SmartElement

Automatiserad anpassning av webinnehåll

Staffan Enberg

Examensarbete
Informationsteknik
2014

INNEHÅLL

ORDLISTA

DOM Document Object Model. 4, 7

JSON JavaScript Object Notation. 4, 7, 10

JSONP Json-with-padding. 4, 7

1 INLEDNING

Detta arbete dokumenterar systemet SmartElement som utvecklades för företaget Developer's Helsinki Ab. Systemets syfte är att möjliggöra enkel konfiguration av anpassat innehåll för webbsidor, vilket tillåter webbsidors ägare att själva kunna konfigurera regler som bestämmer vad för innehåll som skall lyftas fram för olika användare.

1.1 Målsättning och syfte

Målsättningen med arbetet är att kartlägga den arkitektur som bäst stöder målet av ett effektivt system för innehålls filtrering och leverering, som även är lätt att vidare utveckla och utvidga i form av nya filter och nya algoritmer för filtrering.

1.2 Metoder

Systemet är i sig bara en motor för identifikation av användare och filtrering av innehåll. Det ända gränssnitt som SmartElement har är ett JSON-gränssnitt genom vilket ett skillt grafiskt gränssnitt tillåter användare att konfigurera sitt innehåll.

1.3 Avgränsningar

Detta är en redogörelse för den tekniska implementeringen inte användningen av anpassat innehåll på webbplatser, arbetet tar därmed inte ställning till hur anpassat innehåll skall användas, endast hur det kan implementeras.

Även om en av målsättningarna är att skapa ett effektivt system så finns det ingen klar standard för vad som kan anses vara bra prestanda för ett liknande system, därav kommer arbetet inte att innehålla grundliga prestanda analyser.

2 BAKGRUND

Projektet började våren 2013 som ett internt verktyg på Developer's Helsinki efter ett behov uppstått för ett system som skulle tillåta snabb implementering av personaliserat innehåll på kunders webbsidor.

Eftersom det i den tiden inte fanns någon färdig implementation att använda, bestämmdes det att det skulle uvecklas en mjukvara först för internt bruk men med målsättningen att lanseras som en SaaS produkt.

2.1 Anpassat innehåll

Figur 1. Amazons användarsida med anpassat innehåll

Anpassat innehåll, eller personaliserat innehåll (eng. *Personalized content*), är innehåll som väljs ut på basis av egenskaper hos användaren. I fallet av webbsidor kan det hand-

la om information som användarens geografiska läge, användarens språk, användarens webbläsare, antalet besök som användaren gjort till sidan, med mera. Tanken är att förse användaren med det den behöver eller vill ha utan att denne behöver be om det. (Mulvenna et al. 2000)

Anpassat innehåll har bland annat användts inom nätburiker för att visa reklam anpassad för kunden på basis av dennes bäställningshistorik, bilden 1 visar användarsidan som visas då man loggar in på amazons webb-butik. Inom social media har används anpassat innehåll för att lyfta fram innehåll som antas vara intressant för användaren (Uppdateringar från vänner som användaren ofta är i kontakt med, reklam från företag användaren har gillat m.m.). (Guy et al. 2010)

2.2 Behov

I takt med att stora aktörer införde allt mer anpassat innehåll så började även mindre kunder fråga om möjligheten att använda tekniken på sina webbsidor. Det visade sig att det inte fanns något företag på finska marknaden som erbjöd ett tillräckligt flexibelt system för anpassning av innehåll och därmed bestämdes det att Developer's Helsinki skulle utveckla ett sådant.

Systemet skulle kunna leverera innehåll dels i form av text (HTML eller rå text) för direkt injicering i webbsidans Document Object Model (DOM), dels i form av data i JavaScript Object Notation (JSON) format och dels som Json-with-padding (JSONP) svar vilket tillåter exekvering av javascript vid svaret. För filtrering skulle det samlas in data om beökares webbläsare och dator men det skulle även vara möjligt att tillägga egen data som sedan kunde användas för filtrering.

