Projekt

Uppdatering 2013-12-11

Redovisningar av projekt kan ske antingen före jul eller efter nyår. Exak

- Deadline för projekt om man vill redovisa före jul preliminärt 16/12 kl 10-12 och om det blir fulltecknat dagen innan 15/12.
- Deadline för projekt om man vill redovisa efter nyår blir preliminärt 14/1 9-16 samt 16/1 9-12. Om tiderna inte räcker till kommer jag även avsätta tider veckan därpå.

Projektuppgift

I denna projektuppgift ska du göra en XML-baserad internettjänst för flerkanalspublicering av någon typ av data. Exempel på vad internettjänsten kan avhandla är ett kalendarium, ett skivregister och ett redaktionellt system, men ni är fria att komma på egna tjänster. Tjänsten ska kunna utnyttjas genom minst två olika ut-enheter, förslagsvis genom en webbläsare på en dator och en smartphone. Användare ska kunna mata in data för publicering genom ett webbgränssnitt på minst en enhet.

Jag har gjort två screencasts, som finns under kursinnehåll->föreläsningar som beskriver ett enkelt projekt i sin helhet (nästan).

Exakt vilken intenettjänst som ska implementeras är valfritt efter samråd med lärarna, så länge som uppgiften ligger inom ramarna. Några förslag ges nedan. Egna utvidgningar är möjliga och rekommenderade. Exakt hur många poäng som ska ges för egna utdvidgningar beror på uppgiftens svårighet.

Deadline och redovisning

Se schema för preliminär deadline. Deadline för rapporten är ett par dagar innan datumet för er redovisning. Exakta datum för redovisning tas fram mot slutet av kursen då flera grupper börjar känna sig redo. Redovisningen tar ca. 15 minuter var av 10 minuter ska vara en presentation och sedan ca. 5 minuter för diskussion och bedömning. Redovisningen sker i en datorsal, en grupp i taget, direkt till läraren.

Basuppgift (18 poäng + 5 poäng "i-tid-bonus")

Input via HTML-formulär via PHP-sidor (eller motsvarande, t.ex. JSP, ASP eller Ruby on Rails)

I grunduppgiften ingår att användare via webben ska kunna mata in data till systemet via ett webbformulär. Datat ska skickas till PHP-sidor eller motsvarande som bearbetar datat, och stoppar in relevant data i en databas. Om ni väljer någon annan metod än PHP får ni räkna med att i hög grad klara er utan handledning. av programmeringsteknisk karaktär.

Ändring av inlagt data

Det ska även vara möjligt att enkelt modifiera inlagt data. Det enklaste sättet att göra detta är förmodligen att skriva en PHP-sida eller motsvarande som läser in data från systemet och genererar inputformuläret enligt ovan, men med data från systemet förifyllt. Därefter skickas data igen, systemet ser att posten ifråga redan finns och väljer då att göra en update av det tidigare inmatade datat.

Lagring i databas

Datat ska lagras i en mysql-databas. Detta kräver att ni först gör en modell över den databasstruktur ni ska använda, skapar lämpliga tabeller och slutligen implementerar databaskopplingar för att lagra och hämta ut data. Exempel på hur detta kan göras kan ni hitta i RSS/PHP-laborationen och föreläsningsanteckningarna.

Export till XML

Datat i databasen ska på något sätt exporteras till XML för att möjliggöra vidare behandling med XSLT-stylesheets. Ni måste göra en DTD som beskriver de datastrukturer ni ska använda. XML ska genereras dynamiskt via PHP/motsvarande som i RSS/PHP-laborationen.

Stylesheets för generering av presentationsdata för minst två olika ut-enheter

XSL stylesheets ska göras som översätter informationen i XML-dokumenten till minst två olika ut-enheter, (förslagsvis XHTML, XHTML anpassat för liten skärm (som mobiltelefon) eller WML). Detta ska gälla samtliga sidor, även formulärsidor. Målet ska vara att (i möjligaste mån) uppnå *Device Independence* som det formulerades i DIP-1 (föreläsning 5):

DIP-1: Device Independent Access
For some web content or applications to be device independent, it should be possible for a user to obtain a functional user experience associated with its web page identifier via any access mechanism.

Observera att din lösning **inte** får använda frames. Det mesta man kan vilja använda frames till kan lösas via XSLT.

Tillvägagångssätt

Uppgiften utföres i grupper om två personer.

