



Stemmarest

Dokumentation des PSE2 Projekt

Jakob Schaerer, Severin Zumbrunn, Ido Gershoni, Joel Niklaus, Ramona Imhof



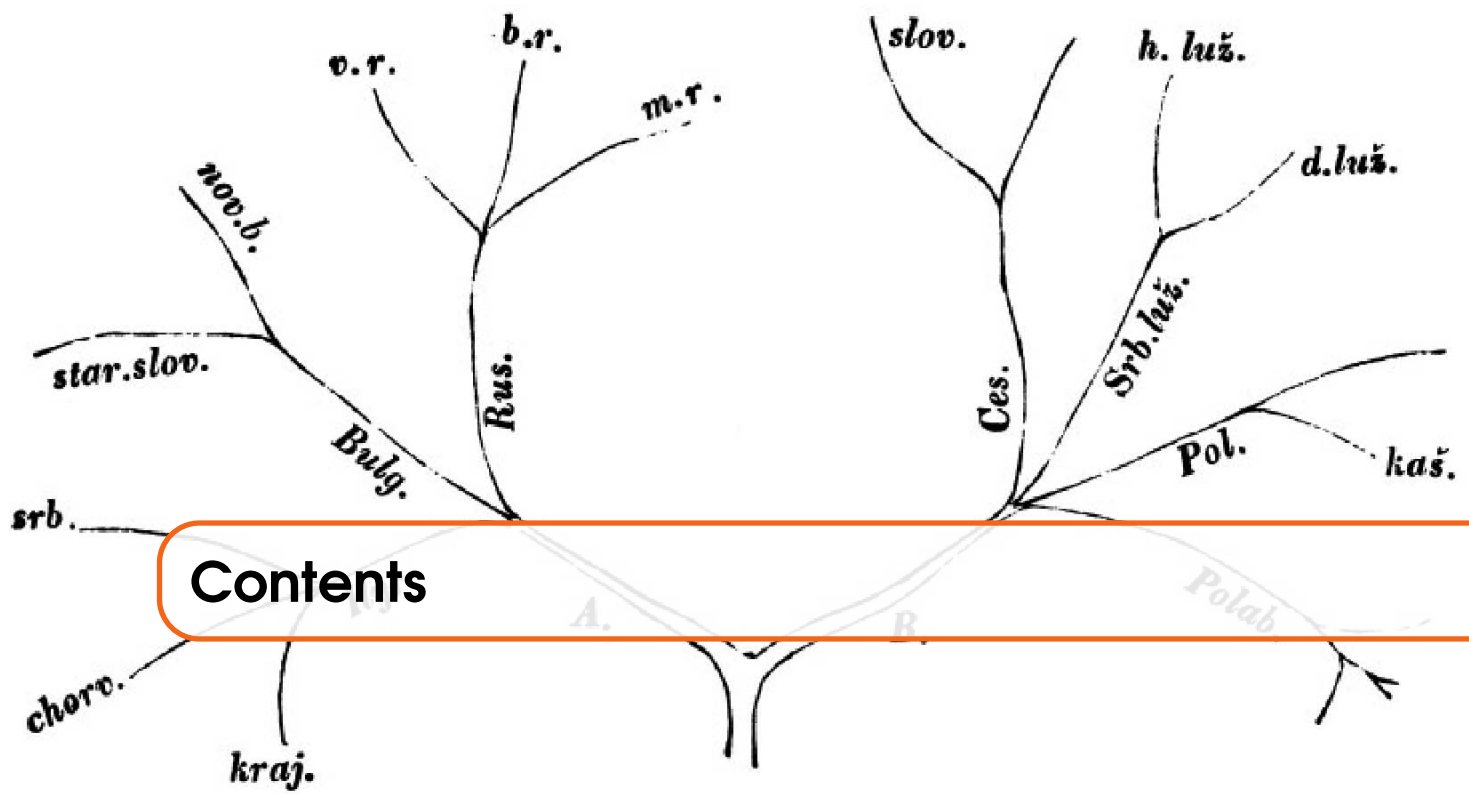
Copyright © 2015 Team PSE2

PUBLISHED BY ...

