Versionshantering med Git och GitFlow, samt webbtjänster med HTTP och REST

DA288A – Molnbaserade Webbapplikationer

Johan Holmberg 2019-04-24

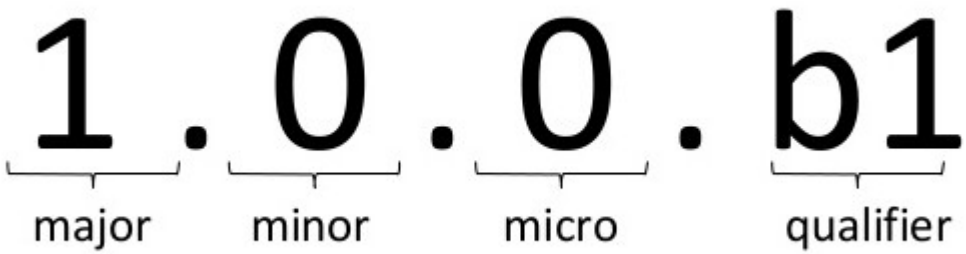
MALMÖ UNIVERSITET

Dagens agenda

- Repetition från tidigare föreläsning
  - Semanstisk versionering
- Versionering med Git
  - Verktyget Git
  - GitHub
  - Gitflow-modellen
- Webbtjänster med HTTP och REST
  - Introduktion till HTTP
  - Introduktion till REST

MALMÖ UNIVERSITET

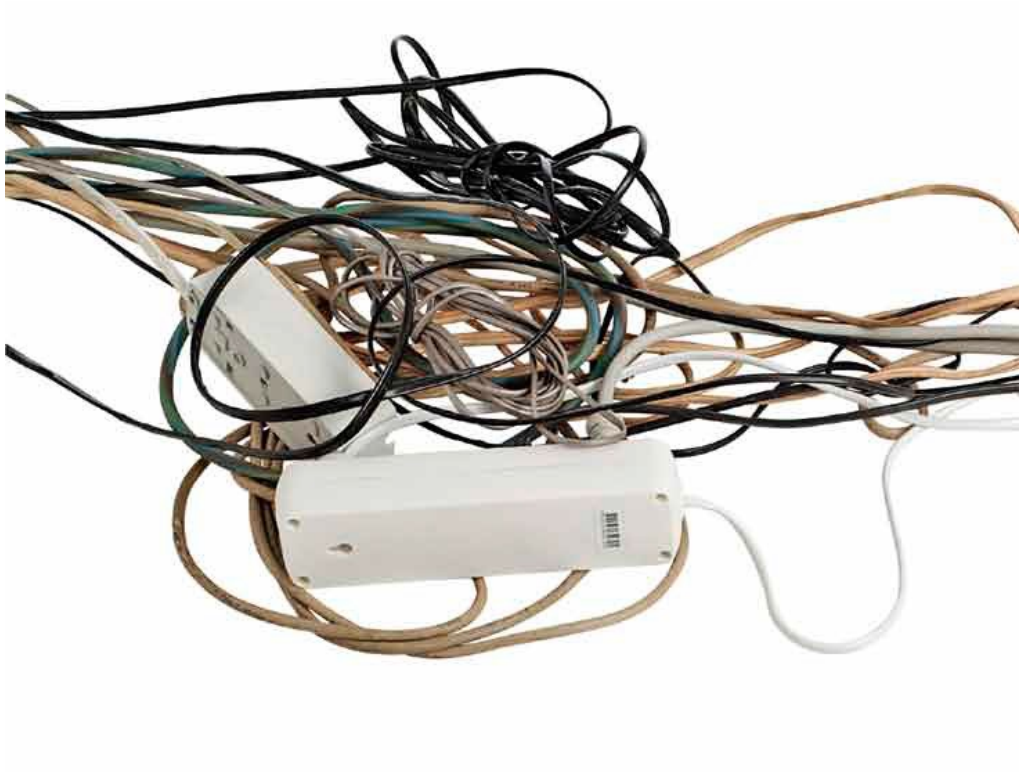
Semantisk versionering



Semantisk versionering: http://semver.org/
Lockart, *Modern PHP*. p. 61

MALMÖ UNIVERSITET

Versionshantering med Git



MALMÖ UNIVERSITET

Versionshantering – varför?

- Samarbete i teamet
  - Förenklar delning av kod och samtida arbete i projektet – även i samma fil (jämför med Google Docs).
  - Olika teammedlemmar kan utveckla flera olika features samtidigt
- Tydlig historik och spårbarhet
  - Återgå till tidigare versioner av projektet
    - Gick något sönder när din kollega lade till kod? Ingen fara, den gamla koden finns kvar!
  - Jämför olika versioner av ett projekt, modul eller fil
  - Vem gjorde vad? Nu vet vi!
- Minska risker
  - Låt inte all källkod leva på en enksild dator
  - Låt inte vem som helst förändra koden

Lockart, *Modern PHP*. p. 157

MALMÖ UNIVERSITET

- Det för tillfället flitigast användna verktyget för versionshantering
- Välprövat
  - Hanterar ohemult stora projekt (exempelvis Linux)
  - Snabbt!
- Hanterar arbetsflöden på ett bra sätt
  - Byggt för att vara bra på att dela upp projekt och slå samman dem igen
    - Fork/pull request och branch/merge
- Distribuerat – ingen central server krävs
  - Pull/Push
- Open Source!