1. Definiera uppgiften och gör en tidsplan

Skriv på ett/ett par A4-papper en beskrivning av systemet du tänker göra. Det ska klart framgå vad syftet med systemet är och vad som ska kunna utföras med systemet (någon form av funktionsbeskrivning).Gör grova skisser på hur gränssnitten kan tänkas se ut. Skriv med eventuella utvidgningar utöver basuppgiften du planerar att göra för att få en bedöming på hur många extrapoäng de kan tänkas ge. Detta moment bör godkännas av mig eller assistenterna innan ni går vidare. Observera att funktionaliteten som beskrivs här absolut inte är helig, det går att modifiera planen efter hand. Om detta moment är inlämnat och är tillräckligt bra senast **innan deadline för detta moment (se schema) erhålles en (1) bonuspoäng**.

2. DTD och databasstruktur

Gör en databasstruktur (dvs de tabeller som ska ingå, vilka kolumner och nycklar de ska ha samt relationerna mellan tabellerna). Gör dessutom en DTD för den/de XML-struktur(er) som ska genereras utifrån innehåll från databasen. Dessa ska innehålla både en innehållsmodell och en navigationsmodell. [**Förtydligande: Innehållsmodellen är i praktiken modellen för innehållet, alltså precis vad ni övat på. En kokbok består av ett eller flera recept, ett recept består av en instruktion och en lista med ingredienser etc. Navigationsmodellen är tänkt att vara en XML-baserad representation av hur man navigerar, och utifrån den ska man kunna generera navigationsmenyer, breadcrumbs mm. Se http://www.webdesignfromscratch.com/website-architecture/navigation-models/ . Det är inte något ni kommer få underkänt om ni inte gjort, eftersom det varit otydligt vad det innebär.]**

En utvidgning är att dessutom göra ett XML Schema som beskriver motsvarande struktur. Detta moment **ska** godkännas innan ni går vidare. Om denna är inlämnad och är tillräckligt bra **innan deadline för detta moment (se schema) erhålles en (1) bonuspoäng**. XML Schemat kan dock göras i efterhand.

Notera att det troligen kommer krävas åtskilliga XSLT-stylesheets, i extremfallet två stylesheet (ett för webb, ett för mobil) för varje gränssnitts-skiss ni gjort i punkt 1 ovan. Förhoppningsvis blir det dock färre.

3. Implementera

Implementera systemet.

4. Rapport och redovisning

Gör en skriftlig rapport där ni beskriver användningsområdet för ert system. Rapporten ska även innehålla skärmdumpar samt köranvisningar. Rapporten bör vara ca 5-10 sidor som en tumregel. En till ett par sidor ska innehålla reflektioner över för- och nackdelar med att bygga ett publiceringssystem på detta sätt. Systemet ska sedan demonstreras för läraren. Om detta moment är redovisat **innan deadline för detta moment (se schema) erhålles tre (3) bonuspoäng**.

Generella utvidgningar

Behörighetskontroll för att skapa/ändra. Individ/gruppnivå (4p)

Skapa ett system för behörighetskontroll med användare och inloggning. Endast registrerade användare med rätt lösenord ska kunna lägga in data i systemet, men vem som helst(?) ska kunna titta på informationen i systemet via HTML/WML-gränssnitten. Dessutom ska endast behöriga personer kunna ändra på redan inlagt material. Personen som lagt in material ska kunna ändra samma material. Dessutom ska en superanvändare finnas som har rätt att ändra i allt material. Detta kan förslagsvis implementeras som att användare kan tilldelas "egenskapen" superanvändare. Uppgiften löses t.ex. med sessionsvariabler eller cookies.

Funktionalitet för att administrera användare (3p, kräver föregående utvidgning)

Gör ett system för att **administrera** användare av systemet, med behörighet och lösenord.

Använd CAS för autenticering (2p)

KTH använder CAS (Central Authentication System) för att autenticera användare, t.ex. när man loggar in i Bilda. phpCAS finns dokumenterat och finns att hämta på https://wiki.jasig.org/display/CASC/phpCAS. Man autenticerar sig mot servern login.kth.se, exempel med det finns på KTH Social.

Sökfuntioner (3p)

Inför möjligheter att söka i tjänsten. Exempelvis söka efter all Massive Attack-skivor i en skivdatabas, efter alla M-fester i ett kalendarium eller söka efter alla sportartiklar i maj månad i ett redaktionellt system.

Ajax/motsvarande (ca 3p)

Använd Ajax (eller någon liknande XML-baserad teknologi) för att uppdatera delar av webbsidan utan att behövauppdatera hela sidan.

Export till fler än 2 kanaler (3p/kanal)

För varje ytterligare utkanal (t.ex. SVG, WML, FO) utöver de obligatoriska två får du ytterligare 3 poäng. Dessa utkanaler ska skapas genom XSLT-transformationer. Förslagsvis görs extralaborationen på respecktive kanal innan.