....ORG

Licensed under the Creative Commons Attribution-NonCommercial 3.0 Unported License (the “License”). You may not use this file except in compliance with the License. You may obtain a copy of the License at <http://creativecommons.org/licenses/by-nc/3.0>. Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an “AS IS” BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License.

First printing, March 2013

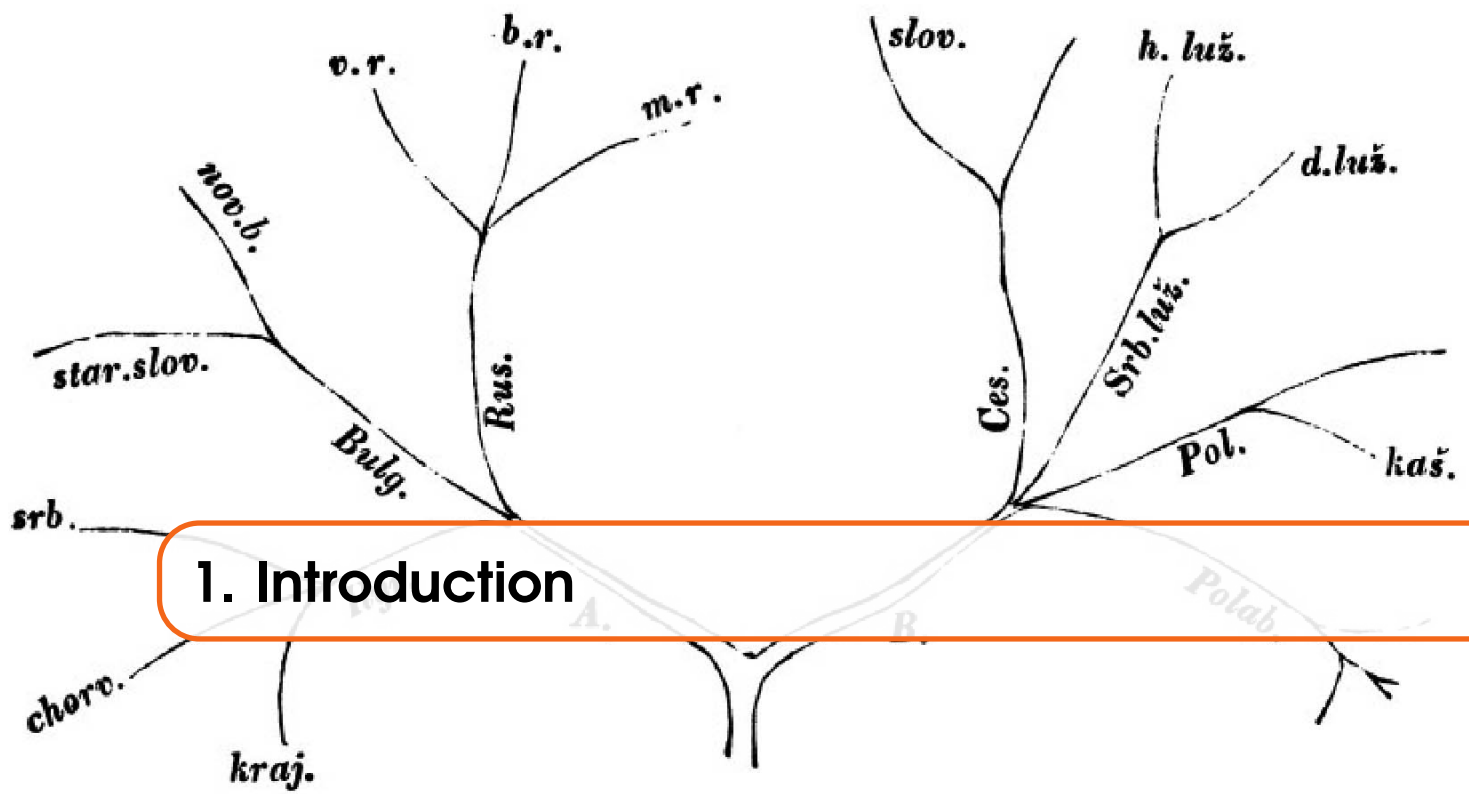


| | | |
|-----------|-------------------------------|-----------|
| I | Project | |
| 1 | Introduction | 6 |
| 2 | Database (neo4j) | 7 |
| 2.1 | Structure | 7 |
| 3 | Jersey | 9 |
| II | RESTful API | |
| 4 | Documentation | 11 |
| 4.1 | /user | 11 |
| 4.2 | /user/create | 11 |
| 4.3 | /user/{id} | 12 |
| 4.4 | /user/traditions/{userId} | 12 |
| 4.5 | /textinfo/{textId} | 12 |
| 4.6 | /tradition/witness/{tradId} | 13 |
| 4.7 | /tradition/new | 13 |
| 4.8 | /tradition/get/{tradId} | 14 |
| | Bibliography | 15 |
| | Books | 15 |



Project

| | | |
|----------|-------------------------------|----------|
| 1 | Introduction | 6 |
| 2 | Database (neo4j) | 7 |
| 2.1 | Structure | |
| 3 | Jersey | 9 |