2.3 Befinttliga lösningar

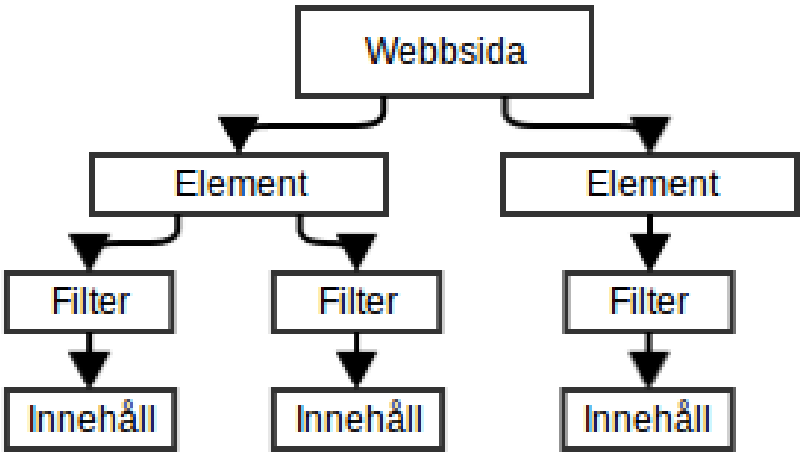
Vid tillfället för beslutet att utveckla produkten var den enda liknande lösningen på finska marknaden nosto.com som levererar personaliserat innehåll med inriktning på webbuti-

ker.(Nosto Website 2014) På grund av produktens klara vinkling passade den inte in i alla de användningsområden var SmartElement skulle användas.

3 SMARTELEMENT

SmartElement är ett back-end system som sköter filtrering av innehåll på basis av information som samlas in om besökare på en webbsida. Upprätthållare av en webbsida kan genom att integrera SmartElements JavaScript tag i sin sida, låta back-end systemet fylla i specicficerade element med personaliserat innehåll.

3.1 Beståndsdelar



Figur 2. Exempel på en hierarki i SmartElement

SmartElement bygger på några grundläggande koncept; webbsidor, element, filter och innehåll. Med dessa fyra byggstenar konfigureras det objekt, visualiserat i bild 2, som back-end systemet använder för att filtrera och leverera innehåll.

3.1.1 Webbsidan

Webbsidan är det högsta elementet som SmartElement handskas med. Det representerar en hel webbplats, under vilken man kan definiera element med innehåll. Varje sida har en unik identifikationsnummer som används för att ladda SmartElement tagen.

3.1.2 Element

Elementen är hållare för den data som returneras från back-end systemet och det som knyter innehållet till webbsidan. Elementet har i sig flere innehåll och filter, filtren används för att välja ut det innehåll som skall visas i elementet. Element har en inom sidan unik kod som används för att specificera vilka element som behöver processeras när man kallar på back-end systemet.

3.1.3 Filter

Filter berstår i SmartElement av flere olika regler som ställer en fråga om besökaren samt en länk till ett innehåll, vilket visas om filtret matchar. För att ett filter skall matcha krävs det att besökaren passar alla regler i filtret. För att hantera fall var två eller flere filter passar för en beökare har filter en prioritetsordning inom elementet, valbar av användaren.

3.1.4 Innehåll

Innehåll är den konkreta data som levereras för ett element efter att ett filter valts som vinnare. Innehållet kan bestå av text, HTML kod eller JSON data, det är upp till användaren att definiera vad för data som sparas.

3.2 Funktion

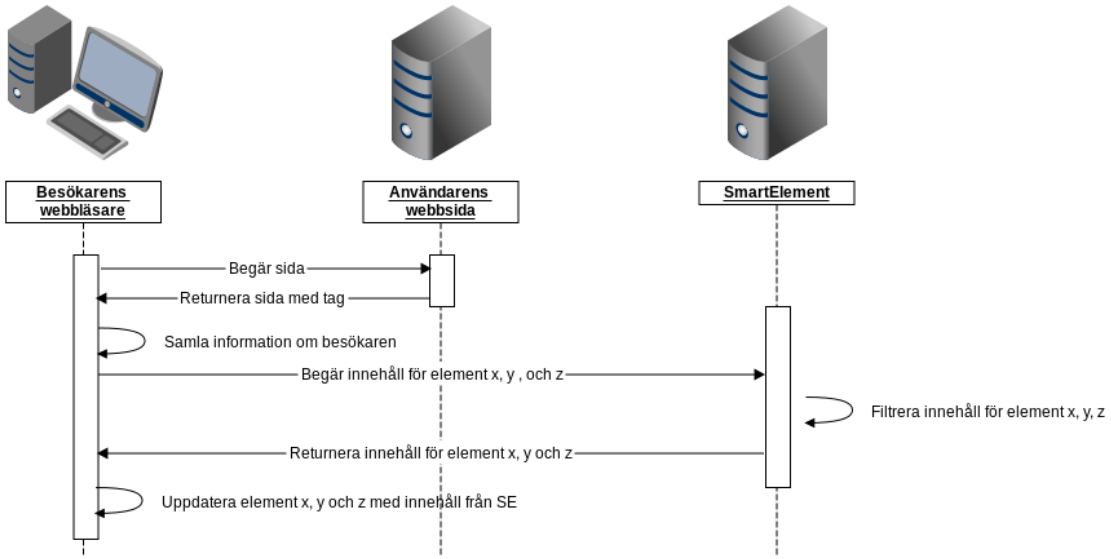
SmartElement har två huvudsakliga funktionsområden, konfiguration av sidor och hantering av sidvisningar.