Git: https://git-scm.com
Lockart, *Modern PHP*. p. 157

MALMÖ UNIVERSITET

Terminologi

Repository: En plats för ett projekt, där all källkod finns samlad

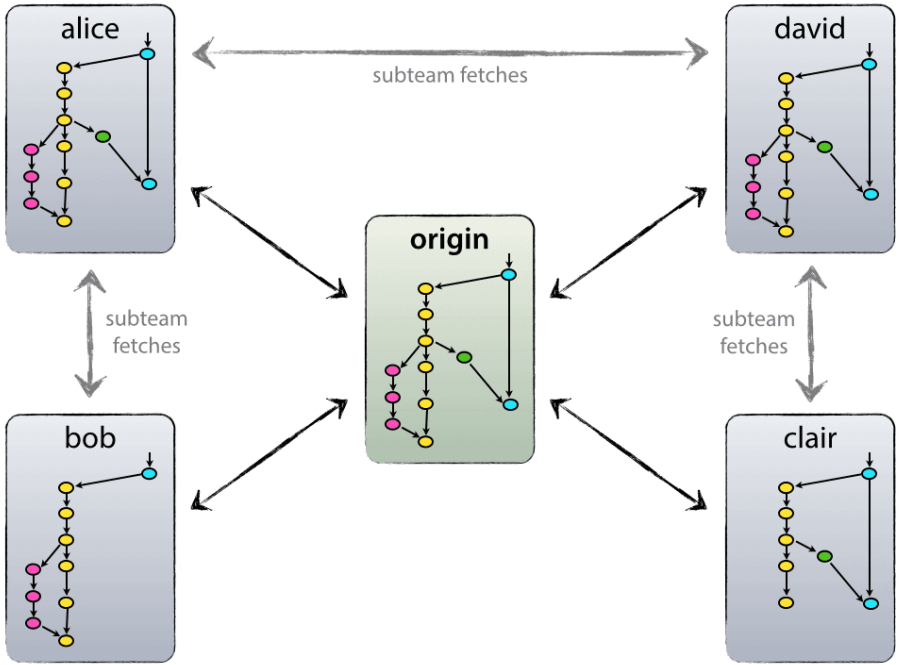
Branch: En avgrening av källkoden

Patch: En föreslagen kodförändring

Merge: Sammanslagning av två branches

Git: https://git-scm.com
Lockart, *Modern PHP*. p. 157

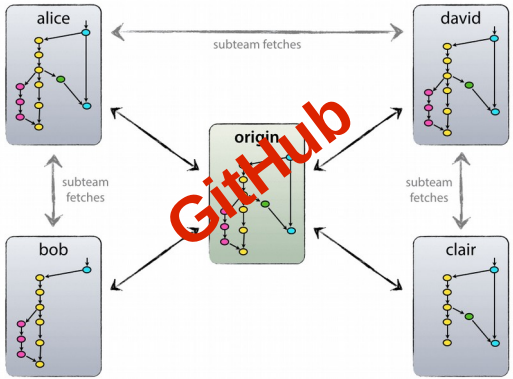
MALMÖ UNIVERSITET



Git: https://git-scm.com
Lockart, *Modern PHP*. p. 157

MALMÖ UNIVERSITET

- En tjänst för lagring och delning av Git-repositories
- Stor i open source-världen
- Erbjuder även kringtjänster såsom wikis och viss ärendehantering



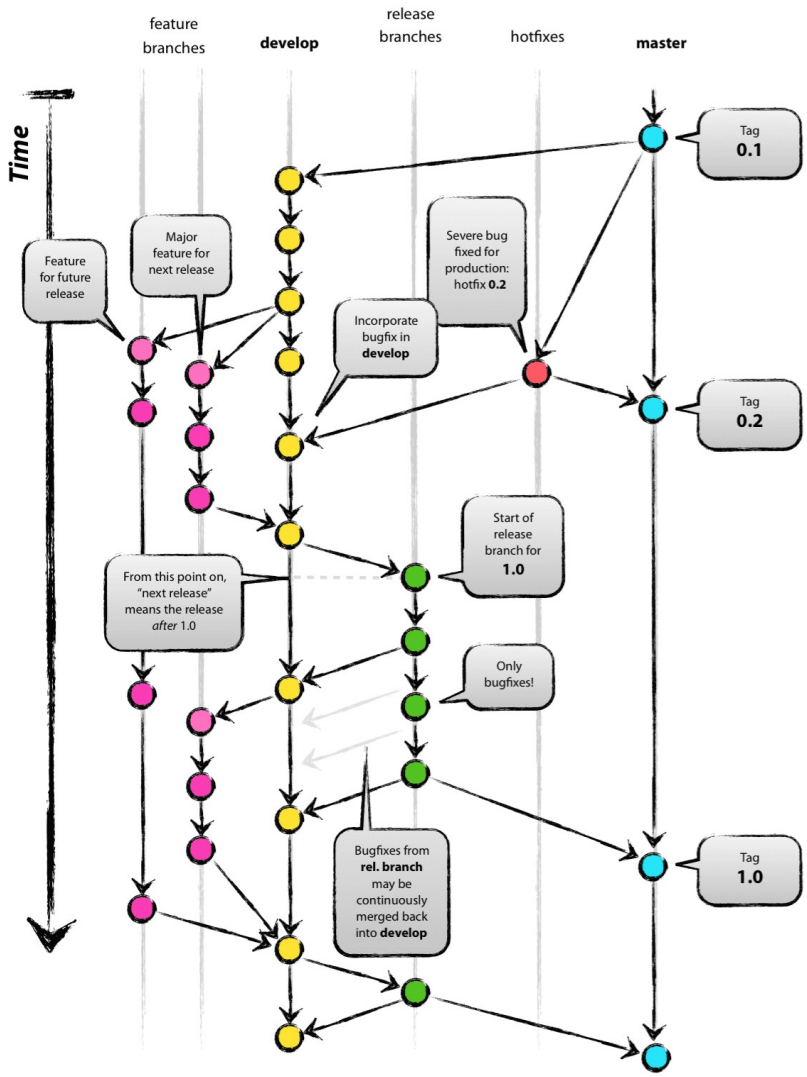
Github: https://github.com

MALMÖ UNIVERSITET

GitFlow-modellen

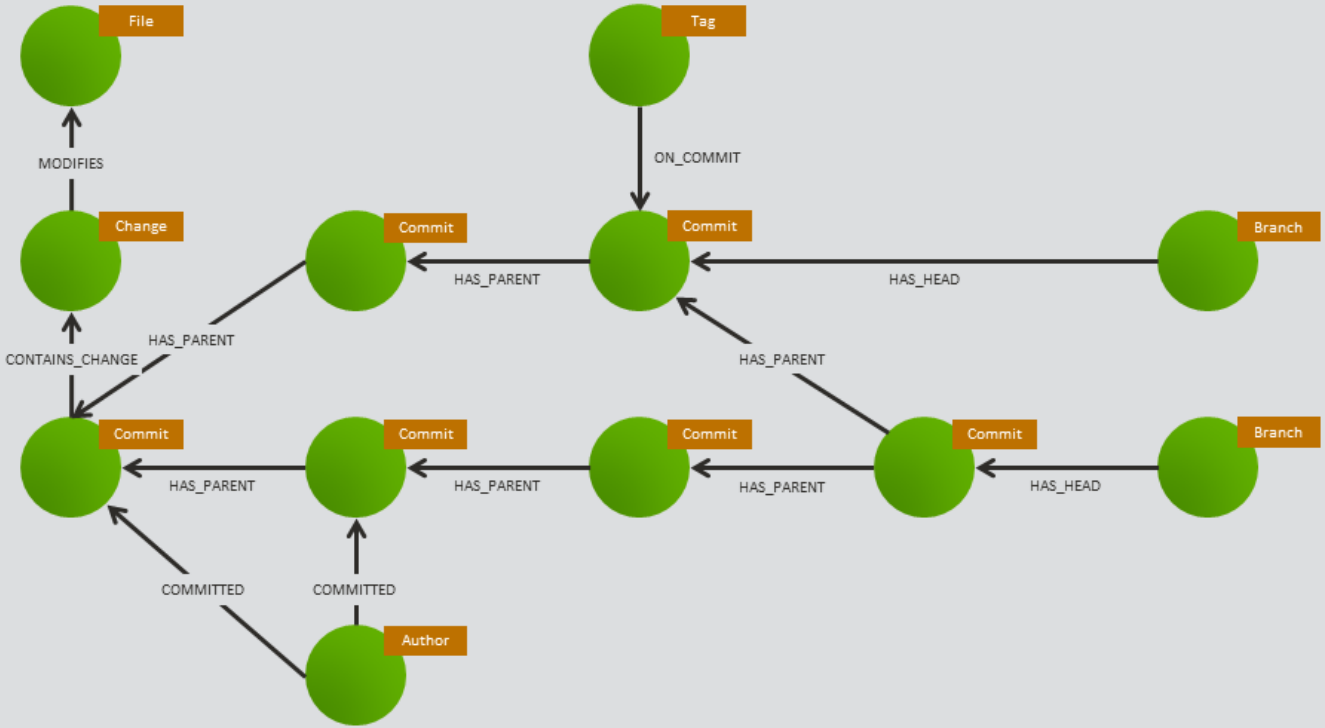
En vanlig modell för att arbeta med Git (Driessen 2010)

- Välanvänt och váltestat
- Använder mycket branching
- Väl anpassat för tighta utvecklingsteam
- Inte lika väl anpassat för större, löst organiserade team
  - Varje utvecklare måste kunna göra en push



Driessen – A successful Git branching model:
http://nvie.com/posts/a-successful-git-branching-model
GitFlow: https://github.com/nvie/gitflow

The Git History Is A Graph!