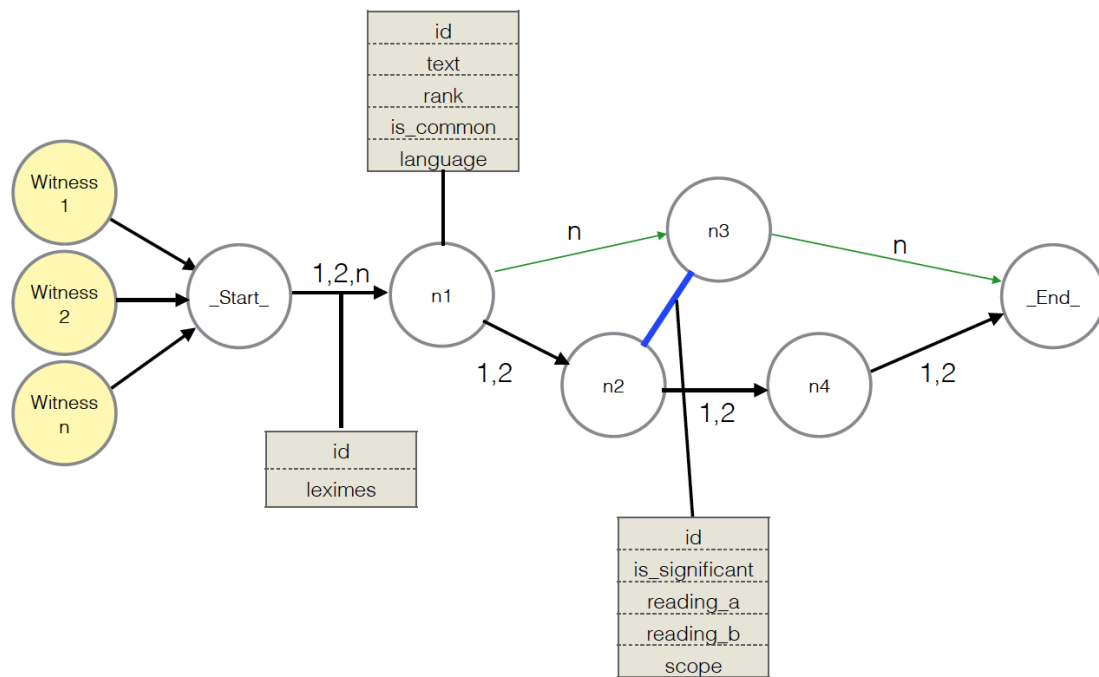
1. Introduction

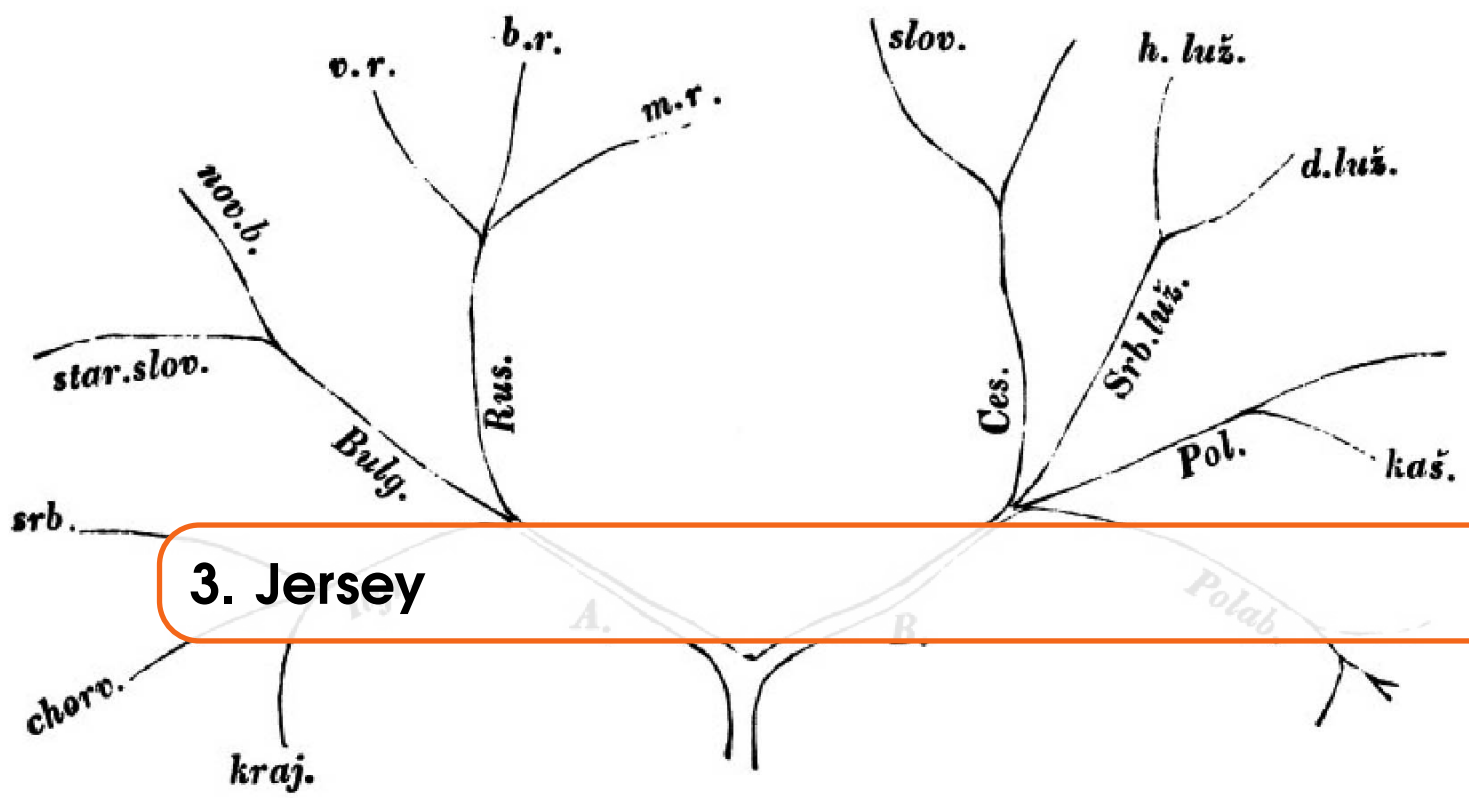
Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua. At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit amet. Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua. At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit amet.



```

graph TD
    Root((Root)) --> User1((User 1))
    Root --> UserN((User n))
    User1 --> Tradition1((Tradition 1))
    User1 --> TraditionN((Tradition n))
    Tradition1 --> Witness1((Witness 1))
    Tradition1 --> Witness2((Witness 2))
    Tradition1 --> WitnessN((Witness n))
    id[id] --- User1
    attributes["name  
-----  
language  
-----  
public"] --- Tradition1
    lexemes["lexemes =  
„name“"] --- Witness1
  
