Getting Started with REST & Symfony

SensioLabs

Requirements

Requirements

- Basic experience with Symfony & Twig
- Basic knowledge of the MVC principle
- Basic knowledge of HTTP

That is all!

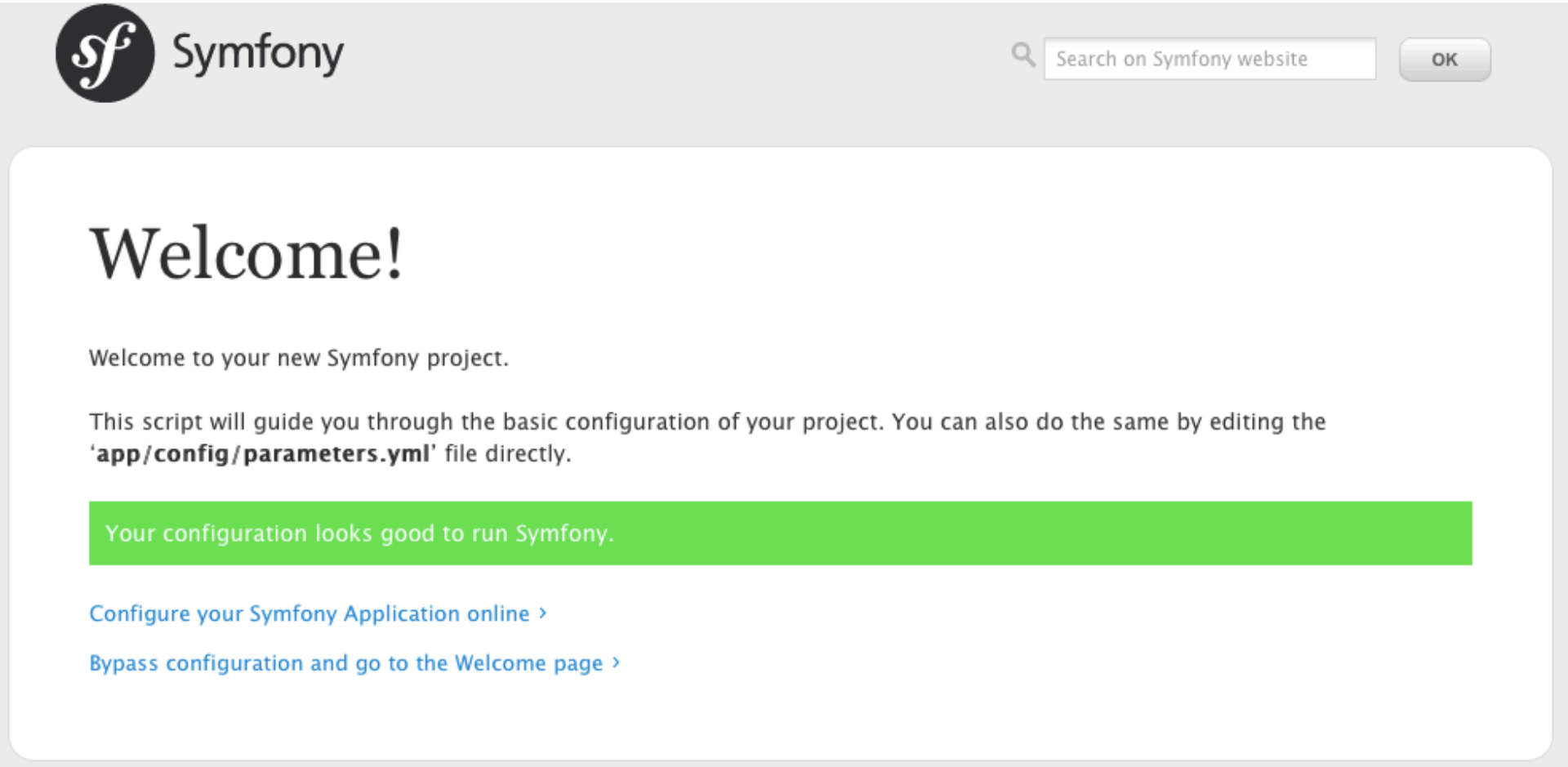
Project installation

Unzip and check application

Look at the README file.

Environment check > Web

http://127.0.0.1:8000/config.php



Configure the database

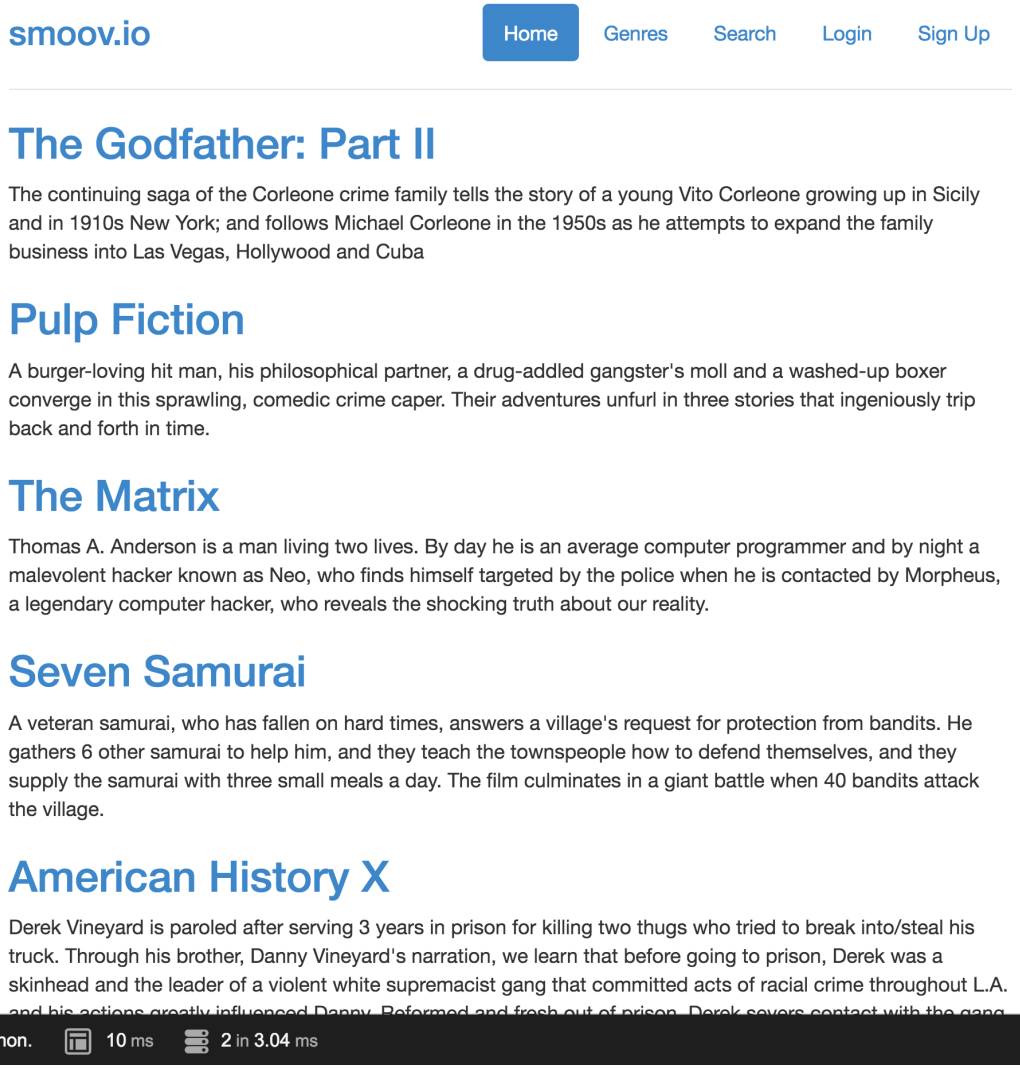
```
$ cat app/config/parameters.yml
parameters:
    database_driver: pdo_mysql
    database_host: 127.0.0.1
    database_port: null
    database_name: smoovio
    database_user: root
    database_password: ~

$ bin/console doctrine:database:create
$ bin/console doctrine:schema:create
$ mysql -uroot -p smoovio < data.sql
$ bin/console smoovio:playlist:create_featured
```

Project’s context

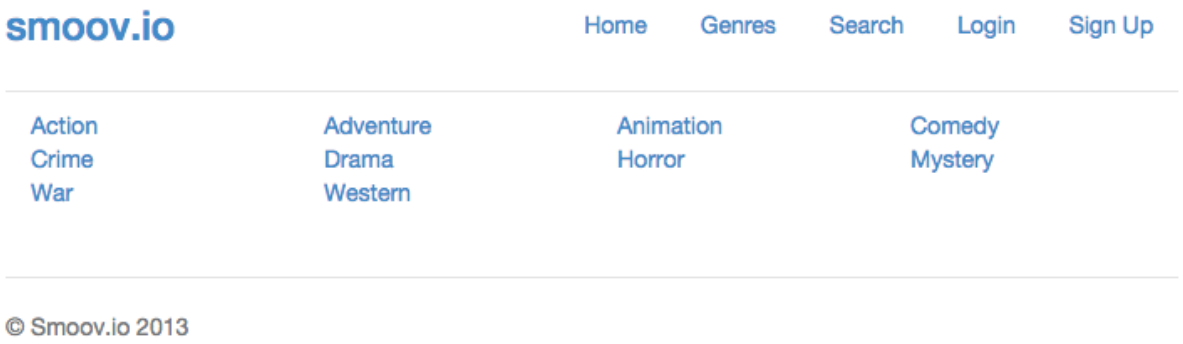
Smoov.io

- A clone of IMDB.com which is THE new trendy **movie database** web app.
- Victim of its own success, developers created a **quick & dirty** web API to share its content with third party apps.
- After lots of API consumers' complaints, it's been decided to enhance it with the help of API designer experts (**that's you!**).

