Hanzehogeschool Groningen
University of Applied Sciences

Instituut voor Communicatie, Media & IT

Thema 4.1 I:
Web Application Development

Opgaven week 1

1. Inleiding

Deze week staat in het teken van de geschiedenis van het web, de werking van webservers, webstandaarden en -protocollen en versiebeheersystemen. Na wat theoretische vragen wordt ingezoomd op webservers in het algemeen en Apache in het bijzonder. Tenslotte wordt één van de lastigste problemen van http behandeld.

Opgave 1: webservers, software en protocollen – de theorie

1. Waarom is het van belang dat er een protocol is dat door zowel de servers als de clients wordt ondersteund?
2. Het http is nu in versie 1.1. Wat zijn de belangrijkste verschillen tussen deze versie en versie 1.0?
3. Installeer een plugin in je browser zodat je de http-headers kunt zien (bijvoorbeeld live http-headers voor Chrome of FireFox, ieHttpHeaders voor Internet Explorer) en ga vervolgens met je browser naar Blackboard. Beschrijf en verklaar een aantal van de headers die langskomen.
4. Wat is de rol van al die cookies die in de headers heen en weer gestuurd worden?

Bestudeer de tekst op de onderstaande twee pagina's:

- ✓ http://www.devshed.com/c/a/Apache/Apache-and-the-Internet/1 en
- ✓ http://www.devshed.com/c/a/Apache/Apache-and-the-Internet/2.

De onderstaande vragen gaan over deze tekst:

5. Waarom is het logisch dat Apache werkt als een deamon?
6. In de tekst is sprake van MPMs. Wat zijn dat en wat is hun functie?
7. Wat zijn de voordelen van de modulaire opzet van Apache?

De Amerikaanse ondernemer en software-ontwikkelaar Marc Andreessen

Bestudeer het artikel *Why Software is Eating the World* van Marc Andreessen (dat je op Blackboard kunt vinden) en beantwoord de volgende vragen:

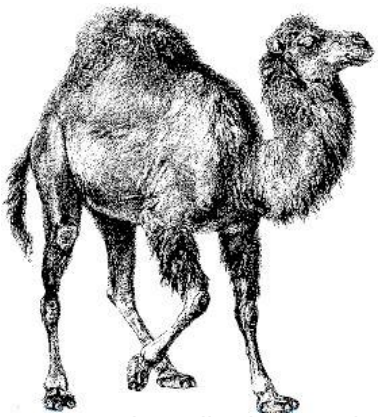
8. Wat is het centrale punt van dit artikel?
9. Andreessen beschrijft twee processen die ertoe leiden dat er in het komende decennium veel bedrijven verstoord (disrupted) zullen worden door software. Welke twee processen zijn dat?
10. Geef zelf drie voorbeelden van bedrijven of activiteiten die in toenemende mate software-gedreven worden.

Opgave 2: apache webserver

Deze opgave gaat in op het configureren van Apache. Download hiervoor de meest recente versie van Apache of maak een kopie van de huidige installatie (het is expliciet *niet* de bedoeling dat er met xampp of vergelijkbare tools wordt gewerkt).

1. Maak ergens op het file system een directory `Opdracht2` aan met daarin directories `html`, `img`, `perl`, `notfound`, `logs` en `secret`.
2. Pas de configuratie dusdanig aan dat Apache het volgende gedrag vertoontopen:

   - ✓ html-bestanden worden opgezocht in de directory `html`;
   - ✓ plaatjes (png, jpe?g en gif) worden gezocht in de `img` directory;
   - ✓ als een bestand (plaatje of html) niet wordt gevonden, wordt het bestand 404.html dat in de directory `notfound` staat teruggegeven;
   - ✓ bestanden die in de directory `perl` staan worden niet als zodanig teruggegeven maar geïnterpreteerd als perl-script en als zodanig uitgevoerd, waarna het resultaat wordt teruggegeven;
   - ✓ als het script in de directory `perl` een fout bevat, wordt deze fout teruggestuurd;
   - ✓ de bestanden in de directory `secret` worden pas getoond wanneer er de juiste credentials worden meegegeven aan het request (er hoeft geen koppeling met een database gemaakt te worden hiervoor - een eenvoudig htpasswd is voldoende).
   - ✓ Alle log-bestanden komen in de directory `log` terecht.
3. Zet in elke directory een bestand om de werking te controleren.
4. Maak een webformulier dat de bezoeker vraagt om haar naam en geboortedatum. Wanneer zij het formulier opstuurt, moet een perl-script worden uitgevoerd dat haar vriendelijk welkom heet en aangeeft wat haar leeftijd is.
5. Gebruik `tail` (of een vergelijkbare tool) om de logbestanden te zien tijdens het opvragen van verschillende bestanden. Vraag een eenvoudige html-pagina op, het perl-script dat in de vorige deelvraag is gemaakt en een pagina die niet bestaat. Leg uit wat er in de log-files te zien is.
6. Demonstreer de werking aan de practicumdocent.

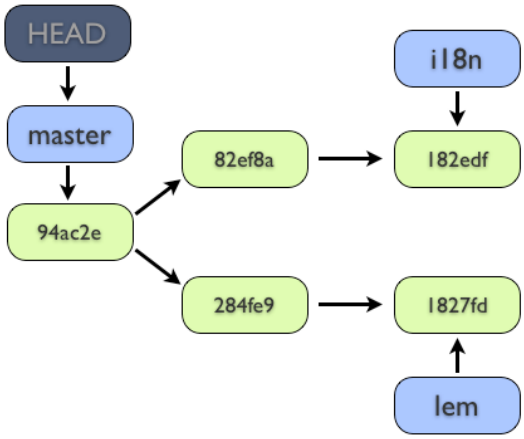


Opgave 3: git

In dit thema moet je gebruik maken van het versiebeheersysteem git. Tijdens het theoriecollege is al aandacht besteed aan versiebeheersystemen in het algemeen en git in het bijzonder. In deze opgave moet je wat theorievragen beantwoorden en wat praktijkopdrachten uitvoeren.

1. Bekijk de introductie van Scott Chacon die op youtube te vinden is. Tijdens het mondeling kunnen hier vragen over gesteld worden.

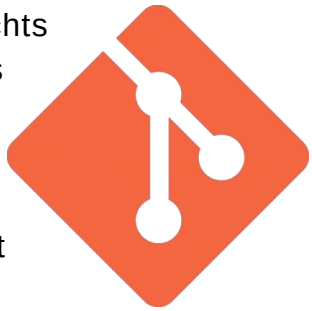
2. Wat wil het zeggen dat git een gedistribueerd systeem is?
3. Wat is een *file based delta storage system* en waarin verschilt dit van de manier waarop git werk?
4. Wat is de rol van de het index-bestand in de .git-directory?



5. Bekijk de onderstaande situatie
   a. Wat betekenen die hexadecimale getallen in de verschillende groene blokjes?
   b. Is een fast-foward merge op dit moment mogelijk bij git merge i18n? Beargumenteer je antwoord.
   c. Is vervolgens een fast-forward merge mogelijk bij git merge lem? Beargumenteer je antwoord.
6. Maak een directory aan op je locale file system, zet daar wat bestanden in en voeg die toe aan een git-repository. Wijzig vervolgens één van deze bestanden en leg uit wat je ziet wanneer je git status doet. Doe nu een git add op het bestand dat je net hebt gewijzigd en opnieuw een git status. Leg het verschil tussen deze twee uit.
7. Haal de bestanden die je in de vorige stap hebt gemaakt weg uit de directory doe vervolgens git commit. Wat valt je op aan deze werkmethode?
8. Maak ergens anders op je file system een clone van de repository uit de vorige opgaven en vergelijk de inhoud van deze clone met het origineel. Wat is het opvallende verschil?
9. Demonstreer de uitkomsten aan de practicumdocent.

Opgave 4: een webserver in Java

Deze opgaven staan in het teken van een webserver in Java, die slechts een zeer beperkte subset van HTTP ondersteunt. Op Blackboard is een aanzet tot implementatie van een webserver te vinden (als een Eclipse project: HTTPServer.zip). De netwerklaag in de vorm van socketafhandeling is al gegeven, deze opgave richt zich op het afhandelen van het HTTP verzoek. Zorgt ervoor dat de server reageert op request vanaf een browser.



