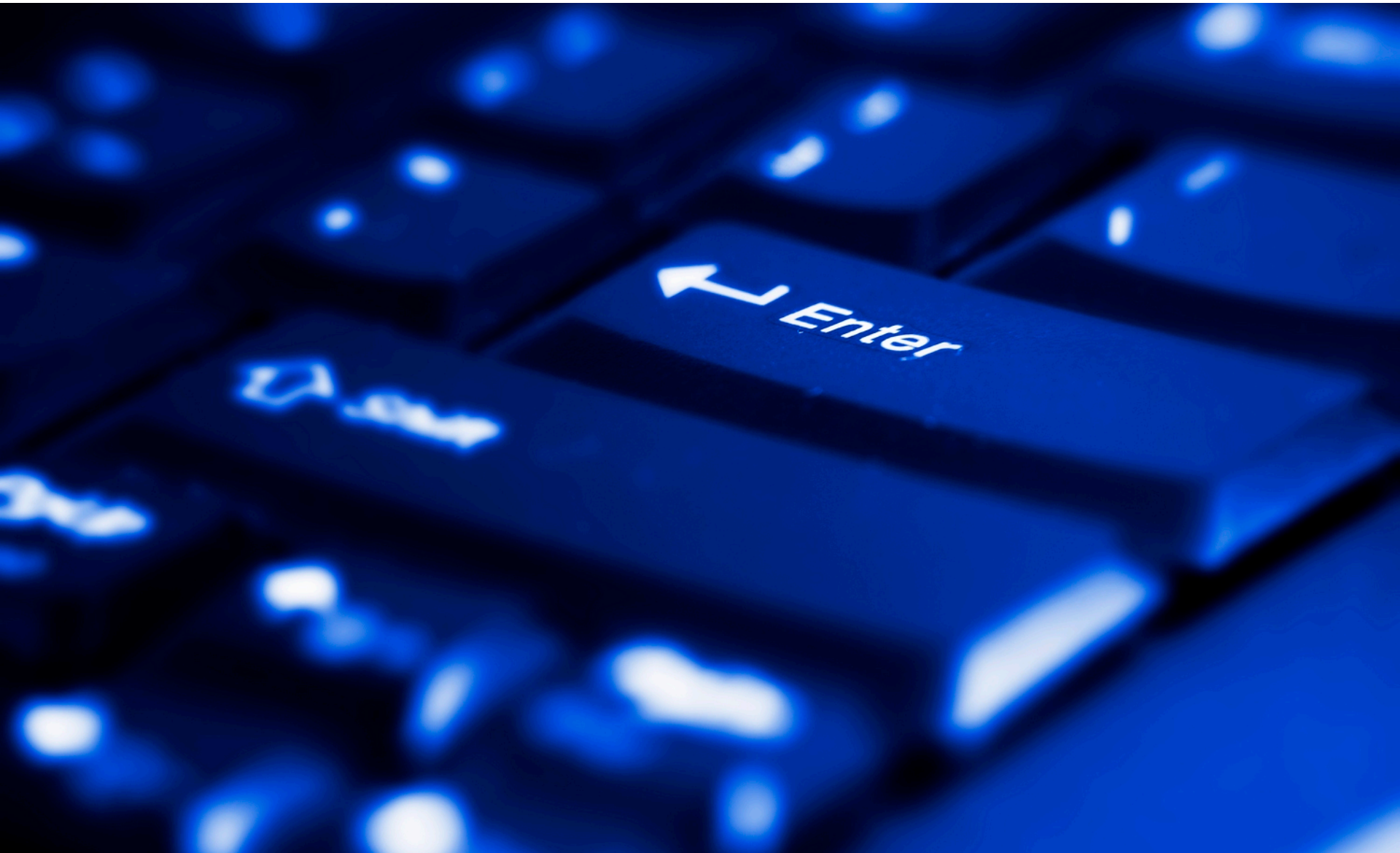
bns it

# REST

## Modern Application Architectures




# REST

# REST

- REST is an architectural constraint based on HTTP 1.1 by Roy Fielding
- Embraces HTTP
- It’s a style, no a standard
- REST embraces HTTP (verb, headers, request params)

# GET

- Requests retrieve information
- Can have side-effects (but it’s not expected)
- Can be conditional or partial
  - If-Modified-Since, Range

```
GET /users/21
```

# DELETE

- Requests a resource to be removed, though the deletion doesn't have to be immediate

```
DELETE /users/21
```

# POST

# A request with the enclosed entity

# Used for create (or update)

```
POST /users
{ „firstName”: „Mariusz”}
```

# PUT

- Request the entity to be stored at a URI

- Used for update (or create)

```
PUT/users/21
{ „firstName”: „Mariusz”}
```

# Status Codes

- Result of the server's attempt to satisfy request
- Categories:
  - 1xx: informational
  - 2xx: success
  - 3xx: redirection
  - 4xx: client error
  - 5xx: server error

# 2xx Status Codes

- **200 OK** – Everythig is ok
- **201 Created** – Returns a location header for a new resource
- **202 Accepted** – Server acceppted the request, but it is not yet complete

# 4xx Status Codes

- **400 Bad Request** – Malformed Syntax
- **401 Unauthorized** – Authentication required
- **403 Forbidden** – Request refused
- **404 Not Found** – Server can’t find a resource for URI
- **406 Incompatible** – Incompatible Accept headers
- **409 Conflict** – Resource conflicts with client request (specific rules are not satisfied eg. attachement size)

# Content negotiation

- Spring MVC supports multiple types of content negotiation through `ContentNegotiationStrategy`
  - `Accept header, URL extension, request parameters`

# HTTP Clients

- Google Chrome **Advanced REST client** plugin
- Firefox **Poster** plugin
- Unix based **curl**

# REST Maturity Model

- Richardson Maturity Model grades API according to the REST constraints with levels of increasing compliance

# Richardson Maturity Model

## Level 0: Swamp of POX

- Uses HTTP as a tunnel through one URI (eg. SOAP, XML-RPC)
- Usually features on HTTP verb (POST)

# Richardson Maturity Model

## Level 1: Resources

- Multiple URIs to distinguish related nouns
- eg. **/articles/1**, **/articles/2** or **/articles**

# Richardson Maturity Model

## Level 2: HTTP verbs

- Leverage transport native properties (eg. **GET, PUT, DELETE, POST**)
- Uses idiomatic HTTP controls like status codes and headers


# Hypermedia

# Richardson Maturity Model

## Level 3: Hypermedia Controls (aka HATEOAS)

- No a prior knowledge of service required
- Navigation options are provided by a service and hypermedia controls

# Links

- Provide possible navigations from a given resource
- Links are dynamic, based on resource state

```
{ href: „http://localhost:8080/users/232/customers”,
rel: „customers” }
```