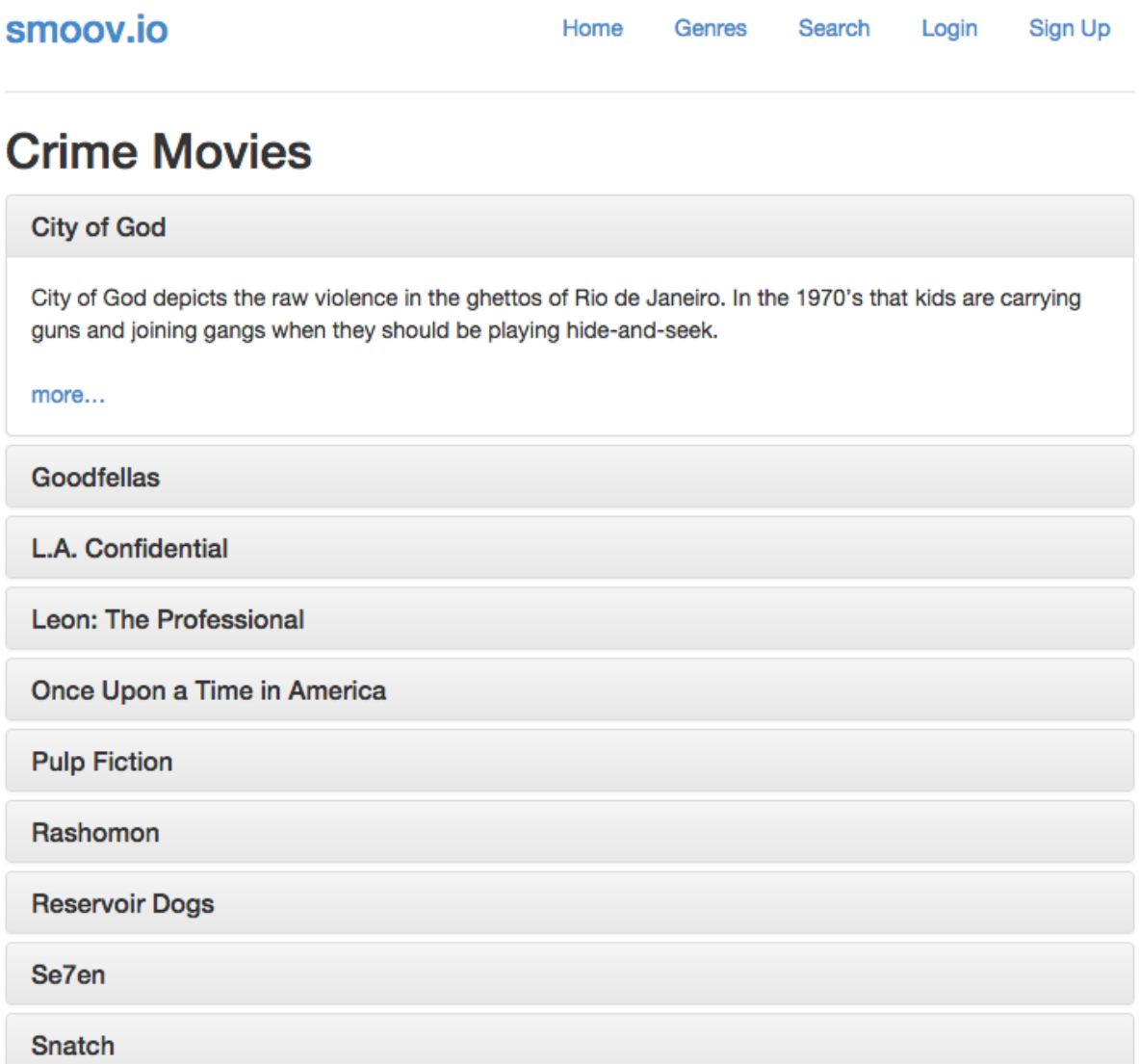
Smoov.io's Tour > GUI > home



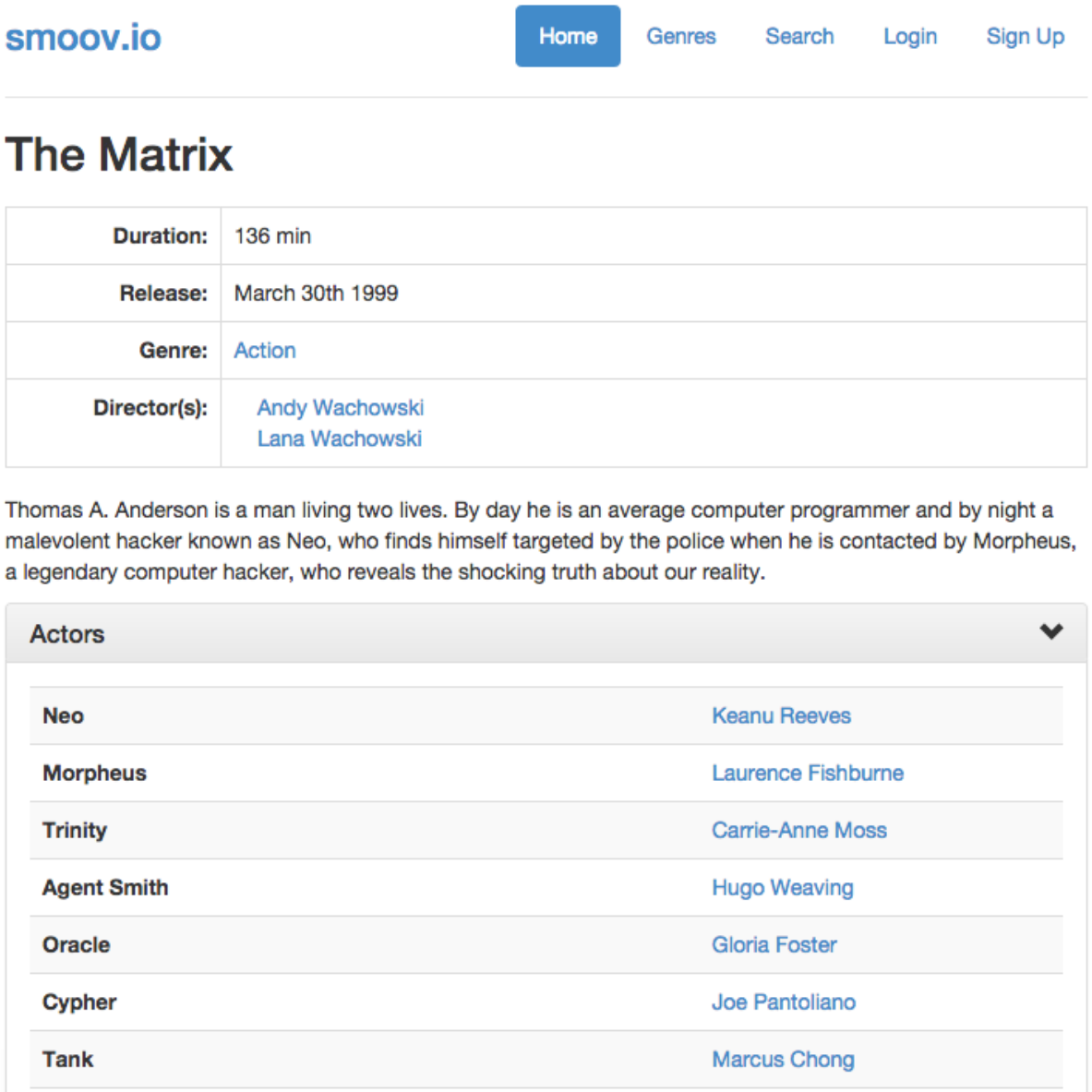
Smoov.io's Tour > GUI > genres



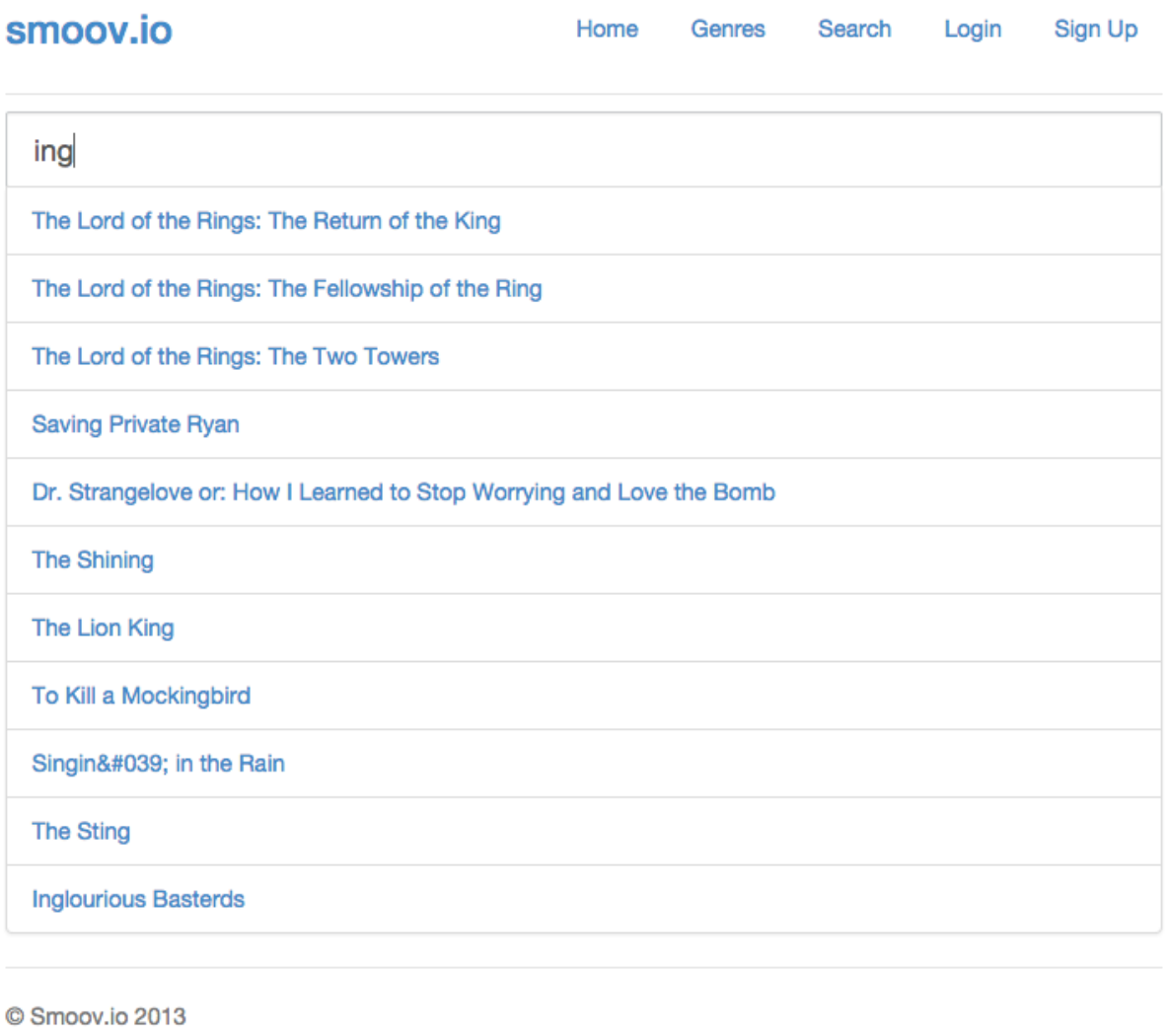
Smoov.io's Tour > GUI > genre



Smoov.io's Tour > GUI > Movie's detail



Smoov.io's Tour > GUI > Search



Your mission:

« Should you choose to accept it. . . »

Mission: Impossible quote

What is REST?

