Ajax

Asynchronous JavaScript and XML

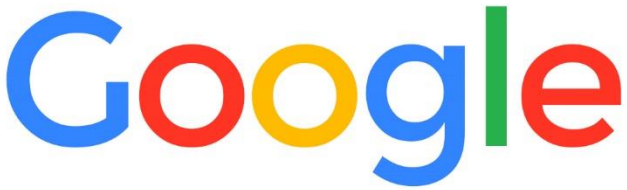
Dagens agenda

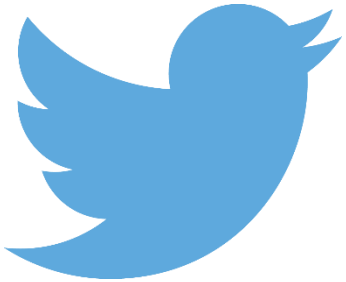
- Vad innebär ajax?
- Hur använder man ajax?
- Exempel på ajax
- Att bygga en enkel tjänst med ajax

*”**AJAX** (en. Asynchronous JavaScript and XML) är ett samlingsnamn för flera olika tekniker som kan användas för att bygga applikationer för World Wide Web med bättre interaktivitet än tidigare webbapplikationer.”*


Web 2.0
Aggregators
Folksonomy
Wikis
Blogs
User Centered
Joy of Use
Participation
Six Degrees
Usability
Pagerank
XFN
Widgets
Recommendation
Social Software
FOAF
Browser
Simplicity
Sharing
Collaboration
Perpetual Beta
Videocasting
Podcasting
AJAX
Audio
IM
Video
Design
Convergence
CSS
Pay Per Click
UMTS
Mobility
Atom
XHTML
SVG
Ruby on Rails
VC
Trust
Affiliation
OpenAPIs
RSS
Semantic
Web Standards
SEO
Economy
OpenID
Remixability
REST
Standardization
The Long Tail
DataDriven
Accessibility
XML
Microformats
Syndication
Modularity
SOAP


AJAX
Asynchronous Javascript And XML


Google




YouTube

Exempel: Google

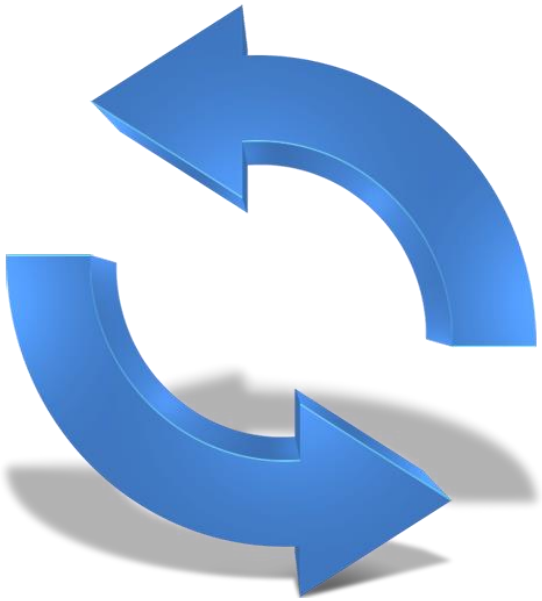
https://www.google.se

”…för flera olika tekniker”

- XMLHttpRequest-objektet, som tillåter JavaScript på en webbsida att göra anrop till en webbserver/filer utan att sidan laddas om.
  - Detta kan var både inom den egna webbplatsen, eller till externa webbplatser
- DOM (Document Object Model) vilket tillåter JavaScript att skriva om innehåll på den aktuella sidan.
- HTML och CSS vilket beskriver utseende för, och märker, innehållet på sidan.
- XML, JSON, m.m. som används för att formatera data som transporteras mellan klient och server. Vanlig text kan också användas som format.

Problem: hämta/skicka information utanför vår html-sida

Lösning: Ajax (oavsett om det är information från vår egen webbplats, eller från annan webbplats)



”Att interaktivt ändra innehållet på vår webbplats”

Från olika data-källor

Exempelområden

- Autocomplete (http://google.com)
  - Vid sökningar av diverse information
- Flerstegsformulär (http://demo.tutorialzine.com/2011/11/chained-ajax-selects-jquery/)
- ”Realtidssystem”
  - T.ex. chattsystem, auto-uppdateringar, etc.
- Direktvalidering av formulär
- Autosparning av information
- Röstningssystem
- Etc.