MALMÖ UNIVERSITET

Grafen

- Varje commit eller revision är en **nod (node)** i en **riktad acyklisk graf (DAG)**
- Varje förändring är en **båge (edge)** i en graf
  - Avgreningar (branches) är bara flera bågar som kommer från en enda nod
  - Sammanslagningar (merges) är bara flera bågar som slutar i samma nod
- Commits kan taggas

Gitflow: CLI-verktyget

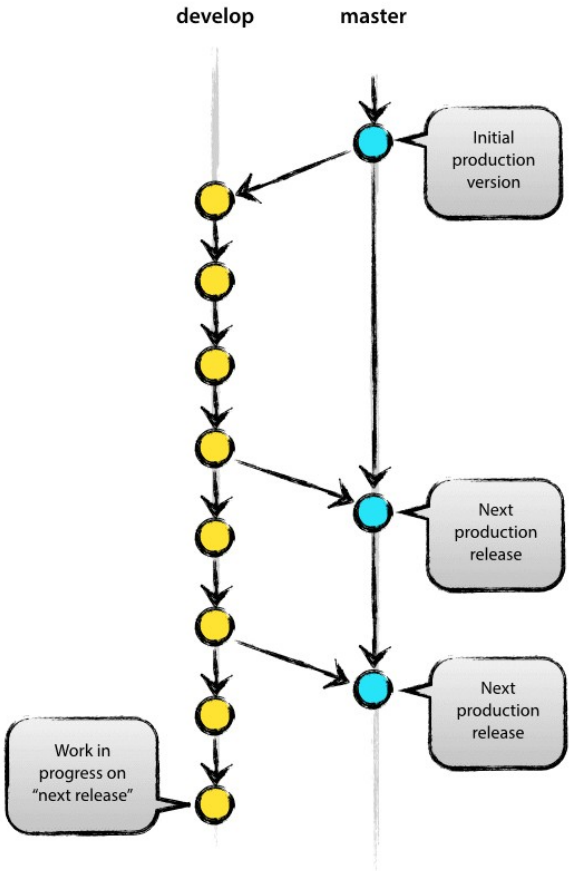
Driessen skapade ett CLI-verktyget för att förenkla användandet av Gitflow, kallat *git-flow*

- Tillgängligt för Windows, macOS, Linux, etc.
- Underhålls inte längre
- Nuvarande “aktiva” version (en fork!): https://github.com/petervanderdoes/gitflow-avh

GitFlow-modellen

Utgår från två branches:

- **master** – kod i produktion
- **develop** – kod under utveckling

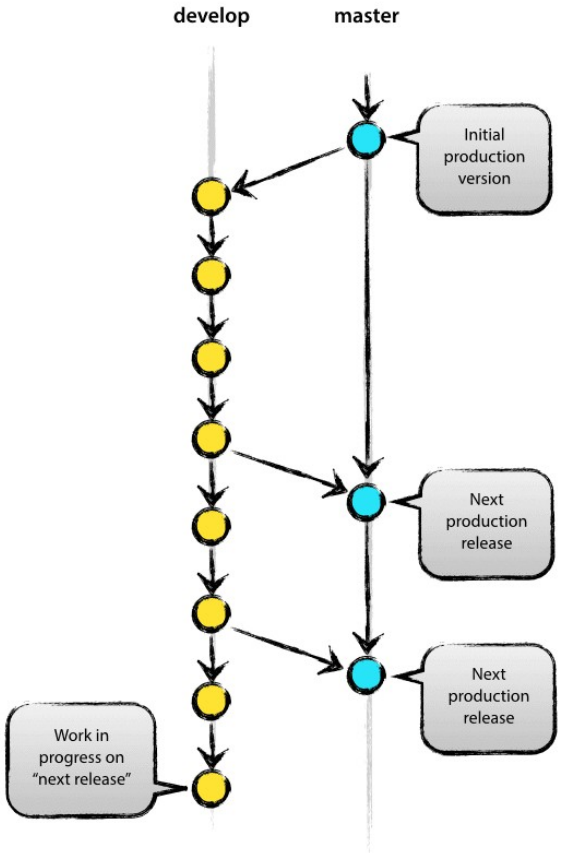


Driessen – A successful Git branching model:
http://nvie.com/posts/a-successful-git-branching-model
GitFlow: https://github.com/nvie/gitflow

master och *develop*

De två eviga avgreningarna

- *master*
  - Den “orörbara” grenen, här sker ingen utveckling
  - Fungerar som en samling av releaser
    - Kod i produktion!
- *develop*
  - Den “smutsiga” grenen, all utveckling sker här
  - Avgrenad från den första commiten till *master*
  - Används som bas för (nästan) alla andra grenar



MALMÖ UNIVERSITET

master och *develop*

Initiera ett repository

- Med *git-flow*

```
git flow init
```

- Standard git

```
git init
git branch develop
git push –u origin develop
```

MALMÖ UNIVERSITET

Stödjande grenar

Alla andra grenar är tillfälliga:

- Uppmuntrar separation-of-concerns
- Förhindrar att för mycket görs i en gren
- Håller historiken hyfsat ren och förståbar

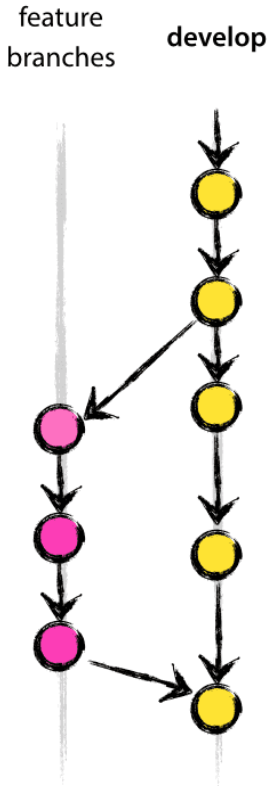
Viktigt! Alla grenar är tekniskt sett lika. Allt detta är bara mänskliga konventioner.