REpresentational State Transfer

It's **not a protocol** but a **style of architecture**.

REST Constraints

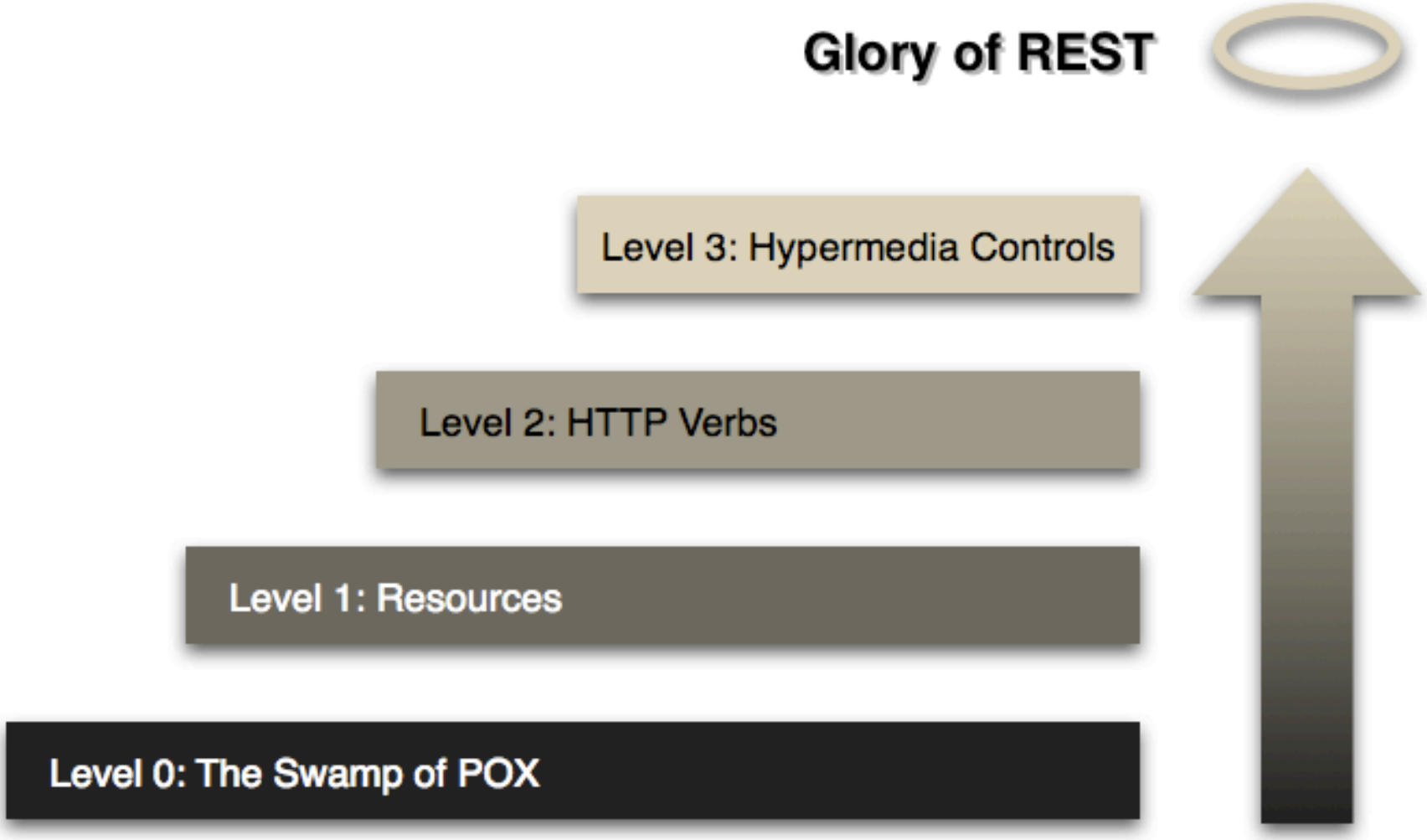
- Client-Server
- Stateless
- Cacheable
- Layered System
- **Uniform Interface**

Uniform Interface

- One unique URL per resource
- Operations on resources are performed with HTTP methods (verbs).
- Operations are not part of the URLs
- An hypermedia format is used to represent the data
- Links between relations are used to browse the API

How to grade a web API?

Richardson maturity model



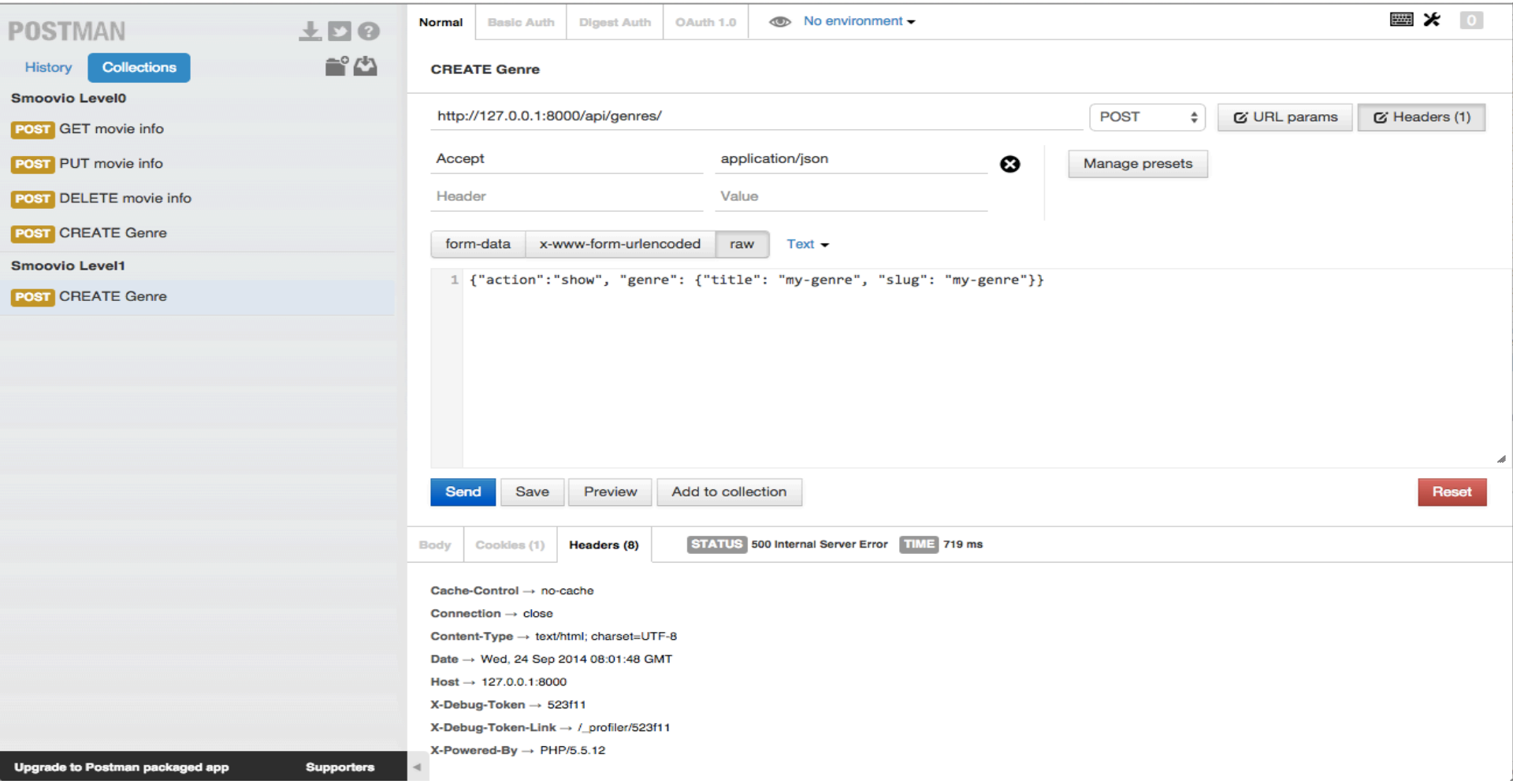
Source: http://martinfowler.com/articles/richardsonMaturityModel.html

API developer’s toolbox

CLI: Curl

```
$ curl 127.0.0.1 -H "Accept: application/
json" -X PUT -u myuser:pass --data
'{"message":"hello"}'

-H      Add an HTTP header to the request
-u      HTTP Authentication
-X      Specify HTTP verb (GET/POST/PUT/...)
-v      Display sent request and received body
-I      Display response headers only
--data Add data to request body
```

GUI: POSTMan chrome extension



https://chrome.google.com/webstore/detail/postman-rest-client/fdmmgilgnpjigdojojpjoooidkmcomcm

REST Compliance of Smoov.io

SensioLabs

Level 0

HTTP as a transport protocol

Level 0 in theory

- HTTP is used just like a **transport protocol**.
- Requests and responses are tunneled without using the protocol to indicate **application state**.
- It uses only one entry point (URI) and one method (HTTP POST usually).
- Examples: SOAP and XML-RPC.

Level 1
Resources

Level 1 in theory

- API can distinguish different **resources**.
- **Multiple URIs** are used.
- Each URI is the entry point to a specific resource (instead of using **/articles**, you actually distinguish between **/article/1** and **/article/2**).
- One single method is used (HTTP POST usually).

Practice > Smoov.io

What are the existing **resources** on Smoovio's API?

Film and **Genre**

What **routes** should be created?

`POST` `/api/movies/{slug}`

`POST` `/api/genres/`

Level 2
HTTP verbs

Level 2 in theory

- API uses the right **HTTP verbs** & status codes.
- Besides **POST** method, **GET** is used to request resources and **DELETE** to remove them.
- Responses now include the appropriate status code (e.g. don't use 200 (OK) code when something went wrong).