RSS/Atom-feed över uppdateringar i systemet (3p)

Om projektet är av en karaktär att det i någon mening vore meningsfullt med en RSS/Atom-feed över inlagda och/eller ändrade poster i systemet är denna utvidgning möjlig.

Podcast av filer som läggs in i systemet (3p)

Om projektet innehåller mediafiler, såsom en bilddatabas, musiksite, videosite eller liknande, och är av en karaktär att det i någon mening vore meningsfullt med en podcast över nyinlagda mediafiler är denna utvidgning möjlig.

Flera språk (3p)

XML är ett bra sätt att spara översättningar mellan olika (mänskliga) språk. Gör en XML-fil (med lämplig struktur) som innehåller all text som visas i gränssnittet på sidan och spara sedan en kopia översatt till minst ett annat språk. Låt därefter användaren välja vilket språk sidan ska visas på och hämta översättningen från respektive fil vid XSLT-transformationen.

In/utcheckning av objekt (3p)

Gör kontrollmekanismer så att två användare inte samtidigt kan vara inne och modifiera i ett objekt. Detta kan ske genom att checka in ett objekt när man ska editera det, och sedan checka ut det när man har ändrat klart i objektet. Så länge objektet är incheckat är det "read only" för övriga.

Dynamisk omskalning av bilder (3p)

Det är ofta onödigt att skicka en full-storleks bild till en mobiltelefon. Bättre är då att skala ner bilder dynamiskt så att de inte överförs data i onödan. Detta kan göras t.ex. genom GD-paketet som finns till exempelvis PHP.

Använda WURFL för att avgöra klientens egenskaper och anpassa stylesheets efter dessa (olika poäng beroende på komplexitet)

Obs! WURFL kan vara knepigt. WURFL håller listor öven en mängd enheter som kan användas för att surfa på internet och vad varje enhet har för egenskaper. Använd dessa egenskaper för att skapa bra anpassade stylesheets.

Använd Twitters API:er för att uppdatera en twitterstatus (3p)

Twitter har utmärkta APIer. Använd lämpligt API för att få ändringar som sker på din site att dyka upp även i statusuppdateringar på Twitter.

Koppa twitteruppdateringarna ovan till RSS+Facebook (1p)

Det finns webbtjänster som gör detta enkelt, hitta och utnyttja dem.

Använd APIer mot andra nätverkstjänster för att göra något kul (ca 2-3p per koppling)

Exempel kan vara Flickr, Vimeo eller YouTube.

Gör rapporten både som screencast och textrapport (2p)

Standard är att göra en textrapport, men om du också gör en screencast som kan användas som exempel på kursen i framtida kursomgångar får du ytterligare två poäng. Screencasts kan göras direkt från webbläsaren med http://www.screentoaster.com, från QuickTime som finns i Snow Leopard eller med en uppsjö andra programvaror. Lägg upp videon på YouTube eller ScreenToaster.

Skriv rapporten i DocBook-format och generera PDF från denna (2p)

Se t.ex. Jaxe eller OpenOffice för en gratis DocBook-editor (OpenOffice kan spara i DocBook-format).

Ni kan sedan gå tillväga så här:

Skriv rapporten i OpenOffice, och spara den i DocBook-format. Se till att eventuella bilder ligger i samma katalog som du sparar dokumentet i, så slipper du ändra sökvägar manuellt senare.

För över DocBook-filen (t.ex. rapport.xml), samt alla bilder du använt till din hemkatalog på CSC-skolan. Logga in på my.nada.kth.se med ett terminalfönster och skriv (om rapporten heter rapport.xml)

```
my:~>module add fop
my:~>fop -xml rapport.xml -xsl ~bjornh/Public/docbook-xsl-1.74.0/fo/docbook.xsl out.pdf
```

Då har du förhoppningsvis resultatet i filen out.pdf

Det som händer under ytan är att fop gör en XSLT-transformation av er DocBook-fil tillsammans med de mycket bra XSLT-stylesheets som finns på http://docbook.sourceforge.net/, och gör om det till en PDF-fil. Det finns otaliga parametrar man kan ändra för att få rapporten att se ut som man vill, för mer info se t.ex. http://wiki.docbook.org/topic/DocBookXslStylesheets. För att få godkänt räcker det dock att ni använder defaultstylesheetsen.

Exempel: Anteckningsbok

Gör ett system som kan användas för att skriva, redigera och läsa anteckningar. Anteckningarna bör gå att koppla till en eller flera kategorier, t.ex. en viss kurs. Prata gärna med Björn om du är intresserad av att göra denna uppgift.