Feature branches

Används för ny, mer omfattande funktionalitet, även för framtida releaser

Viktiga punkter:

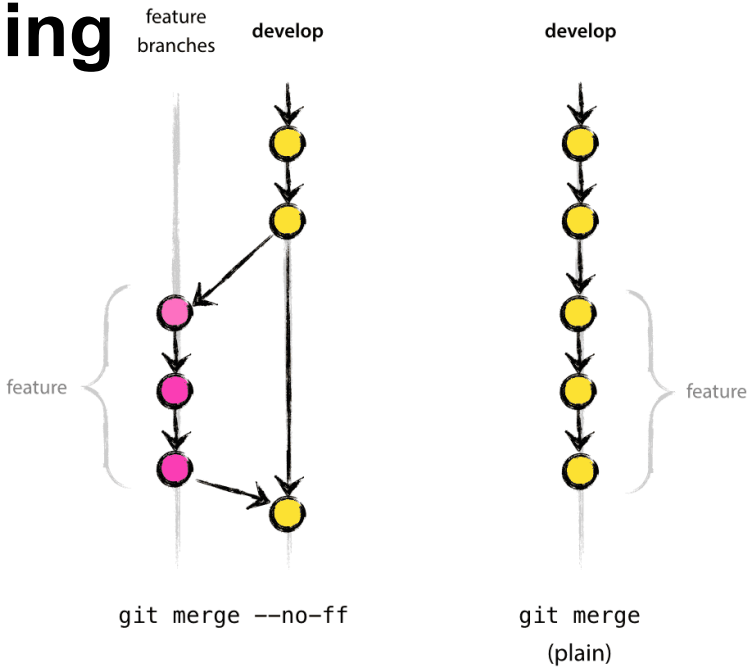
- Avgrenas alltid från *develop*
- Slås alltid samman med *develop*
- Kan heta vad som helst (Johan föreslår *feature/<description>*, t.ex. *feature/login*)



MALMÖ UNIVERSITET

Feature branches: merging

- *--no-ff* (“no fast forward”) behåller historik och trädets grenar
- Fast forward håller trädet tunt, men förlorar historisk mening
  - Svårare att följa förändringar
  - Svårare att gå tillbaka i historiken



MALMÖ UNIVERSITET

Feature branches

Skapa en feature-branch

- Med *git-flow*

```
git flow feature start branch_name
```

- Standard git

```
git checkout develop
git checkout -b branch_name
```

MALMÖ UNIVERSITET

Feature branches

Avsluta en feature-branch

- Med *git-flow*

```
git flow feature finish branch_name
```

- Standard git

```
git checkout develop
git merge --no-ff branch_name
```

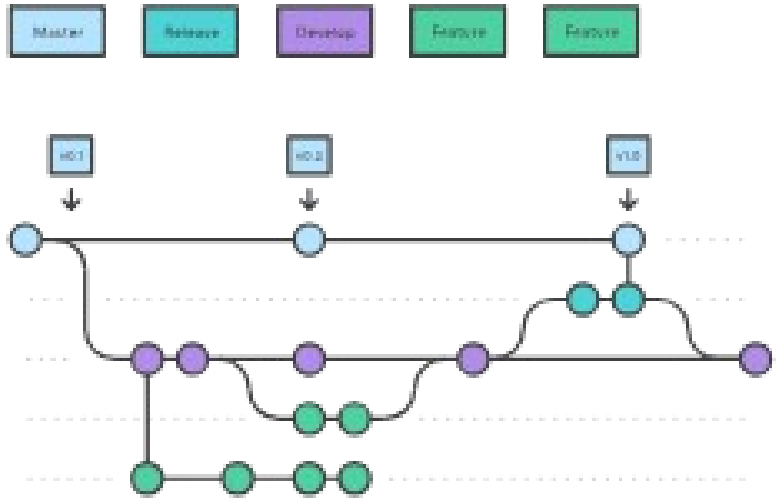
MALMÖ UNIVERSITET

Release branches

Används för att hantera releaser

Viktiga punkter:

- Avgrenas alltid från *develop*
- Slås alltid samman med *develop* och *master*
- Bör alltid heta *release/<identifier>*, t.ex. *release/1.0*



MALMÖ UNIVERSITET

Release branches

Skapa en release-branch

- Med *git-flow*

```
git flow release start identifier
```

- Standard git

```
git checkout develop
git checkout -b release/identifier
```

MALMÖ UNIVERSITET

Release branches

Avsluta en release-branch

- Med *git-flow*

```
git flow release finish identifier
```

- Standard git

```
git checkout master
git merge --no-ff release/identifier
git tag -a identifier
git checkout develop
git merge --no-ff release/identifier
git branch -d release/identifier
```

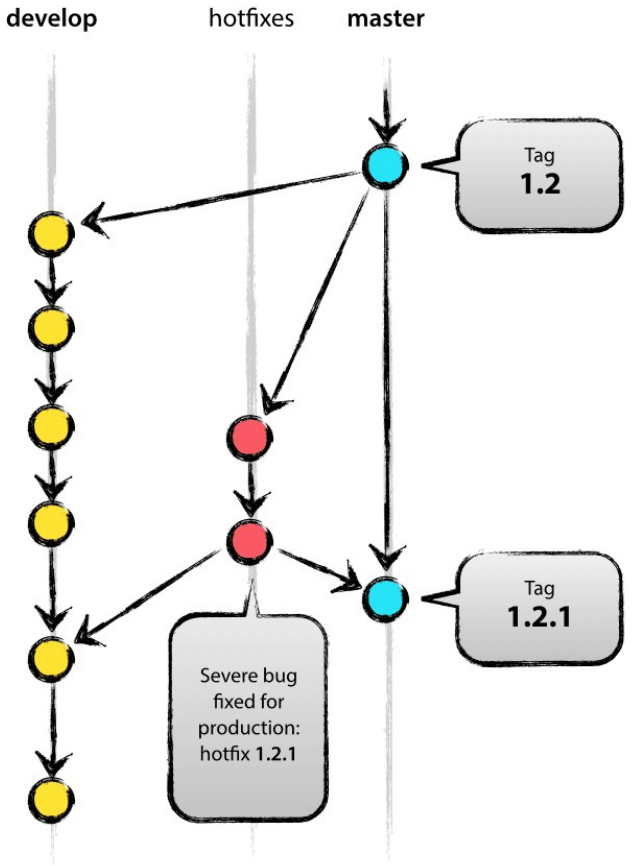
MALMÖ UNIVERSITET

Hotfix branches

Används för att fixa kritiska fel i produktionskod

Viktiga punkter:

- Avgrenas alltid från *master*
- Slås alltid samman med *master* och *develop* OM DET INTE finns en aktuell release-gren. Om det finns: slå samman med *master* och *release*
- Bör alltid heta hotfix*/<identifier>*, t.ex. hotfix*/1.0.1*



MALMÖ UNIVERSITET

Hotfix branches

Skapa en hotfix-branch

- Med *git-flow*

```
git flow hotfix start identifier
```

- Standard git

```
git checkout master
git checkout -b hotfix/identifier
```

MALMÖ UNIVERSITET

Hotfix branches

Avsluta en hotfix-branch

- Med *git-flow*

```
git flow hotfix finish identifier
```

- Standard git

```
git checkout master
git merge --no-ff hotfix/identifier
git tag -a identifier
git checkout develop
git merge --no-ff hotfix/identifier
git branch -d hotfix/identifier
```

MALMÖ UNIVERSITET

Kritik mot Gitflow

- Kan vara svårt för nya användare
- Lite för komplext för många fall (Samokhin, 2018, Ruka, 2017)
- Långlivade grenar tenderar att leda till integrationsproblem (Samokhin, 2018)
- “Buskig” historik kan vara svår att följa (Ruka, 2017)

Samokhin, Vadim. *Gitflow is a Poor Branching Model Hack*. 2018. Source: https://hackernoon.com/gitflow-is-a-poor-branching-model-hack-d46567a156e7
Ruka, Adam. *Gitflow considered harmful*. 2017. Source: https://www.endoflineblog.com/gitflow-considered-harmful

Mer om Gitflow

- En piffig latund:
  http://danielkummer.github.io/git-flow-cheatsheet/
- GitHubs genomgång av Git/Github:
  https://guides.github.com/activities/hello-world/
- Atlassians genomgång av Gitflow:
  https://www.atlassian.com/git/tutorials/comparing-workflows/gitflow-workflow

MALMÖ UNIVERSITET

Webbtjänster med HTTP och REST

MALMÖ UNIVERSITET

Vad är HTTP?