Practice > Safe HTTP Methods

In particular, the convention has been established that the **GET and HEAD methods SHOULD NOT have the significance of taking an action other than retrieval**. **These methods ought to be considered "safe"**. This allows user agents to represent other methods, such as POST, PUT and DELETE, in a special way, so that the user is made aware of the fact that a possibly unsafe action is being requested.

w3c-1 w3.org protocols rfc2616-sec9

Practice > Idempotent Methods

Methods can also have the property of "idempotence" in that the side-effects of N > 0 identical requests is the same as for a single request. The GET, HEAD, PUT and DELETE methods share this property.

Also, the OPTIONS and TRACE methods SHOULD NOT have side effects, and so are inherently idempotent.

w3c-1 w3.org protocols rfc2616-sec9

Practice > Smoov.io

What **routes** should be created?

GET	`/api/movies/{slug}`
DELETE	`/api/movies/{slug}`
PUT	`/api/movies/{slug}`
PATCH	`/api/movies/{slug}`
POST	`/api/genres/`

Practice > Smoov.io

What kind of **routes** should be created?

GET	`/api/movies/{slug} => 200`
DELETE	`/api/movies/{slug} => ~~200~~ 204`
PUT	`/api/movies/{slug} => 200 204`
PATCH	`/api/movies/{slug} => 200 204`
POST	`/api/genres/ => ~~200~~ 201`

Level 3
Hypermedia controls

Level 3 in theory

- HATEOAS describes for a given **resource** what **actions** can be done from it.
- HATEOAS representation has no real standard which support multiple formats
  - **HAL** specification: cross format (XML/json) but it is a personal initiative.
  - **JSON-LD**: specification supported by W3C but it only supports JSON.
  - **Siren**: https://github.com/kevinswiber/siren

Level 3 in practice

Granted actions on a bank account with a **balance > 0**:

```
GET /account/12345 HTTP/1.1

HTTP/1.1 200 OK
<?xml version="1.0"?>
<account>
    <account_number>12345</account_number>
    <balance currency="usd">100.00</balance>
    <link rel="deposit" href="/account/12345/deposit" />
    <link rel="withdraw" href="/account/12345/withdraw" />
    <link rel="transfer" href="/account/12345/transfer" />
    <link rel="close" href="/account/12345/close" />
</account>
```

Level 3 in practice

Granted actions on a bank account with a **balance < 0**:

```
GET /account/12345 HTTP/1.1

HTTP/1.1 200 OK
<?xml version="1.0"?>
<account>
    <account_number>12345</account_number>
    <balance currency="usd">-25.00</balance>
    <link rel="deposit" href="/account/12345/deposit" />
</account>
```

Advantages of using HATEOAS

- **Compatibility**: API can change target's links without breaking consumer's apps.
- Self-documented API.
- Easy to browse.

Practicing REST on Smoov.io

SensioLabs

User Story #1

As a consumer, I want to get movies genres as JSON.

Smoovio > Genres

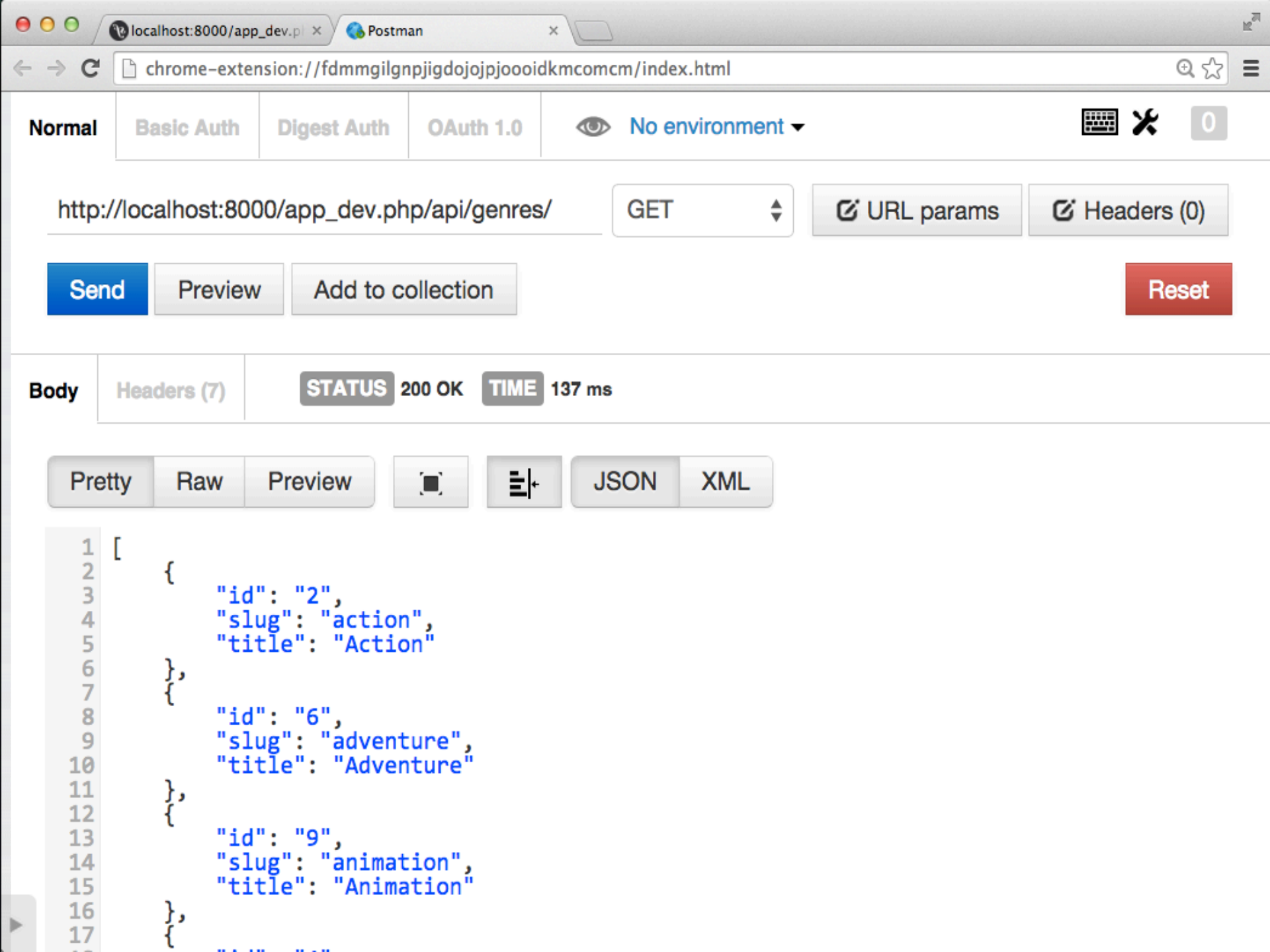
- Create one GenreController,
- Implement JSON responses with raw Twig templates,
- Use Genre repository to fetch data,

Smoovio > List of Genres

```php
namespace Smoovio\Bundle\ApiBundle\Controller;

use Sensio\Bundle\FrameworkExtraBundle\Configuration\Route;
use Sensio\Bundle\FrameworkExtraBundle\Configuration\Method;
use Symfony\Bundle\FrameworkBundle\Controller\Controller;

/** @Route("/genres") */
class GenreController extends Controller
{
    /**
     * @Route("/", name="smoovio_api_genres", defaults={"_format"="json"})
     * @Method("GET")
     */
    public function getGenresAction()
    {
        return $this->render('SmoovioApiBundle:Genre:genres.json.twig', [
            'genres' => $this->get('smoovio_core.repository.genre')->getGenres(),
        ]);
    }
}
```

Smoovio > Genres

```
# SmoovioApiBundle:Genre:genres.json.twig
[
{% for genre in genres %}
    {
        "id"    : "{{ genre.id }}",
        "slug" : "{{ genre.slug }}",
        "title": "{{ genre.title }}"
    }
    {{ not loop.last ? ',' : '' }}
{% endfor %}
]
```