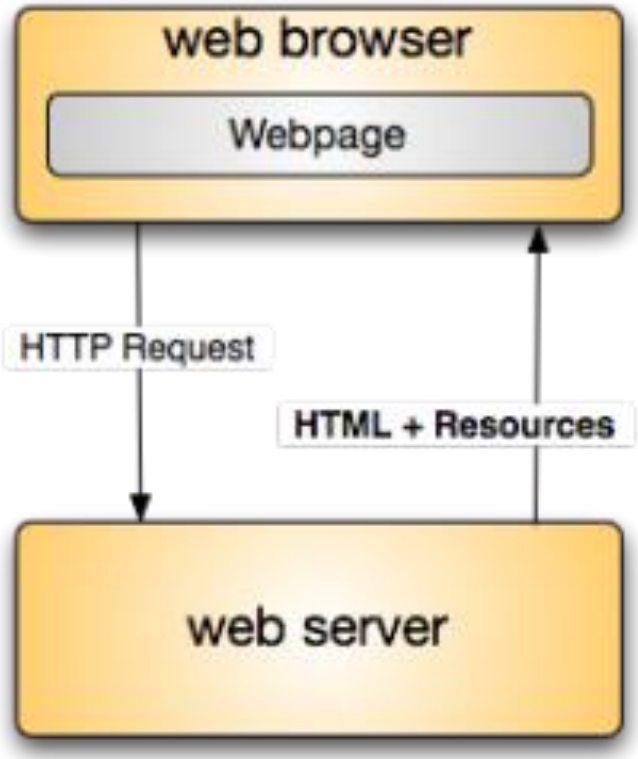
Fördelar med Ajax

- Snabbar upp användandet av webbplatser
  - Ladda bara om den information vi behöver – inte allt
- Färre siduppdateringar på webbplatsen
- Reducerar nätverkstrafik
- Minskar serverbelastning
- Ger ett mer interaktivt GUI

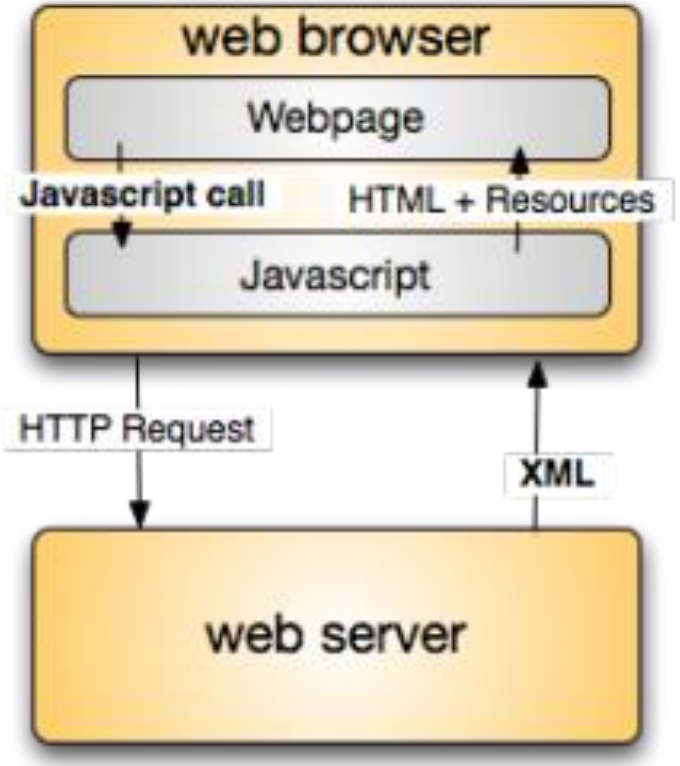
Nackdelar med Ajax

- Bokmärken slutar fungera (går att lösa i många fall)
- Sökmotorer indexerar inte innehållet som laddas genom ajax
- ”Bakåtknappen” slutar att fungera
- JavaScript kan enkelt inaktiveras från klientsidan