- Det nätverksprotokoll som ligger till grund för kommunikation över World Wide Web.
    - Använder TCP för överföring.
- Kan överföra data oavsett innehållstyp

HTTP-specifikationen: https://tools.ietf.org/html/rfc2616
Hypertext Transfer Protocol enligt Wikipedia:
https://en.wikipedia.org/wiki/Hypertext_Transfer_Protocol

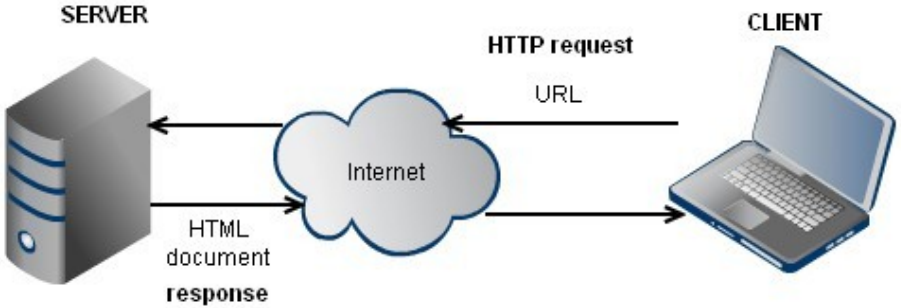
Terminologi

- **IP-adress** – en adress som identifierar en specifik maskin (fysisk eller virtuell) på ett nätverk.
- **TCP (Transmission Control Protocol)** – ett protokoll för överföring av data över ett nätverk. Protokollet garanterar att datan som sänds är korrekt överförd.
- **Portnummer** – ett nummer som läggs till en IP-adress för att adressera en specifik applikation på en maskin.
- **URL (Uniform Resource Locator)** – ett sätt att referera till en resurs på www.

Internet protocol suite enligt Wikipedia:
https://en.wikipedia.org/wiki/Internet_protocol_suite

Hur kommunikation över HTTP fungerar

- Två program – en klient (exempelvis en webbläsare) och en server (exempelvis Apache).
- Klienten anropar servern över en TCP-koppling med hjälp av HTTP-anrop.



En HTTP-session

IP-adress: 192.168.1.42
Port: 50222

www.mau.se
IP-adress: okänd
Port: 80

http://www.mau.se  ?

MALMÖ UNIVERSITET

En HTTP-session

IP-adress: 192.168.1.42
Port: 50222

Ge mig IP-adressen
för www.mau.se

DNS-server hos Telia
IP-adress: 195.198.26.35
Port: 53

MALMÖ UNIVERSITET

En HTTP-session

Ge mig IP-adressen
för www.mau.se

www.mau.se:
195.178.227.58

IP-adress: 192.168.1.42
Port: 50222

DNS-server hos Telia
IP-adress: 195.198.26.35
Port: 53

MALMÖ UNIVERSITET

En HTTP-session

IP-adress: 192.168.1.42
Port: 50222

www.mau.se
IP-adress: 195.198.26.35
Port: 80

http://www.mau.se → 195.198.26.35

MALMÖ UNIVERSITET

En HTTP-session

GET / HTTP/1.1

HTTP/1.1 200 OK
..

<html>..</html>

IP-adress: 192.168.1.42
Port: 50222

www.mau.se
IP-adress: 195.198.26.35
Port: 80

http://www.mau.se

195.198.26.35

MALMÖ UNIVERSITET

HTTP-verb/metoder

Verb/metod	Vad händer?
GET	Läser en resurs
POST	Skapar en ny resurs
PUT	Skapar en ny resurs eller uppdaterar en befintlig resurs
DELETE	Raderar en resurs
OPTIONS	Listar godkända operationer för en resurs
HEAD	Returnerar endast headers vid anrop

Metadata – Request Headers

```
GET http://www.w3.org/Protocols/rfc2616/rfc2616.html HTTP/1.1
Host: www.w3.org
Accept: text/html,application/xhtml+xml,application/xml; …
User-Agent: Mozilla/5.0 (Windows NT 6.3; WOW64) AppleWebKit/537.36 …
Accept-Encoding: gzip,deflate,sdch
Accept-Language: en-US,en;q=0.8,hi;q=0.6
```

```
HTTP/1.1 200 OK
Date: Sat, 23 Aug 2014 18:31:04 GMT
Server: Apache/2
Last-Modified: Wed, 01 Sep 2004 13:24:52 GMT
Accept-Ranges: bytes
Content-Length: 32859
Cache-Control: max-age=21600, must-revalidate
Expires: Sun, 24 Aug 2014 00:31:04 GMT
Content-Type: text/html; charset=iso-8859-1
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN" "http://www.w3.org
<html xmlns='http://www.w3.org/1999/xhtml'>
<head><title>Hypertext Transfer Protocol -- HTTP/1.1</title></head>
<body>
...
```

MALMÖ UNIVERSITET

Svarskoder

Exempel:

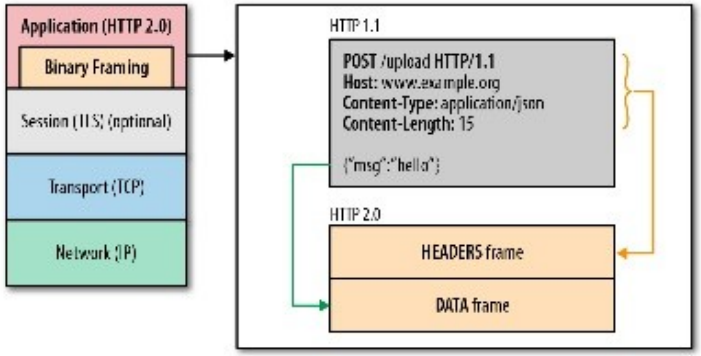
- 200 – OK
- 201 – Created
- 404 – Not Found
- 405 – Method Not Allowed
- 500 – Internal Server Error
- https://sv.wikipedia.org/wiki/Lista_%C3%B6ver_HTTP-statuskoder

MALMÖ UNIVERSITET

HTTP/2 – den moderna versionen

HTTP/2 in one slide...

- One TCP connection
- Request → Stream
  - Streams are multiplexed
  - Streams are prioritized
- Binary framing layer
  - Prioritization
  - Flow control
  - Server push
- Header compression



HTTP/2-specifikationen: https://tools.ietf.org/html/rfc7540
HTTP/2 enligt Wikipedia: https://en.wikipedia.org/wiki/HTTP/2

MALMÖ UNIVERSITET

Vad är REST?

MALMÖ UNIVERSITET

Vad är REST?