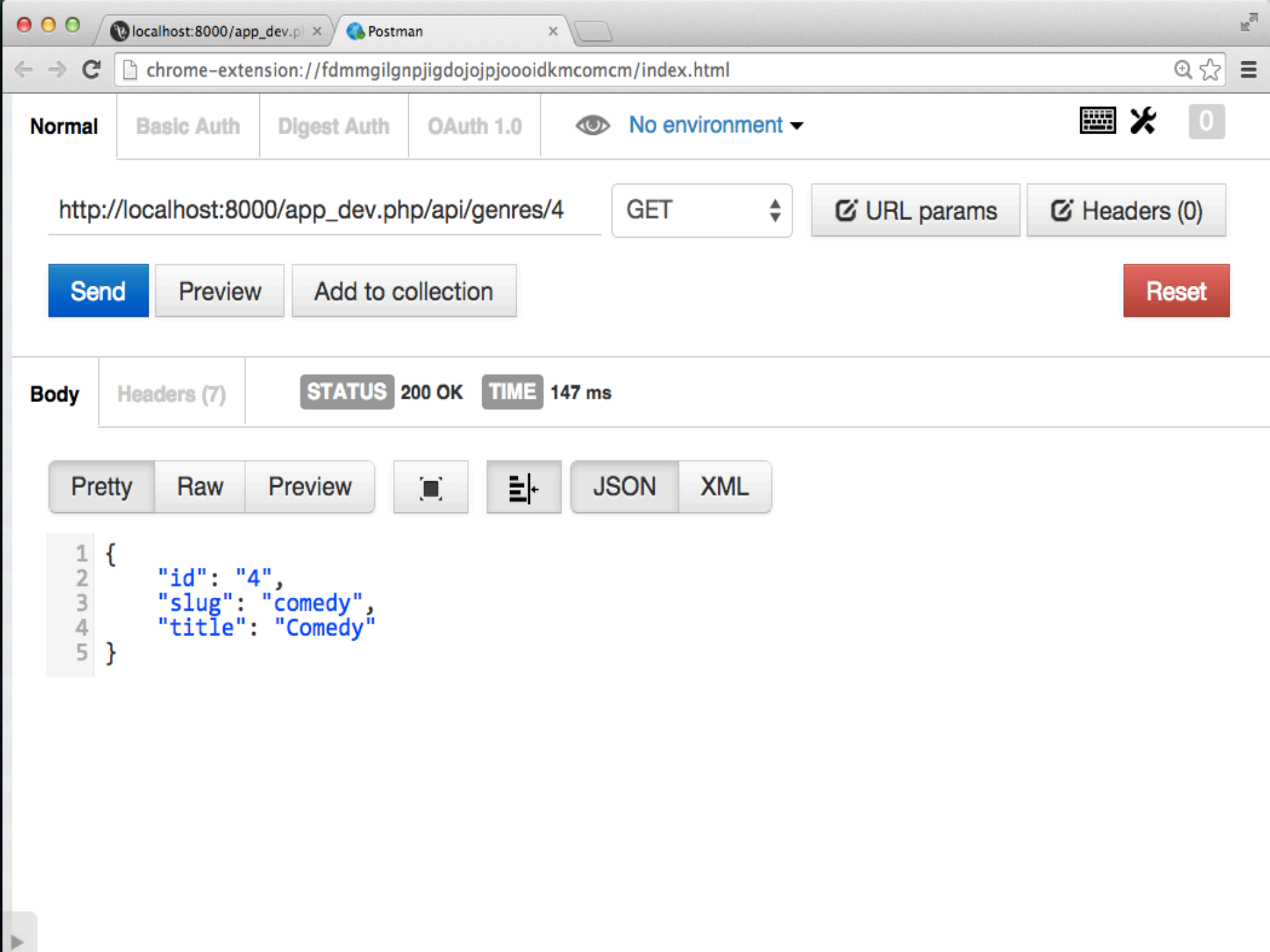

localhost:8000/app_dev.p
Postman
chrome-extension://fdmmgilgnpjigdojojpjoooidkmcomcm/index.html
Normal
Basic Auth
Digest Auth
OAuth 1.0
No environment
0
http://localhost:8000/app_dev.php/api/genres/
GET
URL params
Headers (0)
Send
Preview
Add to collection
Reset
Body
Headers (7)
STATUS 200 OK
TIME 137 ms
Pretty
Raw
Preview
JSON
XML
[
{
"id": "2",
"slug": "action",
"title": "Action"
},
{
"id": "6",
"slug": "adventure",
"title": "Adventure"
},
{
"id": "9",
"slug": "animation",
"title": "Animation"
},
{

Smoovio > Single Genre

```php
/** @Routing\Route("/genres") */
class GenreController extends Controller
{
    /**
     * @Route(
     *     path         = "/{id}",
     *     name         = "smoovio_api_genre",
     *     defaults     = {"_format"="json"},
     *     requirements = {"id"="\d+"}
     * )
     * @Method("GET")
     */
    public function getGenreAction($id)
    {
        if (!$genre = $this->get('smoovio_core.repository.genre')->find($id)) {
            throw $this->createNotFoundException(sprintf('No Genre found with id #%u.', $id));
        }
        return $this->render('SmoovioApiBundle:Genre:genre.json.twig', ['genre' => $genre ]);
    }
}
```

Smoovio > Genres

```
# SmoovioApiBundle:Genre:genre.json.twig
{
    "id"    : "{{ genre.id }}",
    "slug"  : "{{ genre.slug }}",
    "title": "{{ genre.title }}"
}
```


localhost:8000/app_dev.p
Postman
chrome-extension://fdmmgilgnpjigdojojpjoooidkmcomcm/index.html
Normal
Basic Auth
Digest Auth
OAuth 1.0
No environment
0
http://localhost:8000/app_dev.php/api/genres/4
GET
URL params
Headers (0)
Send
Preview
Add to collection
Reset
Body
Headers (7)
STATUS 200 OK
TIME 147 ms
Pretty
Raw
Preview
JSON
XML
{
"id": "4",
"slug": "comedy",
"title": "Comedy"
}

Smoovio > Refactoring

- Use Doctrine Param Converters,
- Avoid Twig code duplications,

```php
use Smoovio\Bundle\CoreBundle\Entity\Genre;
// ...
class GenreController extends Controller {
    // ...
    public function getGenreAction(Genre $genre) {
        return $this->render('...', ['genre' => $genre]);
    }
}
```

```twig
# SmoovioApiBundle:Genre:genres.json.twig
[
{% for genre in genres %}
    {{ include('SmoovioApiBundle:Genre:genre.json.twig') }}
    {{ not loop.last ? ',' : '' }}
{% endfor %}
]
```

User Story #3

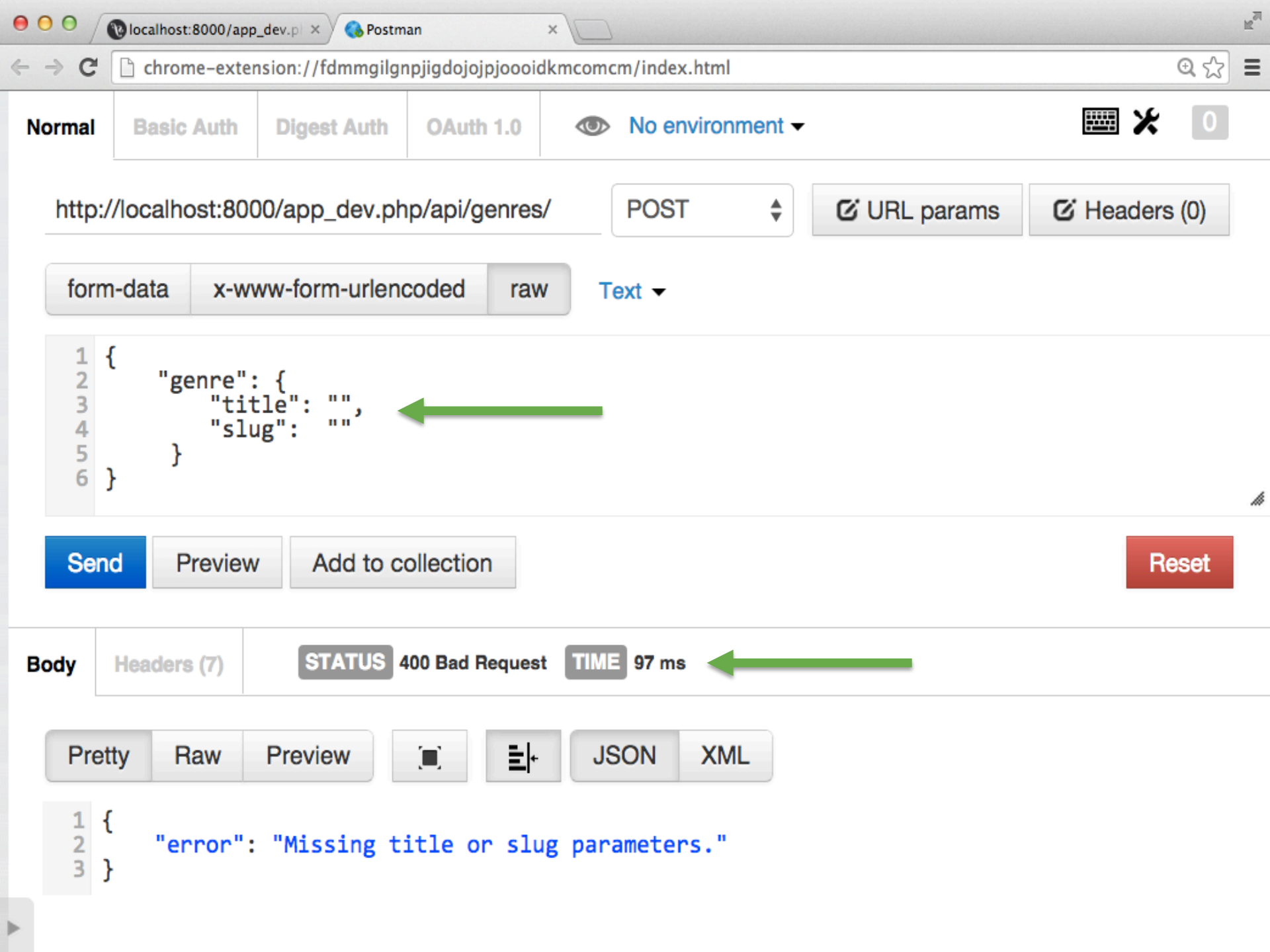
As a consumer, I want to create a new movie genre.