```



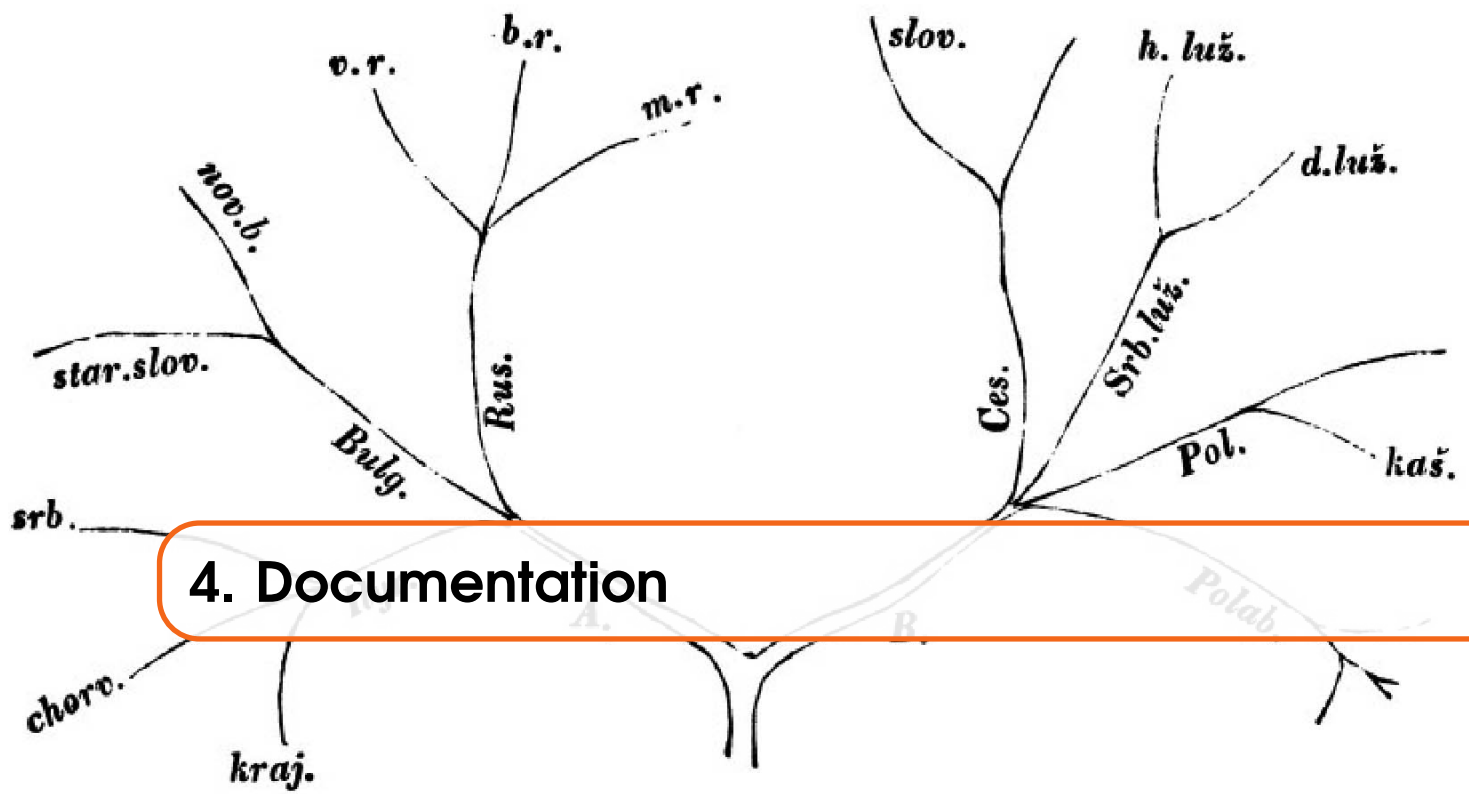


Lorem Ipsum



RESTful API

| | | |
|----------|-----------------------------|-----------|
| 4 | Documentation | 11 |
| 4.1 | /user | |
| 4.2 | /user/create | |
| 4.3 | /user/{id} | |
| 4.4 | /user/traditions/{userId} | |
| 4.5 | /textinfo/{textId} | |
| 4.6 | /tradition/witness/{tradId} | |
| 4.7 | /tradition/new | |
| 4.8 | /tradition/get/{tradId} | |
| | Bibliography | 15 |
| | Books | |
| | Articles | |



4.1 /user

GET /user

Summary

Returns a welcome message

Parameter

Return — SUCCESS. text/plain
"User!"

4.2 /user/create

POST /user/create

Summary

Creates a user.

Parameter application/json

{ "userId":<userId>, "isPublic":<isAdmin> }

Return — CREATED. application/json

{ "userId":<userId>, "isPublic":<isAdmin> }

Return — CONFLICT. application/json

Error: A user with this id already exists

4.3 /user/{id}

```
GET /user/{id}
```

Summary

Returns the user as JSON Object

Parameter URL

Id: the user id

Return — OK. application/json

```
{ 'userId': <userId>, 'isAdmin': <isAdmin> }
```

The information about the user

Return — NOT_FOUND. application/json

The information about the user

4.4 /user/traditions/{userId}

```
GET /user/traditions/{userId}
```

Summary

List all Traditions of a user

Parameter URL

userId: the id of the user

Return — OK. application/json

```
{ "traditions": [ { "name": <traditionName> } ] }
```

Return — NOT_FOUND. application/json

Error: A user with this id does not exist!

4.5 /textinfo/{textId}

```
POST /textinfo/{textId}
```