- **Re**presentational **S**tate **T**ranser
- En arkitektonisk stil för att designa nätverksbaserade applikationer
- Syftar till att förenkla datautbyte mellan olika appikationer
    - Använd naken HTTP istället för exempelvis SOAP eller mer komplexa tekniker
        - HTTP-standarden definierar metoderna *Get*, *Post*, *Put* och *Delete* för CRUD-operationer
        - HTTP-standarden definierar ett antal statuskoder för att förmedla resursens tillstånd

REST som webbtjänst

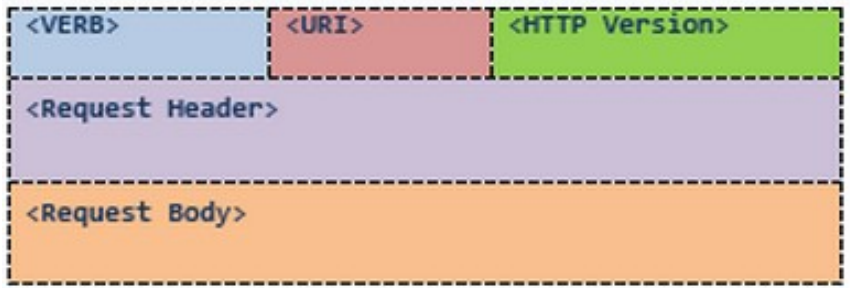
- Som de flesta andra tekniker för webbtjänster (exempelvis Web Services) är REST
  - Plattformsoberoende
  - Språkoberoende
  - Byggt på vedertagna standarder (HTTP)
- REST erbjuder exempelvis inte säkerhet, kryptering eller sessionshantering, men detta kan realiseras:
  - Autentiering med REST görs ofta med tokens
  - Kryptering med REST görs lättast med HTTPS
  - Sessionshantering görs ofta med cookies och tokens

REST som webbtjänst (forts)

- **ST** i REST står för ***S**tate **T**ransfer*, vilket innebär
  - Varje anrop är självständigt och innehåller all information som behövs för att genomföra en förfrågan → den är idempotent

REST – Meddelanden

- Klient och server pratar med varandra genom meddelanden
  - Det är klienten som initierar kommunikationen
- Klienten skickar en förfrågan till servern genom att
  - Data skickas som Request Body
  - Metadata skickas som Request Headers



MALMÖ UNIVERSITET

Enkla REST-anrop

Lägg märke till att anropen görs med vanliga URLer

```
http://www.google.com/search?q=horses
```

MALMÖ UNIVERSITET

Mer komplexa REST-anrop

- Förra exemplet använde bara en parameter (en sökterm)
  - Det är enkelt att bygga in fler parametrar i sina anrop

```
http://www.acme.com/phonebook/UserDetails?firstName=John&lastName=Doe
```

- Behöver man skicka många parametrar eller mycket data (exempelvis en fil) används metoden POST
- Tumregler:
  - För att läsa data – använd GET, som inte ska förändra tillståndet hos en resurs
  - För att skriva data – använd POST, PUT eller DELETE.

MALMÖ UNIVERSITET

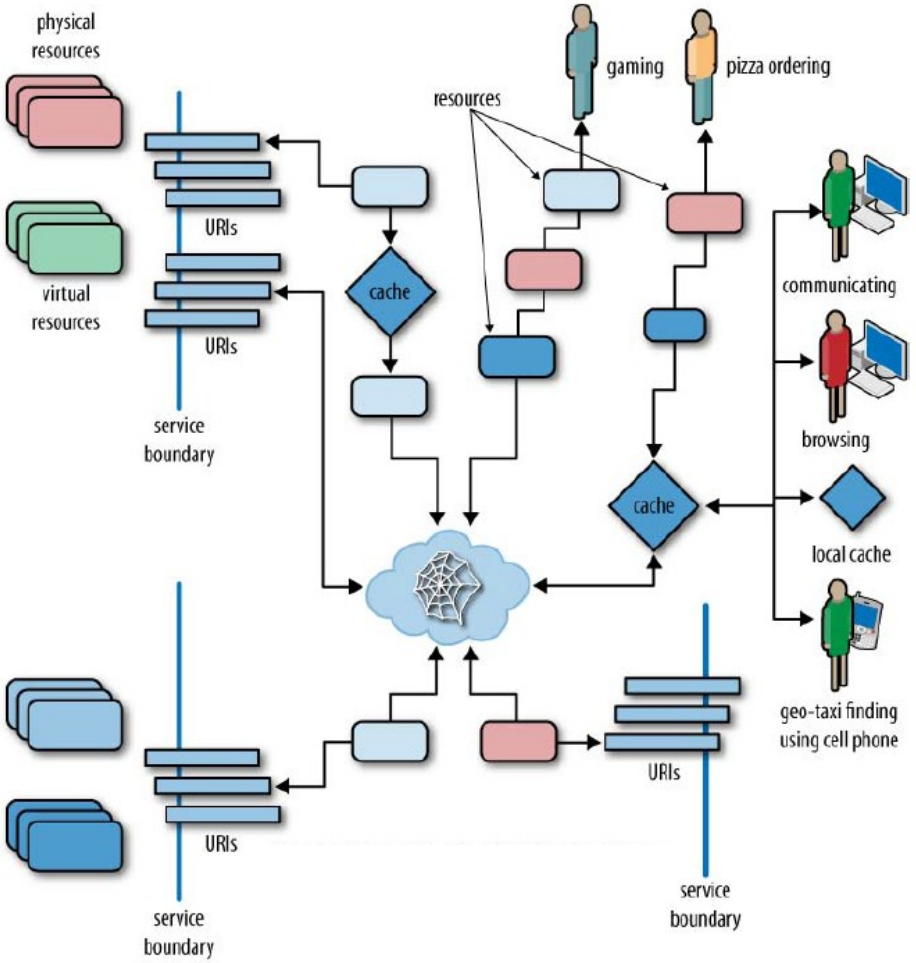
REST på fem fingrar

- **Resurser**, snarare än tjänster, som identifieras med URLer
- **En webb av resurser**, där en resurs inte behöver ge svar på allt utan glatt kan länka vidare till andra resurser
- **Klient** – Server-modell
- **Tillståndslöst (stateless)**, varje anrop är idempotent
- **Cachningsbara resurser**

Resurser
Identifierare

Representationer

HTTP



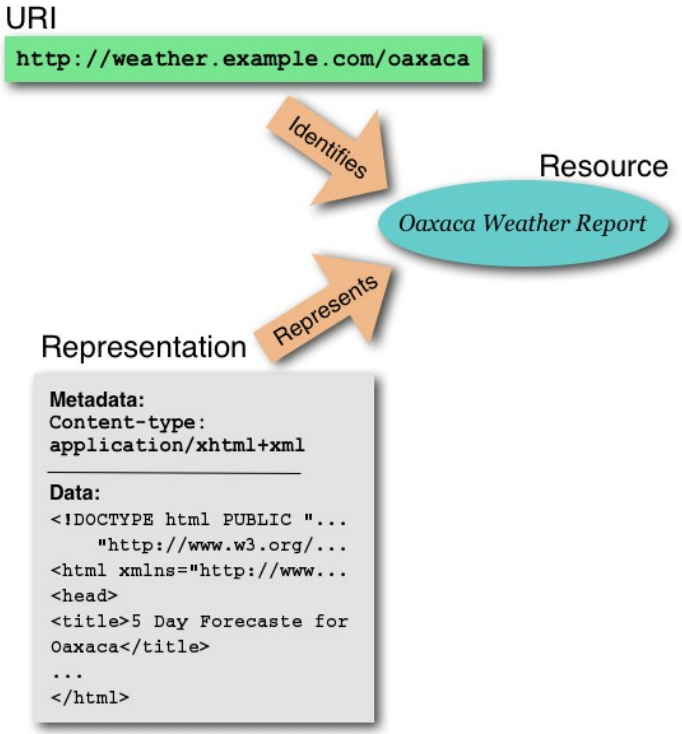
Resurser

Vad är en resurs?