3.2.1 Konfiguration

För att SmartElement skall kunna leverera innehåll, måste användaren konfigurera sidan och dens element på förhand. Processen består av att registrera element under sidobjektet, skapa innehåll samt att konfigurera filter genom att kombinera regler om vilka användare

innehållet passar för.

3.2.2 Sidvisning



Figur 3. Processen vid en sidvisning

Bild 3 representerar kommunikationen vid en sidvisning. När en användares webbsida laddas, inkluderar den en JavaScript fil som innehåller SmartElements tag. Efter att sidan laddats klar körs tagen, som samlar ihop information om besökaren samt eventuell extra data som användaren specificerat i tagen, och skickar informationen till back-enden.

Backenden tar emot data från tagen och söker upp sidan som motsvarar det id som skickats. Efter att en sida har hittats går back-enden igenom de element som tagen begär innehåll för och filtrerar deras innehåll på basis av den data som tagen skickat. Allt innehåll kompileras till en lista med element-id länkat till innehåll och skickas tillbaka till besökarens webbläsare.

Om användaren inte valt att ändra på beteendet av tagen så kallar backenden på tagen för att fylla elementen med den data som returnerats.

4 TEKNIK

SmartElement bygger i stort sett på fem centrala tekniker, PHP som användts för att programmera progamlogiken, MySQL som databaslager var data lagras, Memcache som caching lager vilket tillåter att information som är tung att beräkna kan sparas mellan förfrågningar, dokument databsen MondgoDB för lagring av information om besökare och till sist JavaScript som används för att skapa det informationspaket om användaren som skickas till servern för användning vid filtreringen.

4.1 PHP

PHP är ett skriptspråk som utvecklats med målet att snabbt kunna skapa dynamiska webbsidor. (PHP Manual 2013) Språket valdes för att det är enkelt att arbeta med och det var språket som företagets andra produkter va skrivna med. Det finns också en mängd information och färdiga kodpaket till förfogande för programmerare, vilket underlättar och försnabbar utvecklingen.

4.2 MySQL

För lagring av data valdes databasmjukvaran MySQL. En relationell databas passar sig bra för lagring av den information som systemet behandlar eftersom det handlar som entiteter som är starkt länkade till varandra.

4.3 MongoDB

Dokumentdatabsen MongoDB används för lagring av användarinformation.

4.4 Memcache

Eftersom SmartElements datamodeller är relativt tunga att bygga så bestämmdes det att ett caching lager skulle användas för att hålla dessa datamodeller och på så vis låta mjukvaran gå direkt till filtrerings logiken då en förfrågan behandlas.

Målet med cache lagret är att hålla sidorna i minnet på servern, och endast läsa från databasen när ett objekt byggs om i samband med en uppdatering. För detta ändamål valdes Memcache, en väletablerad mjukvara för caching.

4.5 JavaScript

För att kunna identifiera så mycket information som möjligt om användaren är det nödvändigt att köra kod på klienten innen innehållet kan presenteras. I dagens läge finns det endast ett skriptspråk som kan köras på godtycklig klient, och det är javascript, vilket valdes även för detta projekt.

5 PROCESSER

Processer för identifikation av användare och filtrering av innehåll.

5.1 Information om användaren

En genomgång av de olika typerna av information man kan samla in om användare samt fördelar och svårigheter med dessa.

5.1.1 IP Geolokalisering

Geolokalisering på basis av användarens IP-address. Vilket land kommer användaren ifrån, vilken stad, vilken region.

5.1.2 Besökarstatistik

Information om hur månte gången besökaren återvänder, hur ofta denne sett sidan, allmän information om beteemde.

5.1.3 Tid

Vad är den lokala tiden för sidan, vad är den lokala tiden för användaren, hur stor tdisskillnad är det mellan de två, hur länge har användaren varit på sidan.

5.1.4 Hänvisningsinformation

Varifrån kom användaren, vilka sökord använde den eventuellt. Vilken sida visades först. Varifrån har användaren tidigare anländt till sidan.

5.1.5 Användardefinierad information

Information som websidan själv tillhandahåller.

5.2 Filtrering

Hur kan man filtrera på basis av den insamlade informationen?

6 ARKITEKTUR

Förklaring av arkitekturen

6.1 Back-End system

Back-end systemet är det som gör den egentliga processeringen av information och väljer ut innehållet som levereras.

6.1.1 Matcher

Matchern är den kod som tar emot ett Visitor objekt som innehåller information om användaren och kör sedan igenom filtren för sidan med detta objekt.

