INTRODUCTION TO REST

MATERIAL BY JANNE PARKKILA

2.03.2015 @ LAPPEENRANTA UNIVERSITY OF TECHNOLOGY

WHAT IS REST?

- Representational State Transfer

= Stateless, Client-server, cacheable communications protocol, over HTTP

WHAT IS REST?

- RESTful application use HTTP requests to handle the usual CRUD-operations
- CRUD =
  - **C**reate
  - **R**ead
  - **U**pdate
  - **D**elete

REST: CRUD?

There is a HTTP method for each operation, stated in the HTTP definition

Create = HTTP POST

Read = HTTP GET

Update = HTTP PUT

Delete = HTTP DELETE

HOW REST?

- REST uses URL routes to communicate with the server

- REST uses often JSONs to exchange information

HOW REST?

- **REST uses URL routes to communicate with the server**
- REST uses often JSONs to exchange information

HOW REST?

- **REST uses URL routes to communicate with the server**
- Example: Get all users

http://www.example.com/users

- Example: Get information of a single user

http://www.example.com/users/Japskua

HOW REST?

- REST uses URL routes to communicate with the server

- **REST uses often JSONs to exchange information**

WHAT JSON?

- JavaScript Object Notation
- Simple & lightweight
- Example: User Japskua:

```
{
    name: Janne Parkkila
    nick: Japskua
    email : janne.parkkila@example.com
}
```

WHY JSON?

- Ubiquity
- Simplicity
- Readability
- Flexibility

WHY REST? (DEBATABLE :-D)

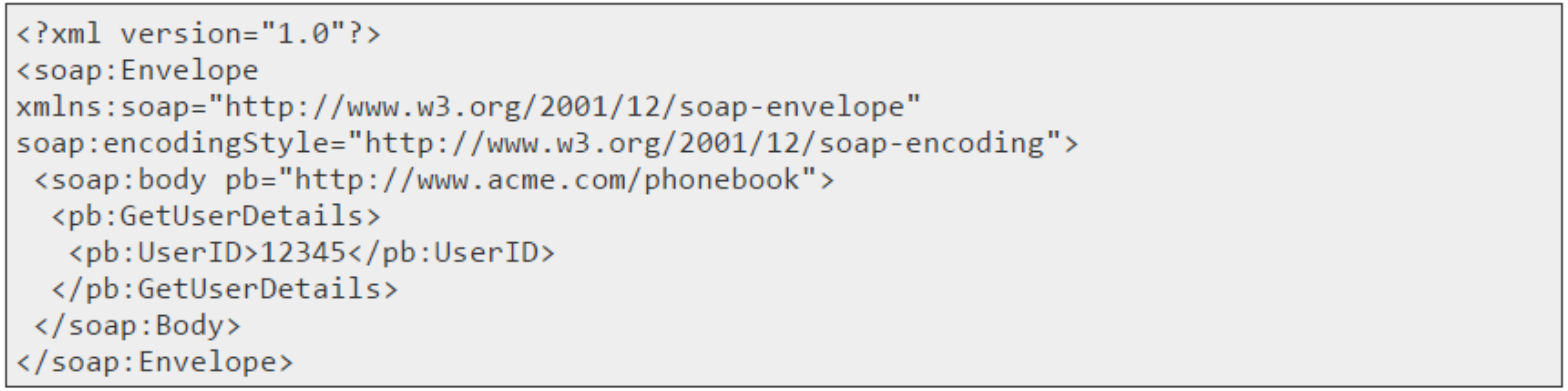
- Scalability
- Generality
- Independence
- Latency (Caching)
- Security
- Encapsulation

WHY REST?

- Platform-independent (only requirement is the ability to use HTTP connection)
- Language independent (client & server don’t have to use the same implementation)
- Standards based (like I said, HTTP!)