“A **resource** is **anything we expose** to **the Web**,
from a **document or video clip**
to a **business process or device**.
From a **consumer’s point of view**,
a resource is anything with
which that consumer interacts
while progressing toward some goal.”

Webber et al. *REST in Practice*

Resurser på webben



MALMÖ UNIVERSITET

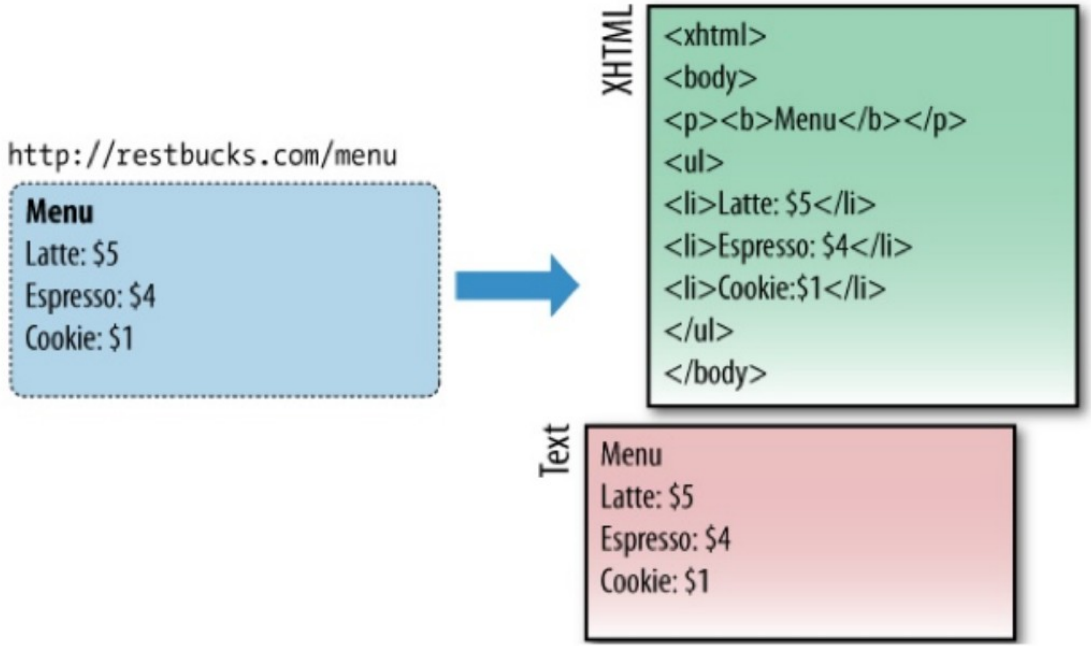


Figure 1-3. *Example of a resource and its representations*

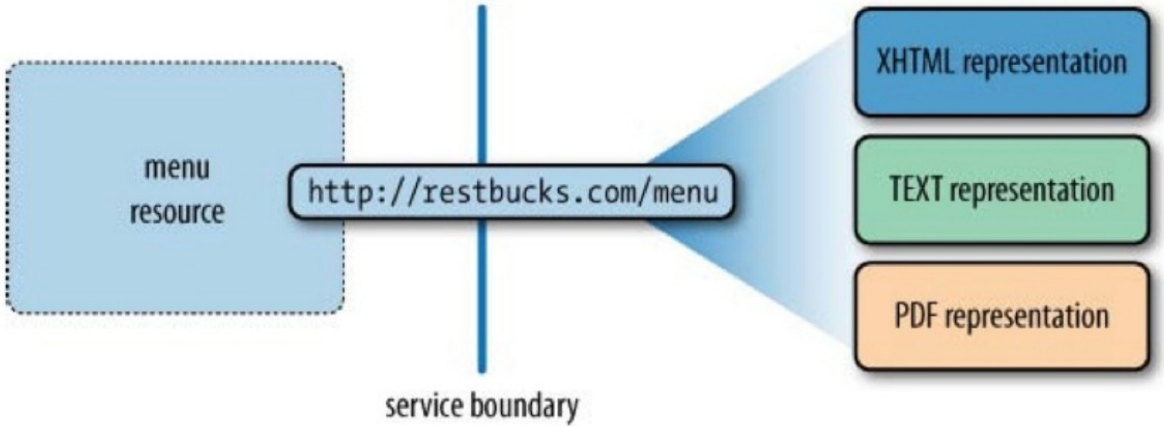


Figure 1-5. *Multiple resource representations addressed by a single URI*

Identifierare

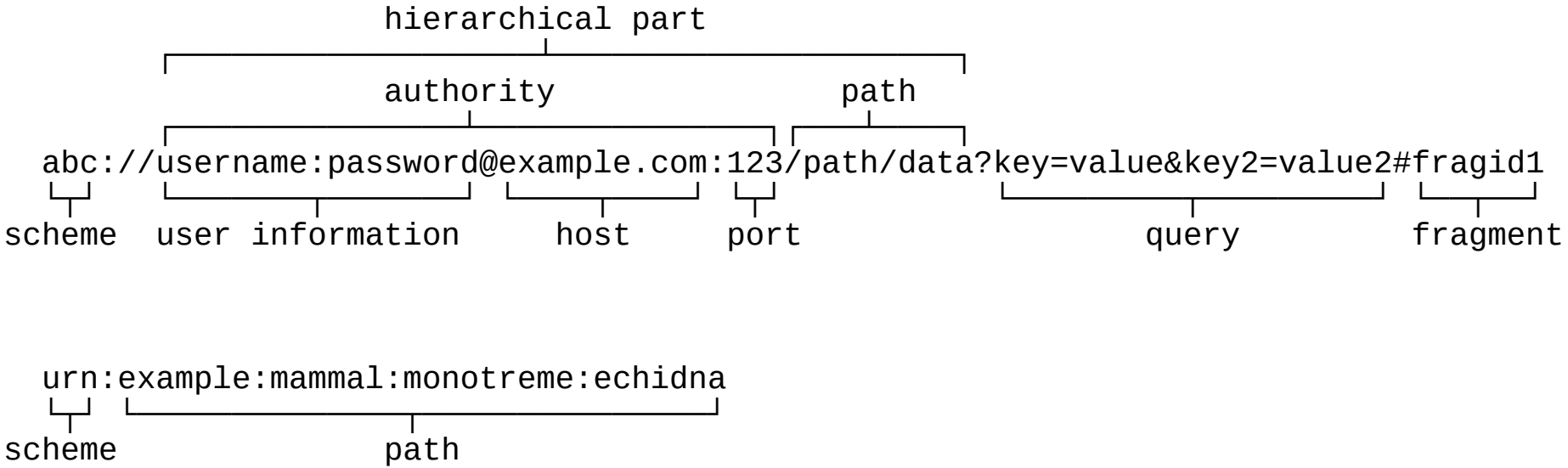
URI	Uniform Resource Identifier
URL	Uniform Resource Locator