1. Implementeer de volgende functionele eisen:

   a. Stel de basisdirectory waar vanuit bestanden worden gehost in. Alleen bestanden in deze directory en subdirectories kunnen worden getoond, hoewel het gebruik van .. als aanduiding voor een parent directory mogelijk is binnen de eerder gestelde begrenzing. De mapping van URL naar bestand gaat op de volgende manier: de url http://localhost:port/foo/bar/demo.html wordt gemapt op het bestand <<basisdirectory>>/foo/bar/demo.html.

   b. Alleen GET-requests worden ondersteund. Als een bestand niet gevonden wordt, moet de server de Status-Code 404 teruggeven.

   c. De onderstaande bestandstypen moeten worden ondersteund. In de http-header moeten ook de overeenkomstige velden Content-Type en Content-Length correct meegestuurd worden. Als een bestand wel bestaat maar niet van een ondersteund type is, moet de server de Status-Code 500 teruggeven.

      ✓ html, css;
      ✓ js;
      ✓ txt.

   d. Door gebruik te maken van een BASE64-encoding moeten de volgende binaire bestandsformaten ook ondersteund worden:

      ✓ gif, png
      ✓ png, jpeg, jpg
      ✓ pdf

   e. Als er in de server een niet opgevangen exceptie optreedt, dient deze eveneens een Status-Code 500 terug te geven. In de body moet een stacktrace worden meegestuurd.

   f. Zorg ervoor dat er voor elke request een nieuw proces wordt aangemaakt (een techniek die bekend staat als *spawning* of *forking*). Je kunt hiervoor bijvoorbeeld ProcesBuilder gebruiken.

2. Maak een git repository van de codebase.

3. Demonstreer de werking aan de practicumdocent.

Opgave 5: sessiemanagement

Deze opgave gaat in op sessiemanagement. Binnen verschillende programmeer-omgevingen en -technieken wordt verschillend omgegaan met sessiemanagement. De bedoeling van deze opgave is de manier waarop de toegewezen programmeeromgeving omgaat met sessies.

1. Waarom zijn sessie noodzakelijk in een webapplicatie die bijvoorbeeld de mogelijkheid van een winkelwagen bevat? Gebruik de statelessness HTTP in het antwoord.
2. Je kunt deze statelessness als een nadeel, maar wat zijn de mogelijke voordelen?
3. Beantwoord de volgende vragen over de toegewezen programmeeromgeving:
   a) Waar worden de sessiegegevens bewaard?
   b) Hoe wordt een sessie gestart?
   c) Hoe wordt een sessie beëindigd?
   d) Op welke manier kunnen client-sided gegevens server side in een sessie opgenomen worden?
4. Wat is een cookie in de context van webapplicaties?
5. Waar worden cookies opgeslagen?
6. Maak een pagina waarin een bezoeker via een invoerveld en een knop een tekst aan de sessie toevoegt. Laat de action parameter in de form tag verwijzen naar dezelfde pagina. Ook wordt op dezelfde pagina een lijst getoond van wat er in de sessie staat.
7. Zet de plugin die in opgave 1.3 is geïnstalleerd aan en beschrijf wat er specifiek met betrekking tot sessiemanagement wordt verstuurd.
8. Welke rol spelen cookies in de implementatie van sessiemanagement?

9. Het is belangrijk om doel (sessiemanagement) en middel (cookies als transportmechanisme) in deze context uit elkaar te houden. Zorg ervoor dat cookies in de browser niet meer geaccepteerd worden en verifieer dat het sessiemanagement in vraag h) het niet meer doet.
10. Is er een manier om sessiemanagement mogelijk te maken zonder cookies? Zoek uit hoe dat werk en pas het script van vraag h) aan. Verifieer dat het nu wel werkt, ongeacht of je cookies wel of niet aan hebt staan in je browser.
11. Demonstreer de resultaten van opgaven 6 tot en met 10 aan de practicumdocent.