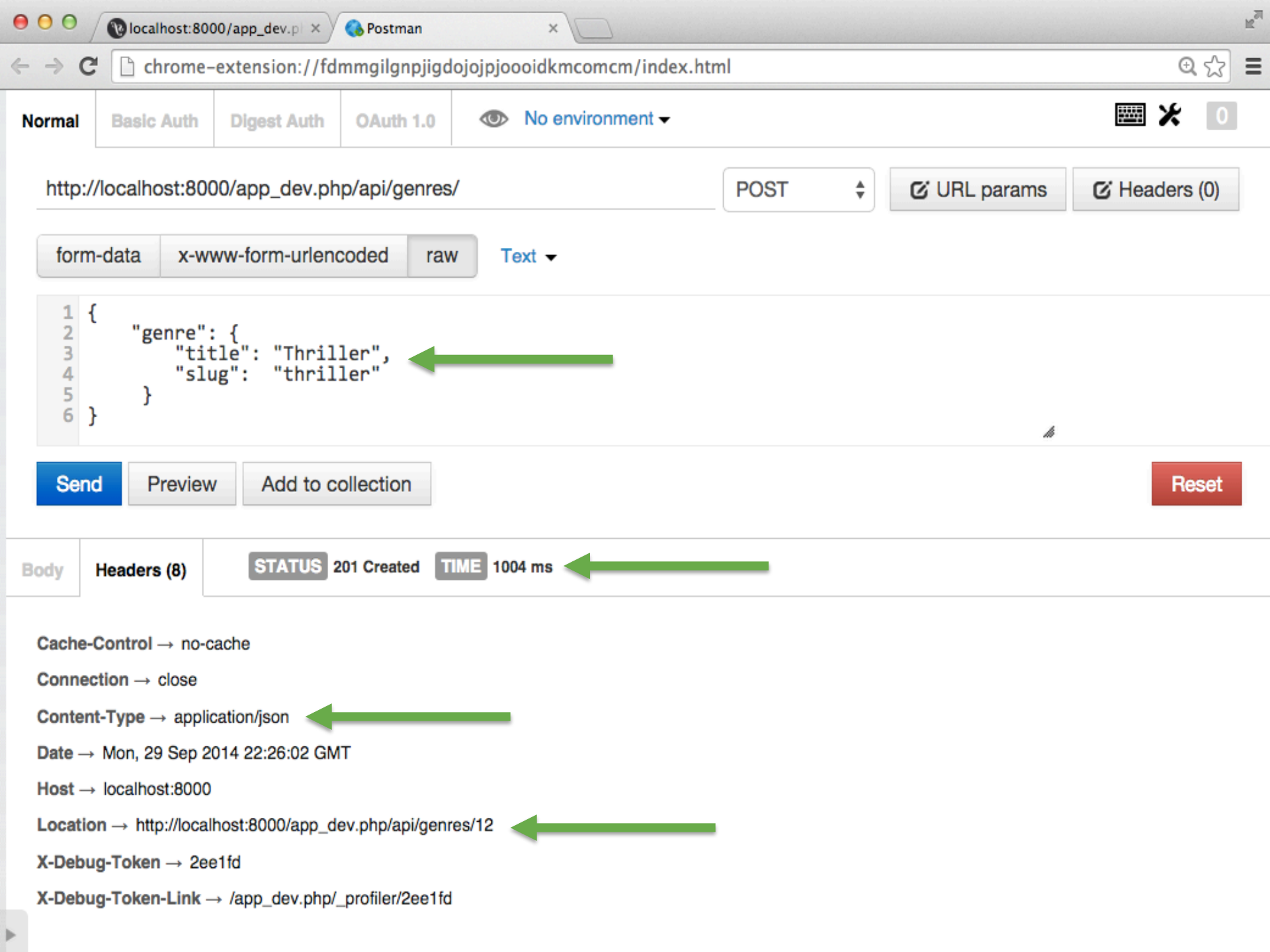
Smoovio > Create Single Genre

- Use POST HTTP method to create genre,
- Send data as JSON in request's body,
- Decode JSON data to PHP arrays,
- Return a 201 response if request is successful,
- Return a 400 response if request fails,

```php
// …

use Symfony\Component\HttpFoundation\JsonResponse;
use Symfony\Component\HttpFoundation\ParameterBag;
use Symfony\Component\HttpFoundation\Request;
use Symfony\Component\HttpFoundation\Response;

// …
class GenreController extends Controller
{
    // …
    private function decodeJsonBody(Request $request)
    {
        $parameters = json_decode($request->getContent(), true);
        $request->request = new ParameterBag($parameters);
    }
}
```

```php
// …
class GenreController extends Controller
{
    /**
     * @Route("/", name="smoovio_api_new_genre", defaults={"_format"="json"})
     * @Method("POST")
     */
    public function postGenreAction(Request $request)
    {
        $this->decodeJsonBody($request);
        $data = $request->request->get('genre');

        if (empty($data['title']) || empty($data['slug'])) {
            return new JsonResponse([ 'error' => 'Missing title or slug.'], 400);
        }

        $genre = new Genre($data['title'], $data['slug']);
        $em = $this->getDoctrine()->getManager();
        $em->persist($genre);
        $em->flush();

        $location = $this->generateUrl('smoovio_api_genre', [ 'id' => $genre->getId() ], true);

        return new JsonResponse('', 201, [ 'Location' => $location ]);
    }
}
```


localhost:8000/app_dev.p
Postman
chrome-extension://fdmmgilgnpjigdojojpjoooidkmcomcm/index.html
Normal
Basic Auth
Digest Auth
OAuth 1.0
No environment
0
http://localhost:8000/app_dev.php/api/genres/
POST
URL params
Headers (0)
form-data
x-www-form-urlencoded
raw
Text
{
"genre": {
"title": "",
"slug": ""
}
}
Send
Preview
Add to collection
Reset
Body
Headers (7)
STATUS 400 Bad Request
TIME 97 ms
Pretty
Raw
Preview
JSON
XML
{
"error": "Missing title or slug parameters."
}


localhost:8000/app_dev.p
Postman
chrome-extension://fdmmgilgnpjigdojojpjoooidkmcomcm/index.html
Normal
Basic Auth
Digest Auth
OAuth 1.0
No environment
0
http://localhost:8000/app_dev.php/api/genres/
POST
URL params
Headers (0)
form-data
x-www-form-urlencoded
raw
Text
1 {
2 "genre": {
3 "title": "Thriller",
4 "slug": "thriller"
5 }
6 }
Send
Preview
Add to collection
Reset
Body
Headers (8)
STATUS 201 Created
TIME 1004 ms
Cache-Control → no-cache
Connection → close
Content-Type → application/json
Date → Mon, 29 Sep 2014 22:26:02 GMT
Host → localhost:8000
Location → http://localhost:8000/app_dev.php/api/genres/12
X-Debug-Token → 2ee1fd
X-Debug-Token-Link → /app_dev.php/_profiler/2ee1fd

User Story #4

As a consumer, I want to edit a movie genre.

Smoovio > Update a Single Genre

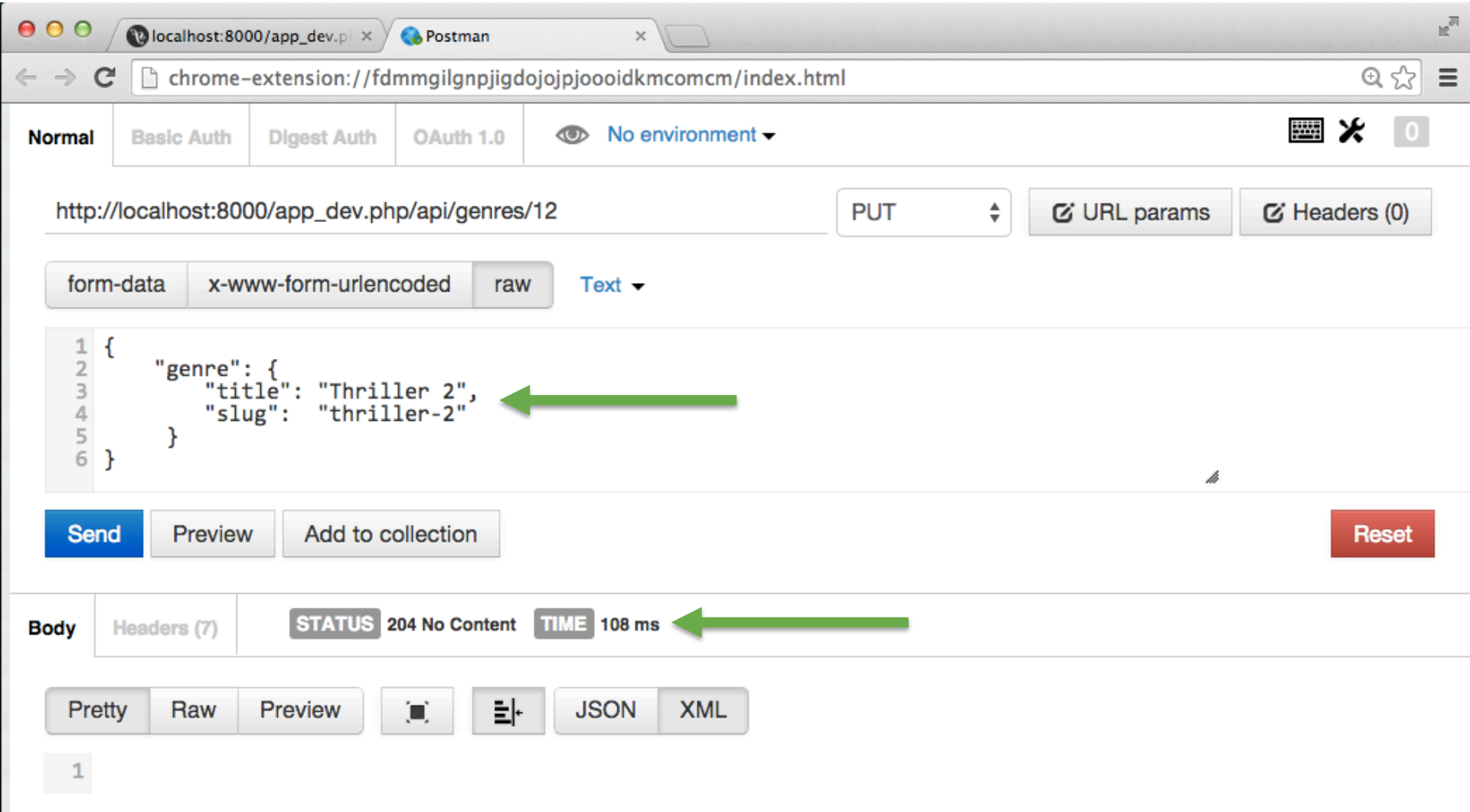
- Use PUT HTTP method to update existing genre,
- Use Doctrine Param Converter
- Decode JSON data to PHP arrays,
- Return a 204 response if request is successful,
- Return a 404 response if resource doesn't exist,
- Return a 400 response if request fails.

Smoovio > Update a Single Genre

```
class GenreController extends Controller
{
    /**
     * @Route(
     *      path         = "/{id}",
     *      name         = "smoovio_api_edit_genre",
     *      defaults     = {"_format"="json"},
     *      requirements = {"id"="\d+"}
     * )
     * @Method("PUT")
     */
    public function putGenreAction(Genre $genre, Request $request)
    {
        // ...
    }
}
```

```php
class GenreController extends Controller
{
    // ...
    public function putGenreAction(Genre $genre, Request $request)
    {
        $this->decodeJsonBody($request);

        $data = $request->request->get('genre');

        if (empty($data['title']) || empty($data['slug'])) {
            return new JsonResponse(
                [ 'error' => 'Missing title or slug parameters.'],
                Response::HTTP_BAD_REQUEST
            );
        }

        $genre->setSlug($data['slug']);
        $genre->setTitle($data['title']);

        $em = $this->getDoctrine()->getManager();
        $em->flush($genre);

        return new JsonResponse('', Response::HTTP_NO_CONTENT);
    }
}
```

Smoovio > Update a Single Genre



User Story #5

As a consumer, I want to delete a movie genre.