6.1.2 Site ojektet

Objektet som innehåller all information om webbsidan, elementen, filtren och innehållen.

6.1.3 Element objekten

Element objekten motsvarar ett HTML element på webbsidan, dessa har filter med vilka det kan definiera vilket innehåll det skall visa.

6.1.4 Filter objekten

Filter objekten innehåller regler som definierar en målgrupp och ett innehåll som det returnerar.

6.1.5 Content objekten

Content objekten är innehåll som returneras, detta kan vara dels HTML innehåll eller javaskript kod.

6.2 Javascript tagen

Javascript tagen är en kort kod som laddas i samband med sidan och samlar ihop informaiton om användaren och sedan skickar denna information till back-end sytemet för processering.

7 SAMMANFATTNING OCH DISKUSSION

Sammanfattning av arbetet, tillbakatitt på produkten. Diskussion om förbättringar som skulle kunna göras i systemet. Diskussion om tekniker som kunde bytas ut för att eventuellt förbättra prestanda. Deiskussion om svårigheter med produkten.

8 FÖR UPPGIFTEN UTVIDGAD DISPOSITION

Istället för att skapa ett skillt dokument valde jag att skriva använda mig av dokumentet som kommer att bli mitt slutarbete. Här kan man se en grund för arbetet och ett preliminärt skelett för dokumentet.

Jag har påbörjat arbetet med att skriva en kort bakgrund om projektet samt en kort introduktion av systemet som hoppeligen kommer att hjälpa läsaren förstå olika val som kommer att förklaras ki arkitektur delen. Jag har även förökt skriva ned några korta meningar under alla rubriker för att ge en bild av det tänkta innehållet.

Källförteckningen är kanske lite kort men den innehåller två relativt tunga verk, den så kallade Gang of Four boken (Design Patterns: Elements of Reusable Object-Orientated Software) och Introduction to algorithms, vilka i samband med The Pragmatic Programmer och High Performance Mysql ganska långt täcker arkitekturen av projektet. Referenserna sträcker sig inte så långt ennu pga de skrivna kapitlens natur, bakgrunden har någon referens till tidsskrifter som behandlat personalisering. Jag skulle dock säga att process och arkitektur kapitlen kommer vara de som behöver mest referenser, eftersom det är där jag gjort ställningstaganden och val som kan behöva stödas. Stilen är Harvard.

Av källorna jag valt är Design Patterns och Introduction to Algorithms båda rätt kompletta verk inom respektive fält så de kan anses vara rätt trovärdiga. The pragmatic programmer är en bok som titt som tätt rekommenderas inom mjukvarubranschen, den innehåller många beprövade metoder för mjukvaruutvekling som är i bruk världen över. Utöver att söka böcker så har jag användt mig av Association for Computing Machinerys söktjänst för att söka information samt försökt använda nelliportalen, ACM kan anses vara ganska

trovärdig källa som en av världens äldsta föreningar för datavetenskap. Den information som tagits från webben är närmast profilering av produkter, enligt deras egen utsago. Informationen är ju inte nödvändigtvis referentgranskad, men den används ej heller i sådan utsträckning var det skulle behövas.

KÄLLOR

Cormen, Thomas H.; Leiserson, Charles E.; Rivest, Ronald L. & Stein, Clifford. 1990, *Introduction to Algorithms*, Massachusetts: MIT press.

Gamma, Erich; Helm, Richard; Johnson, Ralph & Vlissides, John. 1994, *Design Patterns: Elements of Reusable Object-Oriented Software*, Massachusetts: Addison-Wesley Professional.

Guy, Ido; Zwerdling, Naama; Ronen, Inbal; Carmel, David & Uziel, Erel. 2010, Social Media Recommendation based on People and Tags, I: *Proceedings of the 33rd international ACM SIGIR conference on Research and development in information retrieval*, New York: ACM, s. 194–201.

Hunt, Andrew & Thomas, David. 1999, *The Pragmatic Programmer*, Massachusetts: Addison-Wesley Professional.

Mulvenna, Maurice D.; Anand, Sarabjot S. & Büchner, Alex G. 2000, Personalization on the Net using Web mining: introduction, *Communications of the ACM*, årg. 43, , s. 123–125.

2014, *Nosto Website*. Tillgänglig: http://www.nosto.com, Hämtad: 23.3.2014.

2013, *PHP Manual*. Tillgänglig: http://fi2.php.net/manual/en/, Hämtad: 2.3.2014.

Schwartz, Baron; Zaitsev, Peter; Vadim Tkachenko, Jeremy D. Zawodny; Lentz, Arjen & Balling, Derek J. 2008, *High performance MySql*, 2 uppl., Sebastopol: O'Reilly Media, Ltd.