Traditional web model



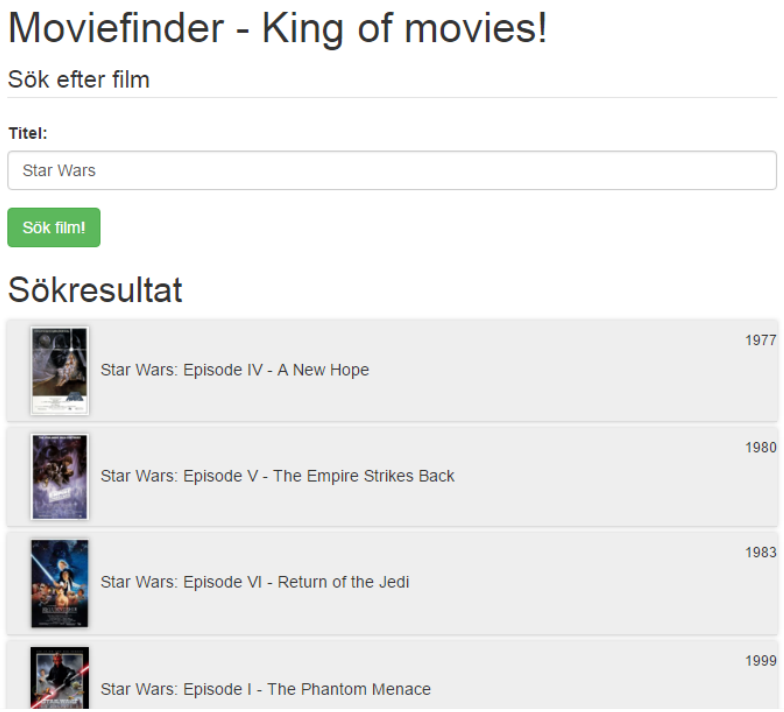
AJAX web model



moviefinder.com/index.html

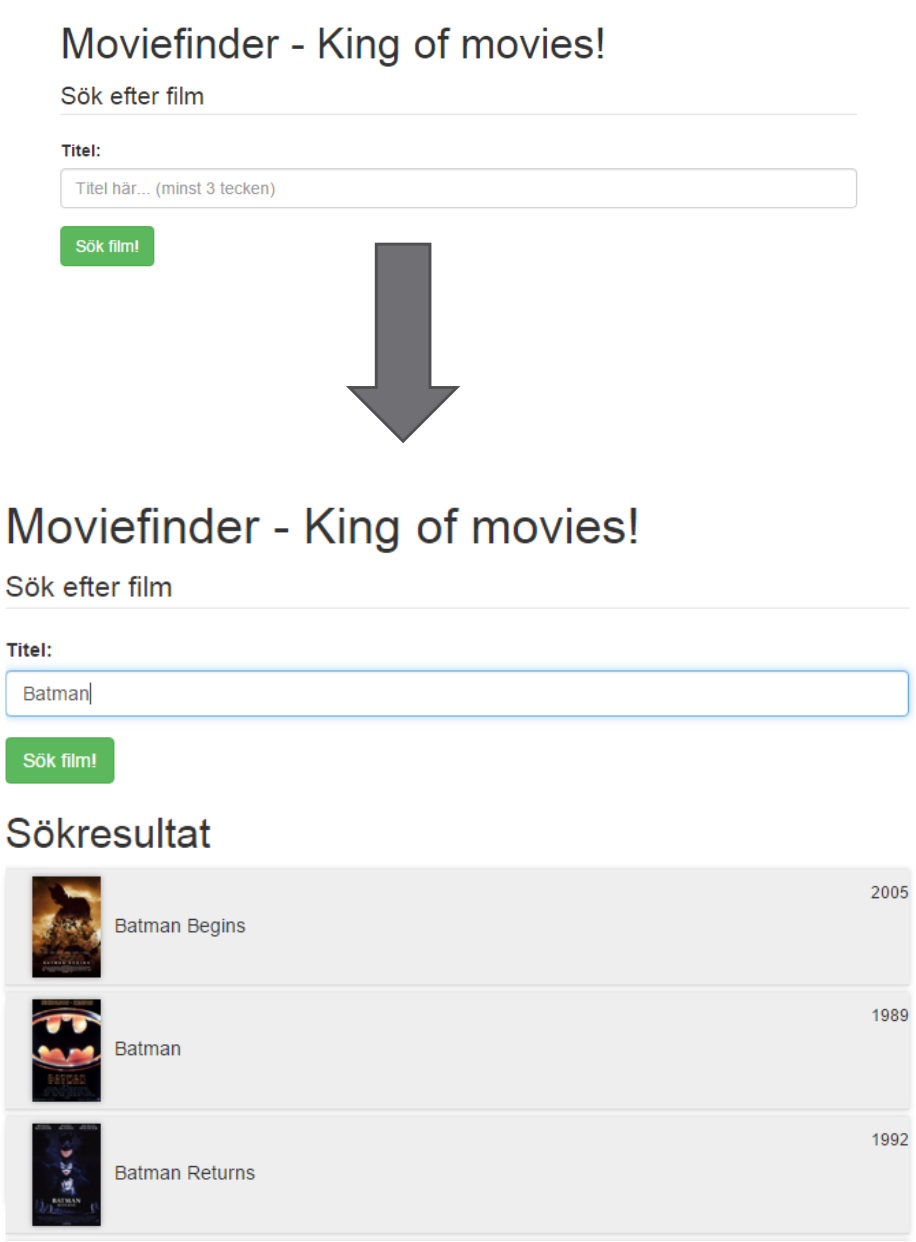
Sidomladdning

moviefinder.com/result.html



Klassisk webbsida

moviefinder.com/index.html



Interaktiv webbsida m.h.a. Ajax

Category: Shorthand Methods

These methods perform the more common types of Ajax requests in less code.

jQuery.get()

Load data from the server using a HTTP GET request.

jQuery.getJSON()

Load JSON-encoded data from the server using a GET HTTP request.

jQuery.getScript()

Load a JavaScript file from the server using a GET HTTP request, then execute it.

jQuery.post()

Load data from the server using a HTTP POST request.

.load()

Load data from the server and place the returned HTML into the matched element.

jQuerys Ajax-metoder

Category: Low-Level Interface

These methods can be used to make arbitrary Ajax requests.

jQuery.ajax()

Perform an asynchronous HTTP (Ajax) request.

jQuery.ajaxPrefilter()

Handle custom Ajax options or modify existing options before each request is sent and before they are processed by $.ajax().

jQuery.ajaxSetup()