Summary

Update the textInfo of a tradition.

Parameter URL

textId: the id of the tradition

Parameter application/json

```
{ 'name': <new_name>, 'language': <new_language>, 'isPublic': <is_public>, 'ownerId':  
<new_ownerId> }
```

Return — SUCCESS. application/json

```
{ 'name': <new_name>, 'language': <new_language>, 'isPublic': <is_public>, 'ownerId':  
<new_ownerId> }
```

The new information of the tradition.

Return — CONFLICT. application/json

"Error: A user with this id does not exist"

If the user does not exist.

Return — NOT_FOUND. application/json

If the tradition was not found.

4.6 /tradition/witness/{tradId}

```
GET /tradition/witness/{tradId}
```

Summary

List all Witness of a tradition

Parameter URL

tradId: the id of the tradition

Return — OK. application/json**Return — NOT_FOUND. application/json**

Error: A tradition with this id does not exist!

4.7 /tradition/new

```
POST /tradition/new
```

Summary

Create a new tradition.

Parameter text/plain

name: The name of the tradition

Parameter multipart/form-data

language: The language of the tradition

public: 0 if the tradition is not public 1 if the tradition is public

name: The name of the tradition

file: multipart file input stream

Return — **CONFLICT.** **application/json**

"Error: No user with this id exists"

Return — **INTERNAL_SERVER_ERROR.** **application/json**

"Error: Tradition could not be imported!" *If the server was not able to parse the input file*

Return — **OK.** **application/json**

"Tradition imported successfully"