MALMÖ UNIVERSITET

URI

<scheme>:<scheme-specific-structure>

URI



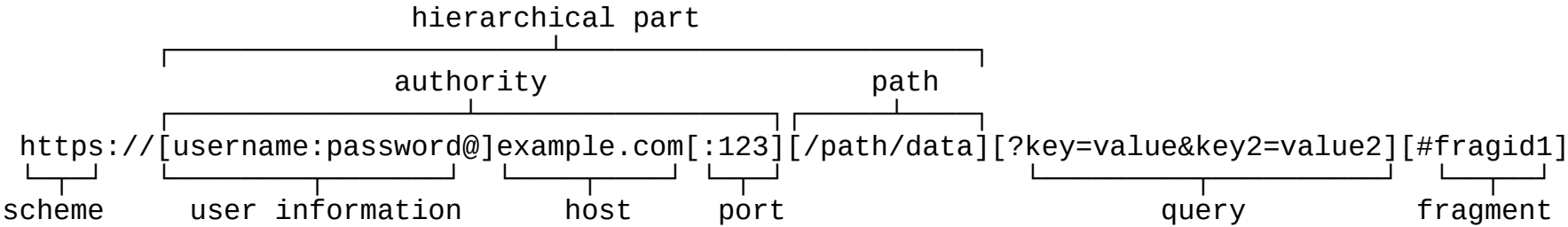
MALMÖ UNIVERSITET

Exempel på URI:er

- ISBN
  - isbn:978-0-33-050811-7
- Mailto Scheme URI (e-post)
  - mailto:johan.holmberg@mau.se
- Tel URI (telefoni)
  - tel:+421-2-529-263-67
- Spotify URI
  - spotify:user:idioti.se:playlist:4xvCr0XKcPCJqzAYTHIR3E
- HTTP Scheme URI (URL)
  - http://www.mau.se/dk

URL

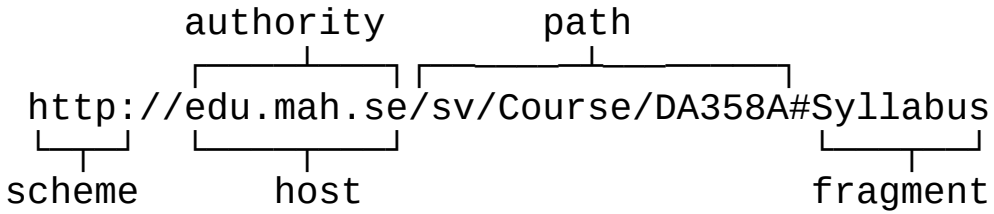
HTTP-schemat används för att lokalisera resurser på nätverket via HTTP-protokollet.



MALMÖ UNIVERSITET

Exempel på URL

Lägg märke till att endast *scheme* och *host* är obligatoriska delar



MALMÖ UNIVERSITET

Representationer

MALMÖ UNIVERSITET

REST och representation

- Resurser och hur vi når dem är RESTs fokus
- Representationen är ett sätt att beskriva informationen hos en resurs
- Resursen kan bestå av eller bero på andra resurser
  - Representationen bör kunna hantera detta
- Resursens representation är inte definierad i någon standard
  - En resurs kan representeras av flera olika format

Olika sätt att representera en resurs

Ada Lovelace

Ada Lovelace [redigera | redigera wikitext]

Ada Lovelace, egentligen *Augusta Ada King, grevinna av Lovelace*, fö
matematiker och skribent. Hon är mest hågkommen för sitt arbete mec
maskinen innehåller den första algoritm som är avsedd att bearbetas r

Lovelace var enda barn till poeten Lord Byron och dennes fru Anne Isa
England fyra månader senare; han dog till slut i sjukdom under Grekla
främjade dotterns intresse för matematik och logik i ett försök att förhin
sitt intresse för sin far och begravdes enligt sin sista vilja vid sidan av h

Ada Lovelace beskrev sitt arbetssätt som "poetisk vetenskap"[5] och si
talanger henne fram till ett långvarigt samarbete med en annan brittisk
maskinen". Åren 1842–43 översatte hon en artikel om maskinen skrive
notapparat. Dessa noter innehåller vad många[vilka?] anser vara det fö
producera siffror.[7] Hennes "poetiska vetenskap" drev henne fram till fi
samhälle å ena sidan och tekniken som ett redskap å den andra.[8]

Innehåll [dölj]

1 Biografi
- 1.1 Barndom
- 1.2 Vuxna år

```
### Ada Lovelace

#### Inledning

Ada Lovelace, egentligen Augusta Ada King, gr
december 1815 i London, död där 27 november 1
skribent. Hon är mest hågkommen för sitt arbe
dator, den analytiska maskinen. Hennes antecl
första algoritm som är avsedd att bearbetas m
som historiens första datorprogrammerare.

Lovelace var enda barn till poeten Lord Byron
Makarna separerade en månad efter att dotter
månader senare; han dog till slut i sjukdom u
var åtta år gammal. Adas mor var bitter på Lo
intresse för matematik och logik i ett försök
utvecklade den galenskap som hon såg hos barn
för sin far och begravdes enligt sin sista vi

### Biografi

#### Barndom

Ada Lovelace föddes 10 december 1815 som dott
baronen av Byron, och Anne Isabella "Annabel"
George Byron förväntade sig att hans nya ätt
```

MALMÖ UNIVERSITET

Vanliga format

- HTML
  - Används oftast för att ge ett visuellt svar – en webbsida
- XML
  - Används ofta när detaljer, såsom datatyper, är viktiga
  - Mycket vanligare förr i tiden
- JSON (**J**ava**S**cript **O**bject **N**otation)
  - Mer kompakt än XML, sparar bandbredd
  - Ofta mer läsbart för människor
- CSV, Excel, PDF, PNG, etc.

XML och JSON

```xml
<?xml version="1.0"?>
<unicorn>
    <id>4</id>
    <name>Tordönsenhörning</name>
    <description>Det här är en irriterand
        tvivelaktiga nöjet att stifta bek
        .</description>
    <reportedBy>Johan</reportedBy>
    <spottedWhere>
        <name>Söderbärke, Sverige</name>
        <lat>59.9801</lat>
        <lon>15.4446</lon>
    </spottedWhere>
    <spottedWhen>2010-05-08 00:05:00</spo
    <image>http://unicorns.idioti.se/bild
</unicorn>
```

```json
{
  "id": "4",
  "name": "Tordönsenhörning",
  "description": "Det här är en irriterande ras.
      tvivelaktiga nöjet att stifta bekantskap me
  "reportedBy": "Johan",
  "spottedWhere": {
    "name": "Söderbärke, Sverige",
    "lat": "59.9801",
    "lon": "15.4446"
  },
  "spottedWhen": {
    "date": "2010-05-08 00:00:00.000000",
    "timezone_type": 3,
    "timezone": "UTC"
  },
  "image": "http://unicorns.idioti.se/bilder/tor.
}
```

MALMÖ UNIVERSITET

Att begära ett format

De flesta REST-tjänster som erbjuder mer än ett format gör det på något av följande sätt:

- Genom sökvägen (del av URL:en, usch):
  - http://www.example.com/**json**/horses
- Genom queries (söksträngar i URL:en, meh):
  - http://www.example.com/horses?**format=json**
- Genom headers (helt klart vackrast):
  - http://www.example.com/horses
  - **Accept: application/json**

MALMÖ UNIVERSITET

Exempel på olika format

Vi använder Johans eminenta enhörnings-API

MALMÖ UNIVERSITET