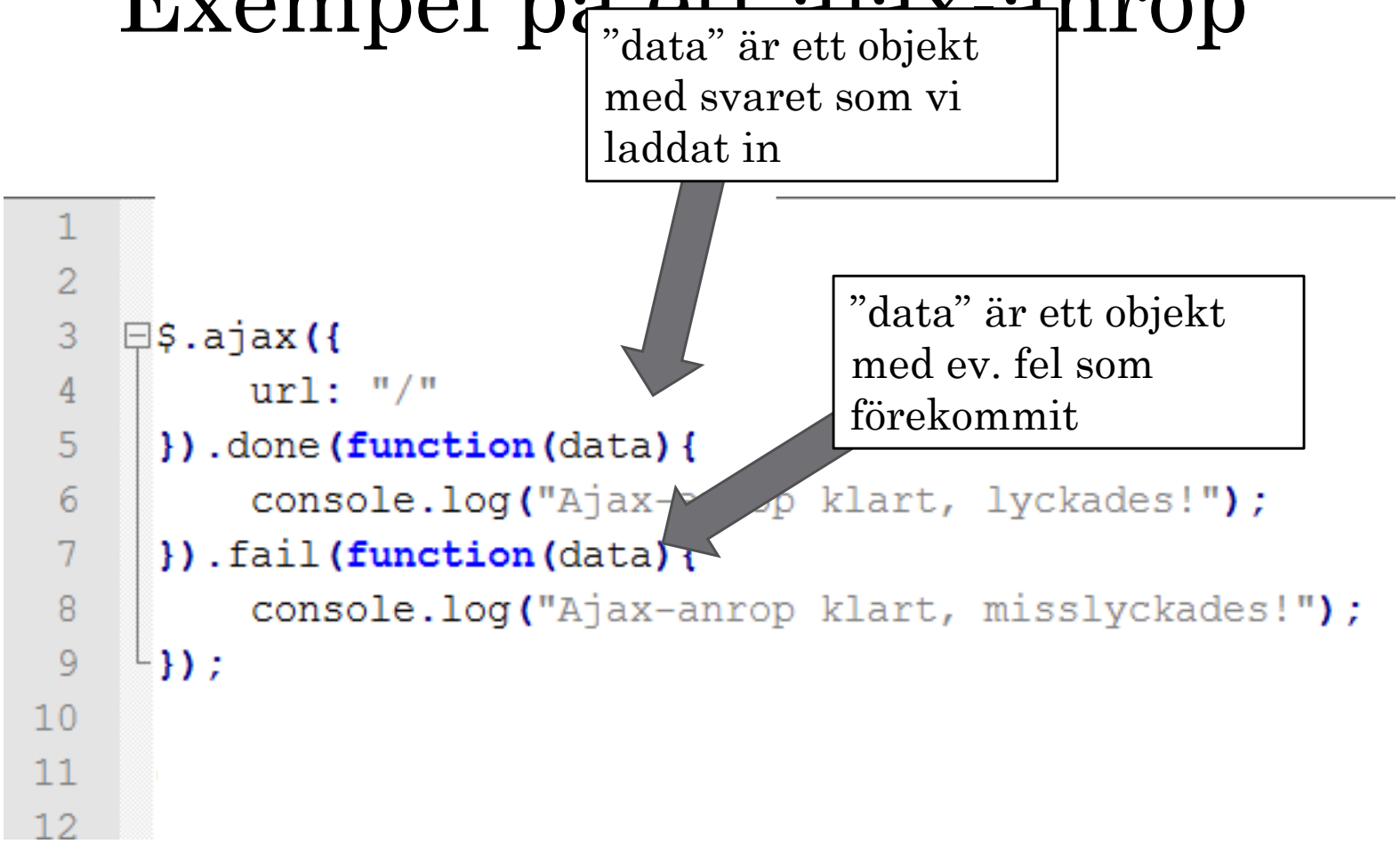
Set default values for future Ajax requests. Its use is not recommended.

jQuery.ajaxTransport()

Creates an object that handles the actual transmission of Ajax data.

jQuerys Ajax-metoder

Exempel på ett ajax-anrop



Asynkront

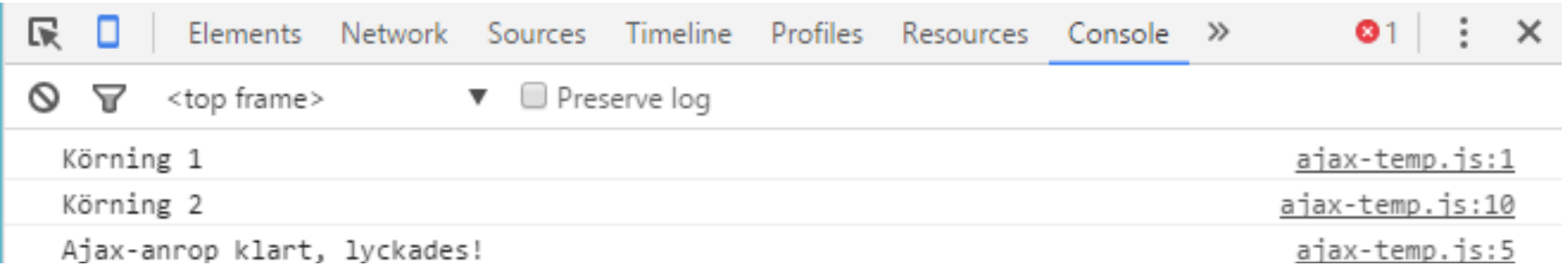
Bra att veta!