4.8 /tradition/get/{tradId}

GET /tradition/get/{tradId}

Summary

Get a graphml of a tradition

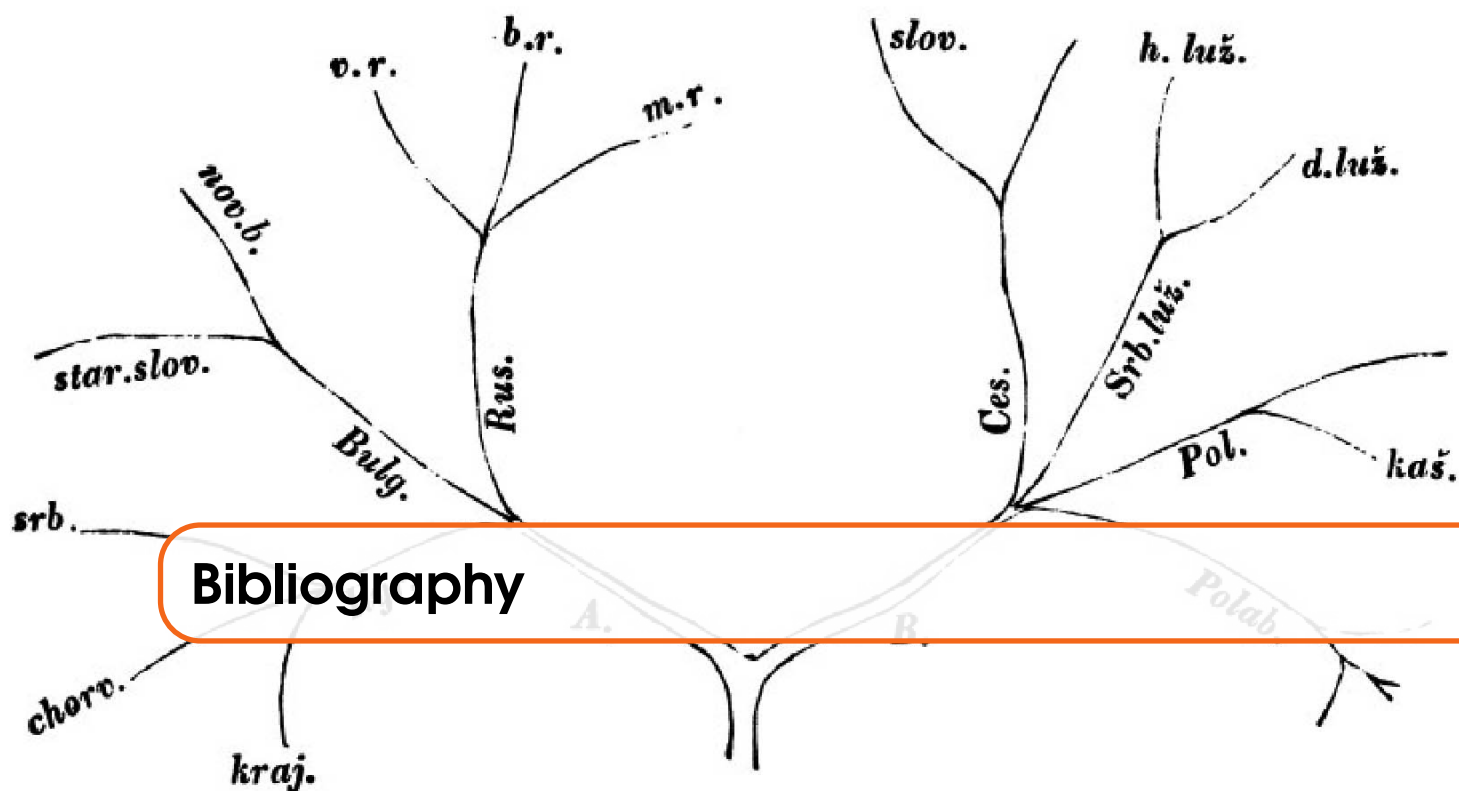
Parameter **URL**

tradId: the id of the tradition

Return — **OK.** **application/xml**

Return — **NOT_FOUND.** **application/json**

Error: A tradition with this id does not exist!



Books
Articles

