://futurefriend.ly/

Wroblewski, L. (2011, 10 01). *Mobile First*. Retrieved 04 11, 2012, from LukeW: http://www.lukew.com/resources/mobile_first.asp

Yahoo. (2011, 04 12). *Best Practices for Speeding Up Your Web Site*. Retrieved 05 29, 2012, from Yahoo Developer Network: http://developer.yahoo.com/performance/rules.html

Young, J. (2012, 02 27). *Is There Ever A Justification For Responsive Text?* Retrieved 02 27, 2012, from Smashing Magazine: http://www.smashingmagazine.com/2012/02/27/ever-justification-for-responsive-text/

Zijp, J. v. (2012, 03 20). *Load*. Retrieved 03 20, 2012, from JLab: http://jlab.nl/load

Zijp, J. v. (2012, 05 16). *Media Server*. Retrieved 05 16, 2012, from Jlab: jlab.nl

Zijp, J. v. (2012, 06 01). *Multi-device website Demo*. Retrieved 06 01, 2012, from JLab: http://jlab.nl/demo/

Bijlagen

Bijlage I. Topiclist

Responsive design

- Naar knelpunten vragen
- Media
- Flexibiliteit
- Content

Mobiele website

- Naar knelpunten vragen
- CMS
- Content
- Detectie / redirect
- Waarom losstaande mobiele website?

Design / Interaction

- Naar knelpunten vragen?
- Screen real-estate
- Mobile first
- Photoshop

Marc Bruisten - Front end

Sven Dinslage - Design

Linda Stuurman – Al veel met mobiel gedaan

Joost Verweij – Interaction Design