Spaced repetition (5 poäng)

Anpassa systemet så det kan användas för spaced repetition. Denna utvidgning kräver möjligheter att logga in.

Delade anteckningsböcker (4 poäng)

Gör det möjligt att avgöra om enskilda anteckningar ska vara läsbara för andra, antingen på gruppnivå eller för samtliga. Det ska då vara möjligt för en användare att se vilka andra anteckningar som är tillgängliga och då lägga till dessa till de privata anteckningarna.

Exempel: Redaktionellt system

Formatering av artiklar (ingår i basfunktionaliteten)

Ett möjligt sätt för att lagra metainformation om själva innehållet i texten är att låta journalisterna/redigerarna använda enkla fomatterings-tags. Förslagsvis kan dessa vara på något XML-format definierat av er, t.ex. <rubrik>Huvudrubrik</rubrik><ingress>en ingress</ingress> och så vidare. Exempel på taggar som bör ingå är:

- Rubrik
- Ingress
- Mellanrubrik
- Brödtext
- Signatur
- (Eventuellt) stycken

Klassificera inkommet material (ingår i basfunktionaliteten)

Materialet kan märkas upp med olika typer av meta-information om en artikel, t.ex. sport, lokalt, ekonomi etc, publicerinsdag osv.

Basera datat på NewsML (2-4 poäng beroende på komplexitet)

Använd NewsML eller liknande existerande format som dataformat.

Bilddatabas (7 poäng)

Gör funktionalitet för att lägga in, hämta ut och söka efter bilder i en bilddatabas. Följande bör ingå:

- Bildformat (tiff, jpeg, etc.)
- Publiceringsdag (bör kunna vara flera dagar)
- Bildtext (från de gånger bilden varit publicerad)
- Beskrivning (lite utförligare beskrivning av vad bilden föreställer, underlättar t.ex. vid sökningar)
- Storlek (filstorlek, optional)
- Tumnagelbild (liten variant av bilden som går snabbare att hämta/visa)

- Sökmöjligheter (kunna söka efter "AIK massaker" osv)

Planeringsverktyg (4 poäng)

Gör ett verktyg för planering av vilka artiklar som ska skrivas. När någon bestämmer att en artikel bör skrivas kan denne någon föra in detta i systemet med hjälp av detta verktyg. Om IFRAtrackdelen implementerats ska ett ifratrackmeddelande skickas. Om

- Planerat publiceringsdatum (om något)
- Vem som ska göra jobbet (om bestämd) (om användarhanteringsuppgiften gjorts bör dessa givetvis vara kopplade mot varandra
- Kommentarer i fritext
- Prioritet (siffra t.ex. 1 - 5)
- Antal tecken (om bestämt)

Exempel: Kalendarium

Gör ett kalendarium där man kan boka aktiviteter. Följande parametrar ska kunna matas in: (ingår i basuppgiften)

- Datum
- Starttid/sluttid
- Plats
- Ansvarig
- Webb-länk (om tillgänglig)
- Fritext
- Grupp (om aktiviteten endast gäller en viss grupp av användare av systemet ska endast dessa kunna se aktiviteten)

Krock-hantering (4 poäng)

Gör kontroll så att ingen kan lägga in events som krockar med varandra i både rummet och tiden.

Periodiska mailutskick (4 poäng)

Gör ett program som skickar ut information via email (utbyggnad: sms?) om vad som händer den närmaste veckan. Ett sådant program kan sedan scheduleras att köras varje vecka med hjälp av unix-kommandot cron.

Prenumeration på aktiviteter (4 poäng)

En (vettig) utbyggnad av detta är att låta användare "prenumerera" på vissa typer av aktiviteter, exempelvis "jag vill prenumerera på alla PRU-aktiviteter och M-aktiviteter. Dessa ska sedan fås genom ett mailutskick (se ovan).

Exempel på övriga uppgifter

- Skivregister
- Wiki
- Glosförhörssystem (utvidgning: spaced repetition (se ovan)). Se http://3rik.net/xamine/ för exempel på hur upplägget kan vara.
- Blog
- Receptbok
- Gruppsystem (liknande kalendarieuppgiften)
- Tidrapporteringssystem
- Newsgrupp-liknande system.
- Börskursbevakningssystem

Björn Hedin skapade sidan | 24 oktober 10:30

... eller skriv ett nytt inlägg

Alla användare med KTH-konto får läsa.

Senast ändrad: 2014-10-24 11:53. Visa versioner

Taggar: Saknas än så länge.

Följ denna sida Anmäl missbruk

---

http://www.kth.se/