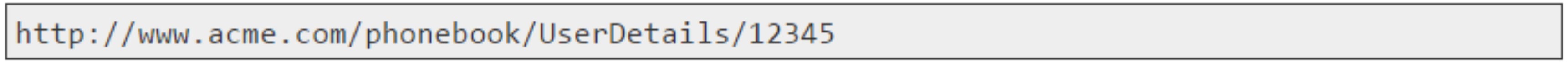
REST VS SOAP

SOAP



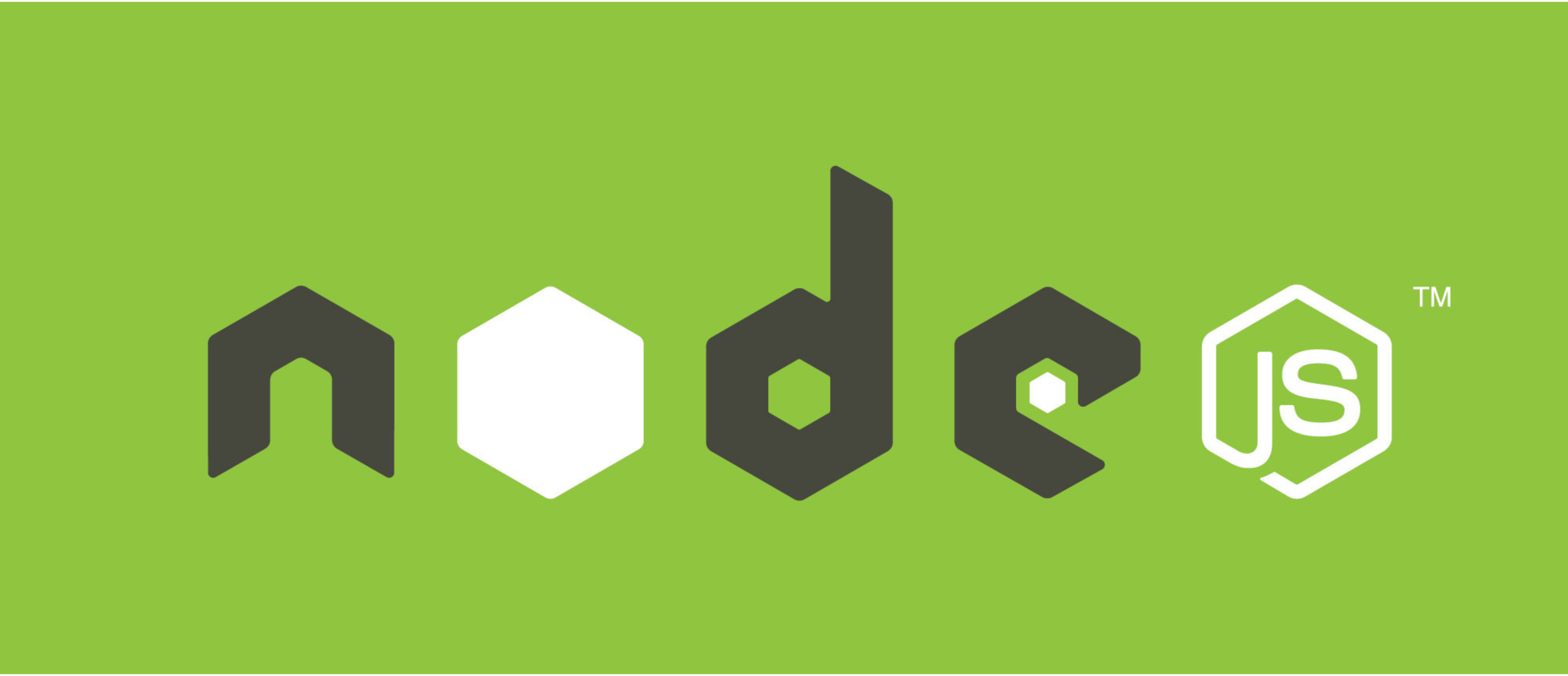
```
<?xml version="1.0"?>
<soap:Envelope
xmlns:soap="http://www.w3.org/2001/12/soap-envelope"
soap:encodingStyle="http://www.w3.org/2001/12/soap-encoding">
 <soap:body pb="http://www.acme.com/phonebook">
  <pb:GetUserDetails>
   <pb:UserID>12345</pb:UserID>
  </pb:GetUserDetails>
 </soap:Body>
</soap:Envelope>
```

REST



```
http://www.acme.com/phonebook/UserDetails/12345
```

Example from: http://rest.elkstein.org/


node
JS
TM

NODE.JS

Server-side JavaScript

“**Node.js® is a platform** built on **Chrome's JavaScript runtime** for easily building fast, scalable **network applications.**

Node.js uses an **event-driven, non-blocking** I/O model that makes it lightweight and efficient, perfect for data-intensive real-time applications that run across distributed devices.”

NODE.JS DEVELOPMENT STACK

BACKBONE.JS

ANGULARJS
by Google

browserify

WHO IS USING THIS STUFF?

- Yahoo!
- LinkedIn
- Ebay
- Dow Jones
- GoDaddy
- Heroku
- DataHero
- WallStreet Journal
- General Electric
- Klout
- Medium
- Peek Inc
- Shutterstock
- Storify
- Trello
- Uber
- Yammer
- Zendesk

WHEN TO USE?

- Streaming, real-time services, web-chat applications, static file servers, etc.
- High level concurrency without worry about CPU-cycles
- Basically always, when developing Web-apps

WHEN NOT?

- Complex processing
- Long running processess
- Not really supporting multi-core of the processor. Runs single threaded.

THE BEST

- One language to rule them all (.js)
- Fast in handling requests
- Awesome package manager, HUGE community
- REST-services
- Want to be part of the API ecosystem? This is the way to go!

SOME MORE RESOURCES ON REST

- http://www.slideshare.net/apigee/restful-api-design-second-edition
- http://rest.elkstein.org/
- http://www.infoq.com/articles/designing-restful-http-apps-roth
- http://www.drdobbs.com/web-development/restful-web-services-a-tutorial/240169069
- http://www.slideshare.net/FDConf/writing-restful-web-services-using-nodejs