SmartElement

Automatiserad anpassning av webinnehåll

Staffan Enberg

Examensarbete
Informationsteknik
2014

INNEHÅLL

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.

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.).

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 system för anpassning av innehåll och därmed bestämdes det att Developer's Helsinki skulle utveckla ett sådant.

Allt flere användargränssnitt innehåller dels manuell och dels automatisk organisering av innehåll, så väl på webben som på terminaler som datorer och mobila enheter.

2.3 Befinttliga lösningar

Lösningar som redan skapats, undersök om det finns nåt att lära sig.

3 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.

3.1 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 i. Det finns ochså en mängd information och färdiga kodpaket till förfogande för programmerare, vilket underlättar och försnabbar utvecklingen.

3.2 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.

3.3 MongoDB

Dokumentdatabsen MongoDB används för lagring av användarinformation.

3.4 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.

3.5 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.

4 PROCESSER

Processer för identifikation av användare och filtrering av innehåll.

4.1 Information om användaren

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

4.1.1 IP Geolokalisering

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

4.1.2 Besökarstatistik

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

4.1.3 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.

4.1.4 Hänvisningsinformation

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

4.1.5 Användardefinierad information

Information som websidan själv tillhandahåller.

4.2 Filtrering

Hur kan man filtrera på basis av den insamlade informationen?

5 ARKITEKTUR

Förklaring av arkitekturen

5.1 Back-End system

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

5.1.1 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.

5.1.2 Site ojektet

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

5.1.3 Element objekten

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

5.1.4 Filter objekten

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

5.1.5 Content objekten

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

5.2 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.

6 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.

KÄLLOR

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