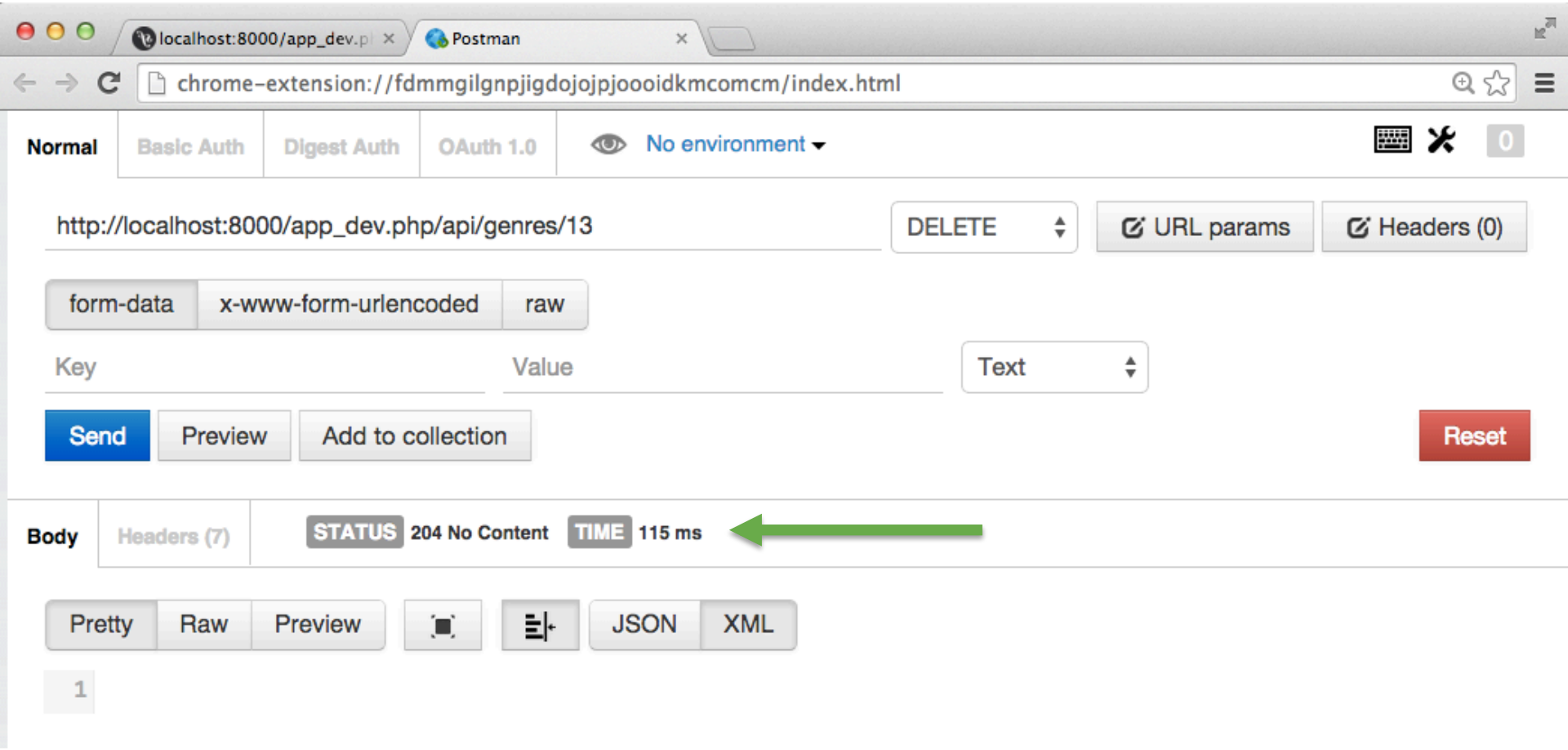
Smoovio > Delete a Single Genre

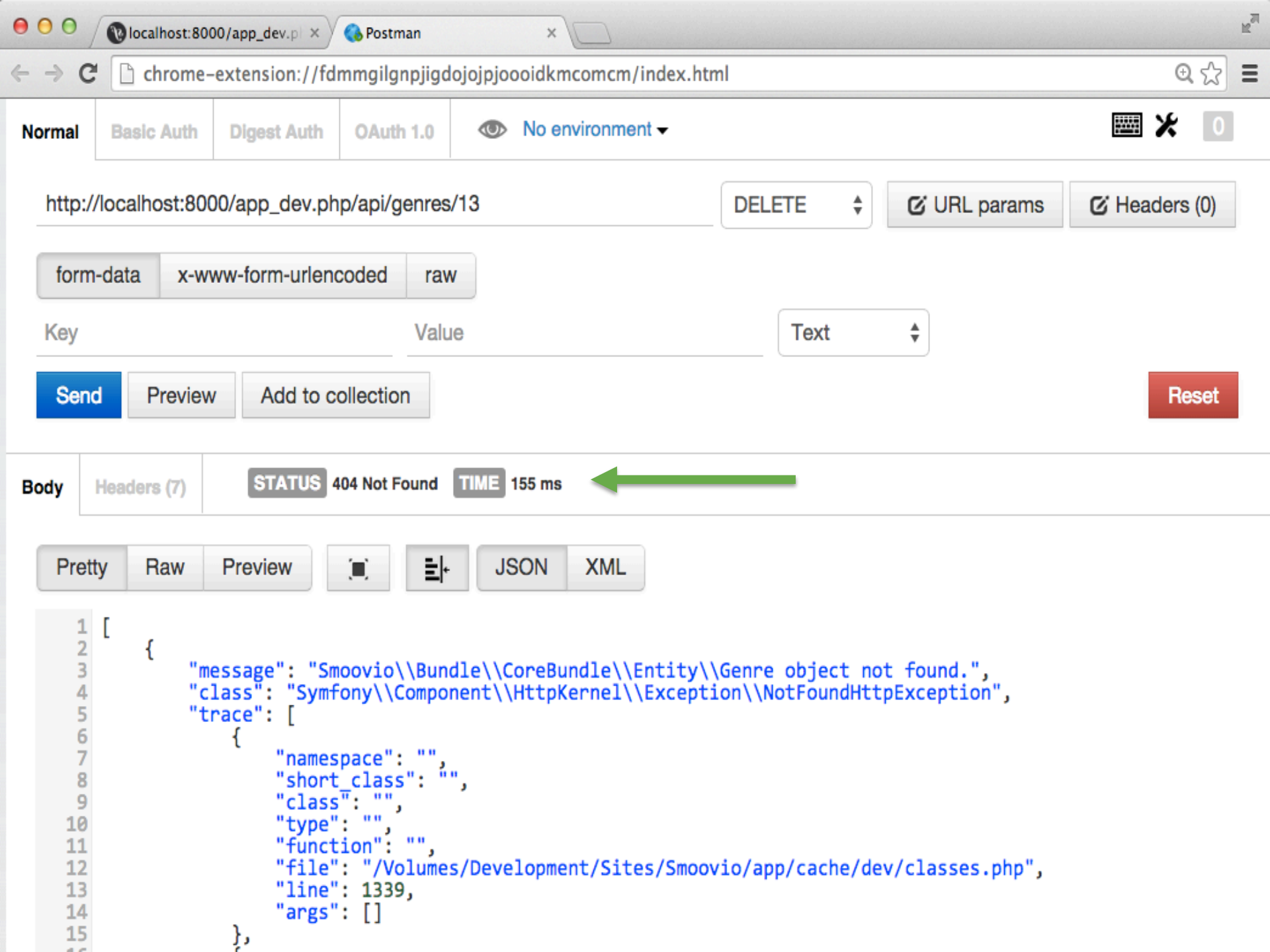
- Use DELETE HTTP method to delete a genre,
- Use Doctrine Param Converter,
- Return a 204 response if request is successful,
- Return a 404 response if resource doesn't exist.

Smoovio > Delete a Single Genre

```
class GenreController extends Controller
{
    /**
     * @Route(
     *      path         = "/{id}",
     *      name         = "smoovio_api_delete_genre",
     *      defaults     = {"_format"="json"},
     *      requirements = {"id"="\d+"}
     * )
     * @Method("DELETE")
     */
    public function deleteGenreAction(Genre $genre)
    {
        $em = $this->getDoctrine()->getManager();
        $em->remove($genre);
        $em->flush();

        return new JsonResponse('', Response::HTTP_NO_CONTENT);
    }
}
```

Smoovio > Delete a Single Genre




localhost:8000/app_dev.p
Postman
chrome-extension://fdmmgilgnpjigdojojpjoooidkmcomcm/index.html
Normal
Basic Auth
Digest Auth
OAuth 1.0
No environment
0
http://localhost:8000/app_dev.php/api/genres/13
DELETE
URL params
Headers (0)
form-data
x-www-form-urlencoded
raw
Key
Value
Text
Send
Preview
Add to collection
Reset
Body
Headers (7)
STATUS 404 Not Found
TIME 155 ms
Pretty
Raw
Preview
JSON
XML
[
{
"message": "Smoovio\\Bundle\\CoreBundle\\Entity\\Genre object not found.",
"class": "Symfony\\Component\\HttpKernel\\Exception\\NotFoundHttpException",
"trace": [
{
"namespace": "",
"short_class": "",
"class": "",
"type": "",
"function": "",
"file": "/Volumes/Development/Sites/Smoovio/app/cache/dev/classes.php",
"line": 1339,
"args": []
},

Ok, we have a very basic CRUD JSON API! Now what?!

The current code works but has **major drawbacks** and **limitations**...

Smoovio > API > Drawbacks & Limitations

- Lots of duplicated code
- No data input validation
- Only JSON support
- Basic error handling
- …

Introduction to Content Negotiation

HTTP Content Negotiation

Content negotiation is a mechanism defined in the HTTP specification that makes it possible to serve different versions of a resource / document at the same URI, so that user agents can specify which version fit their capabilities the best.