Körning av ajax-anropet

```
1  console.log("Körning 1");
2
3  $.ajax({
4      url: "/"
5  }).done(function(data){
6      console.log("Ajax-anrop klart, lyckades!");
7  }).fail(function(data){
8      console.log("Ajax-anrop klart, misslyckades!");
9  });
10
11 console.log("Körning 2");
12
```

Körning av ajax-anropet

```
1  console.log("Körning 1");
2
3  $.ajax({
4      url: "/"
5  }).done(function(data){
6      console.log("Ajax-anrop klart, lyckades!");
7  }).fail(function(data){
8      console.log("Ajax-anrop klart, misslyckades!");
9  });
10
11 console.log("Körning 2");
12
```



Varför asynkront?

- Vi vill inte behöva vänta på slöa servrar! (eller andra saker som kan hindra får sidan från att vara snabb)

Callback-funktioner

- Istället för att göra saker i den ordning som det står (synkront), vill vi istället göra saker när vårt ajax-anrop får ett svar (asynkront).

```
1   console.log("Körning 1");
2
3   $.ajax({
4       url: "/"
5   }).done(function(data){
6       console.log("Ajax-anrop klart, lyckades!");
7   }).fail(function(data){
8       console.log("Ajax-anrop klart, misslyckades!");
9   });
10
11  console.log("Körning 2");
12
```

Utan anonyma funktioner

```
console.log("Körning 1");

$.ajax({
    url: "/"
}).done(onSuccess).fail(onFail);

function onSuccess(data){
    console.log("Ajax-anrop klart, lyckades!");
}

function onFail(data){
    console.log("Ajax-anrop klart, misslyckades!");
}

console.log("Körning 2");
```

Ett standard ajax-anrop

- Med jQuery.

```
console.log("Körning 1");
$.ajax({
    url: "/", // Adressen för anropet
    type: "GET", // Vilken HTTP-metod vi ska använda
    dataType: "JSON", // Hur vi ska tolka vårt svar
    data: { // Eventuella parametrar vi vill skicka med
        arg1: "value1",
        arg2: "value2",
        arg2: "value3"
    }
}).done(function(data){
    console.log("Ajax-anrop klart, lyckades!");
}).fail(function(data){
    console.log("Ajax-anrop klart, misslyckades!");
});

console.log("Körning 2");
```

API – Ett trevligt sätt att hämta data på.

- Vi vill använda redan färdig data i våra applikationer, t.ex.
- Musik (t.ex. https://developer.spotify.com/)
- Film (t.ex. https://www.omdbapi.com/)
- Chuck Norris jokes (http://www.icndb.com/api/)
- … och massvis andra data källor.
- Vi kan även skapa egna data-källor vilket vi ska börja med! =)

Exempel: Raggningsrepliker

Raggningshjälpen!

Ge mig ett tips!

Raggningshjälpen!

Ge mig ett tips!

Tjena kexet, sitter du här och smular?

Förutsättningar

- Vi har en sida som ger oss slumpvalda raggningsrepliker i ren text
  - http://localhost:1234/date.php
- Vi vill nu varje gång vi klickar på knappen ”Ge mig ett tips”, genom JavaScript (och jQuery) ladda in denna sidas innehåll – och visa upp det

Låt oss testa!

Använda externa API

http://www.icndb.com/api/

Chuck Norris Jokes!

Give me a joke!

Chuck Norris Jokes!

Give me a joke!

Chuck Norris got his drivers license at the age of 16. Seconds.

JSON

JavaScript Object Notation

```
var jsonData = {
    "employees":[
        {"firstName":"John", "lastName":"Doe"},
        {"firstName":"Anna",    "lastName":"Smith"},
        {"firstName":"Peter", "lastName":"Jones"}
    ]
};

console.log(jsonData);
console.log(jsonData.employees);
console.log(jsonData.employees[0]);
console.log(jsonData.employees[0].firstName);
console.log(jsonData.employees[0].lastName);
```

```
▶ Object {employees: Array[3]}
▶ [Object, Object, Object]
Object {firstName: "John", lastName: "Doe"}
John
Doe
```

http://api.icndb.com/jokes/random

```
var data = {
    "type":"success",
    "value":{
        "id":243,
        "joke":"Contrary to popular belief, the Titanic didn't
        hit an iceberg. The ship was off course and ran into
        Chuck Norris while he was doing the backstroke across
        the Atlantic.",
        "categories":[]
    }
}
```

http://www.icndb.com

Dokumentation för API:t