Wikipedia

Content Negotiation HTTP Headers

```
Accept:          text/html; q=1.0, text/*; q=0.8
Accept-Language: fr;q=1.0, en-gb;q=0.8, en;q=0.7
Accept-Charset:  iso-8859-5, unicode-1-1;q=0.8
Accept-Encoding: compress;q=0.5, gzip;q=1.0
User-Agent:      CERN-LineMode/2.15 libwww/2.17b3
Vary:            Accept
```

Format Negotiation

```php
$negotiator = new \Negotiation\FormatNegotiator();

$accept = 'text/html,application/xml;q=0.9,*/*;q=0.8';
$priorities = ['text/html', 'application/json', '*/*'];

// text/html
echo $negotiator
    ->getBest($accept, $priorities)
    ->getValue();

// html
echo $negotiator->getBestFormat($accept, $priorities);
```

Language Negotiation

```php
use Negotiation\LanguageNegotiator;

$negotiator = new LanguageNegotiator();

$accept = 'da, en-gb;q=0.8, en;q=0.7';
$language = $negotiator->getBest($accept);

echo $language; // da
```

Charset / Encoding Negotiation

```php
$negotiator = new \Negotiation\Negotiator();

$priorities = ['utf-8', 'big5', 'shift-jis'];
$accept = 'ISO-8859-1, Big5;q=0.6,utf-8;q=0.7, *;q=0.5';

// 'utf-8'
$best = $negotiator->getBest($accept, $priorities);
```

Introduction to FOSRestBundle

FOSRestBundle > Built-in features

- Decodes request body in XML / JSON,
- Handles content negotiation,
- Disables CSRF protection on form types,
- Provides some useful controller annotations,
- Supports JMSSerializerBundle

FOSRestBundle > Configuration

```
# app/config/config.yml
fos_rest:
    body_converter:
        enabled: true
        validate: true
        validation_errors_argument: validationErrors
    view:
        view_response_listener: force
        formats: { json: true, xml: true, rss: false }
    param_fetcher_listener: force
    serializer:
        serialize_null: true
```

FOSRestBundle > Route Annotations

```
@Prefix("/api")
@Head("/genres")
@Get("/genres/{id}")
@Post("/genres")
@Put("/genres/{id}")
@Patch("/genres/{id}")
@Delete("/genres/{id}")
@Link("/movies/{id}/actors")
@Unlink("/movies/{id}/actors")
@Options("/movies")
```

FOSRestBundle > Route Annotation Example

```
/**
 * @Rest\Put(
 *      path = "/genres/{id}",
 *      name = "smoovio_api_edit_genre",
 *      requirements = {"id"="\d+"}
 * )
 */
```

FOSRestBundle > View Annotation

```
/**
 * @Rest\View(
 *  templateVar="",
 *  statusCode=null,
 *  serializerGroups={},
 *  populateDefaultVars=true,
 *  serializerEnableMaxDepthChecks=false
 * )
 */
```

FOSRestBundle > View Annotation Example

```
/**
 * @Rest\View(
 *    "AcmeBlogBundle:Post:new.html.twig",
 *    statusCode = 400,
 *    serializerGroups= {"POST_CREATE"}
 * )
 */
```

FOSRestBundle > The FOSRestController class

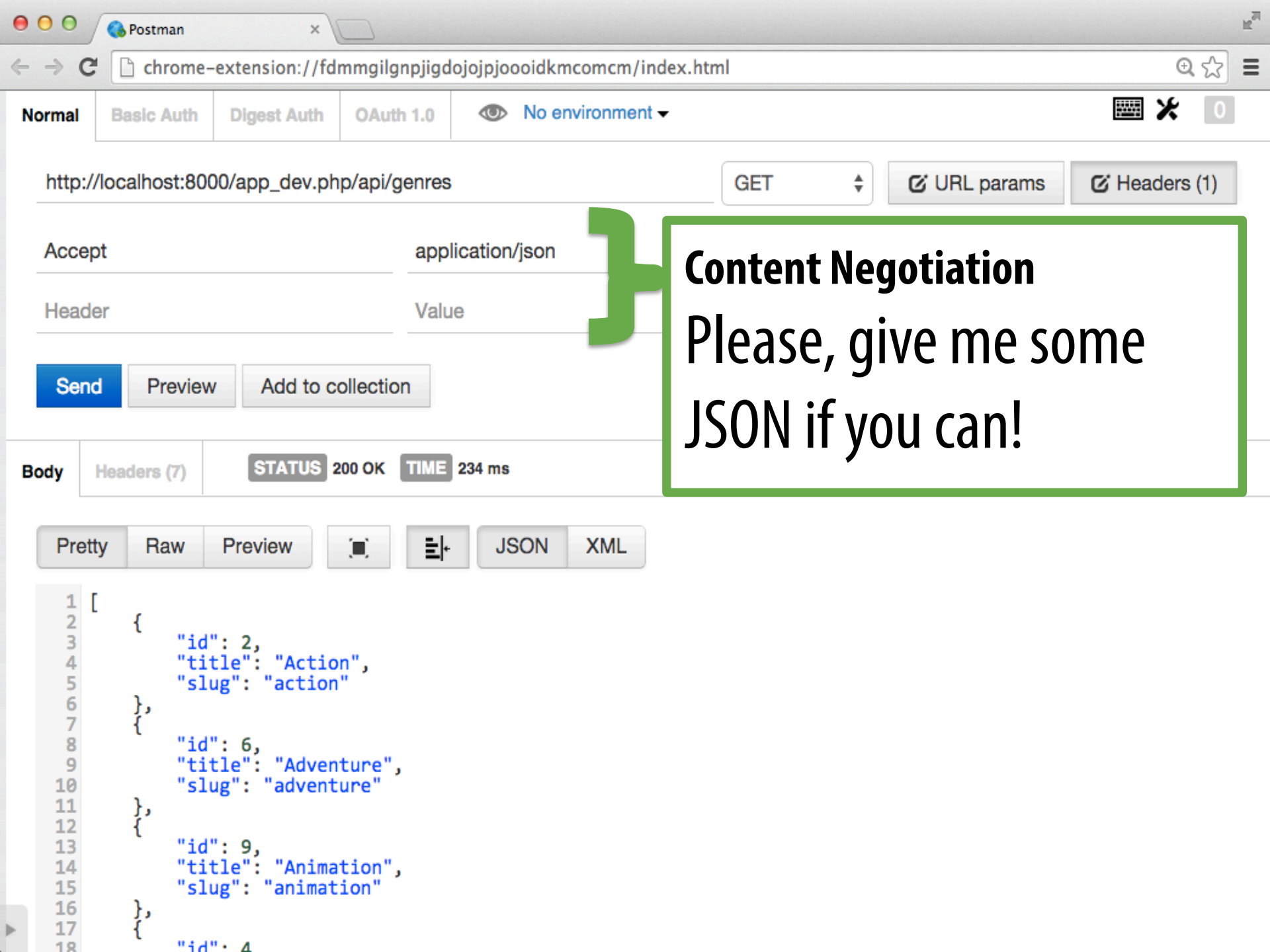
```php
namespace FOS\RestBundle\Controller;

abstract class FOSRestController extends Controller
{
    protected function view(...);
    protected function redirectView(...);
    protected function routeRedirectView(...);
    protected function handleView(...);
}
```

FOSRestBundle > Exercises

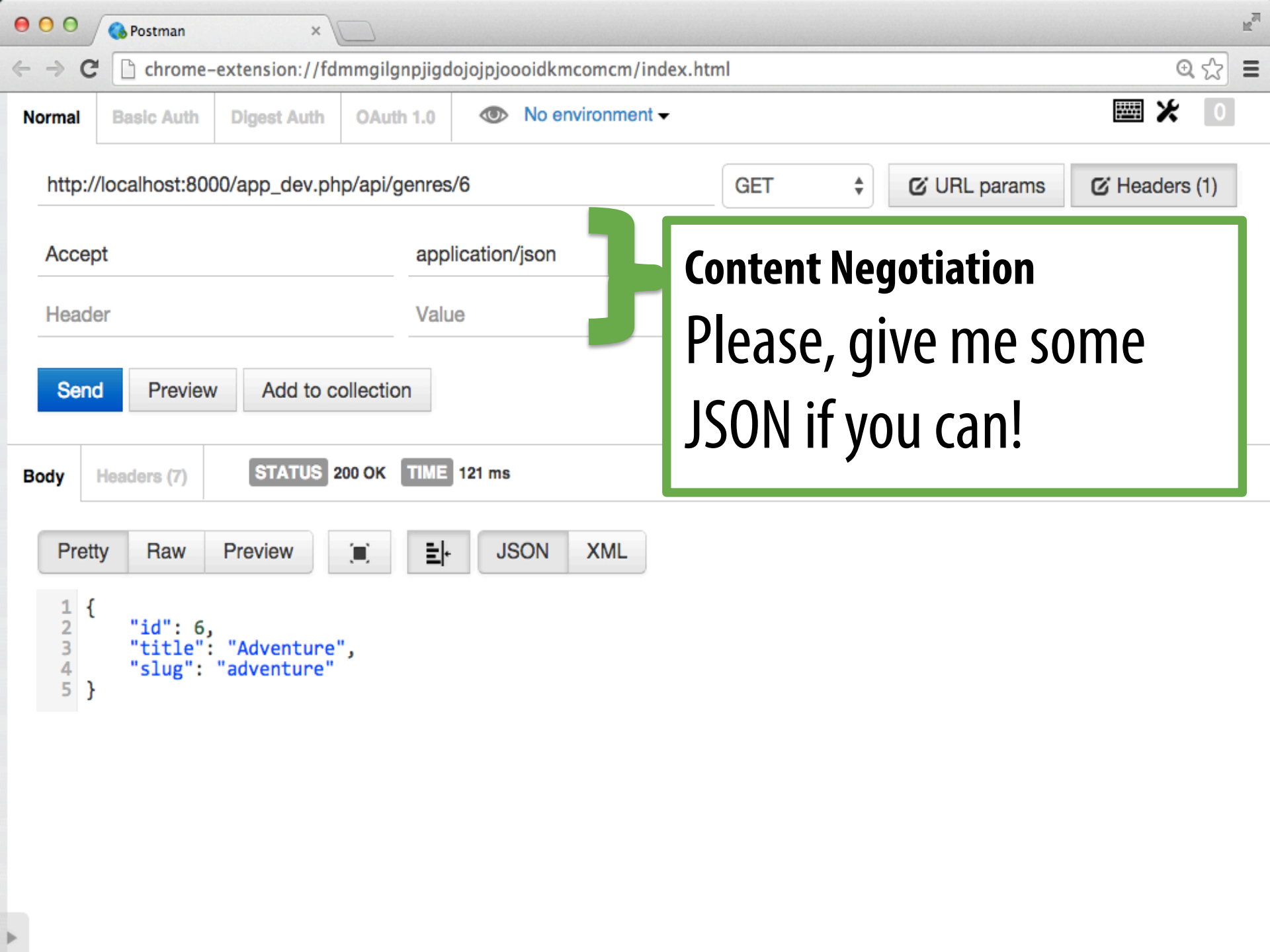
- Use GET/POST/PUT/DELETE route annotations,
- Use base FOSRestController class,
- Remove decodeJsonBody() method,
- Remove « _format » constraint in routes.

FOSRestBundle > Refactoring Actions

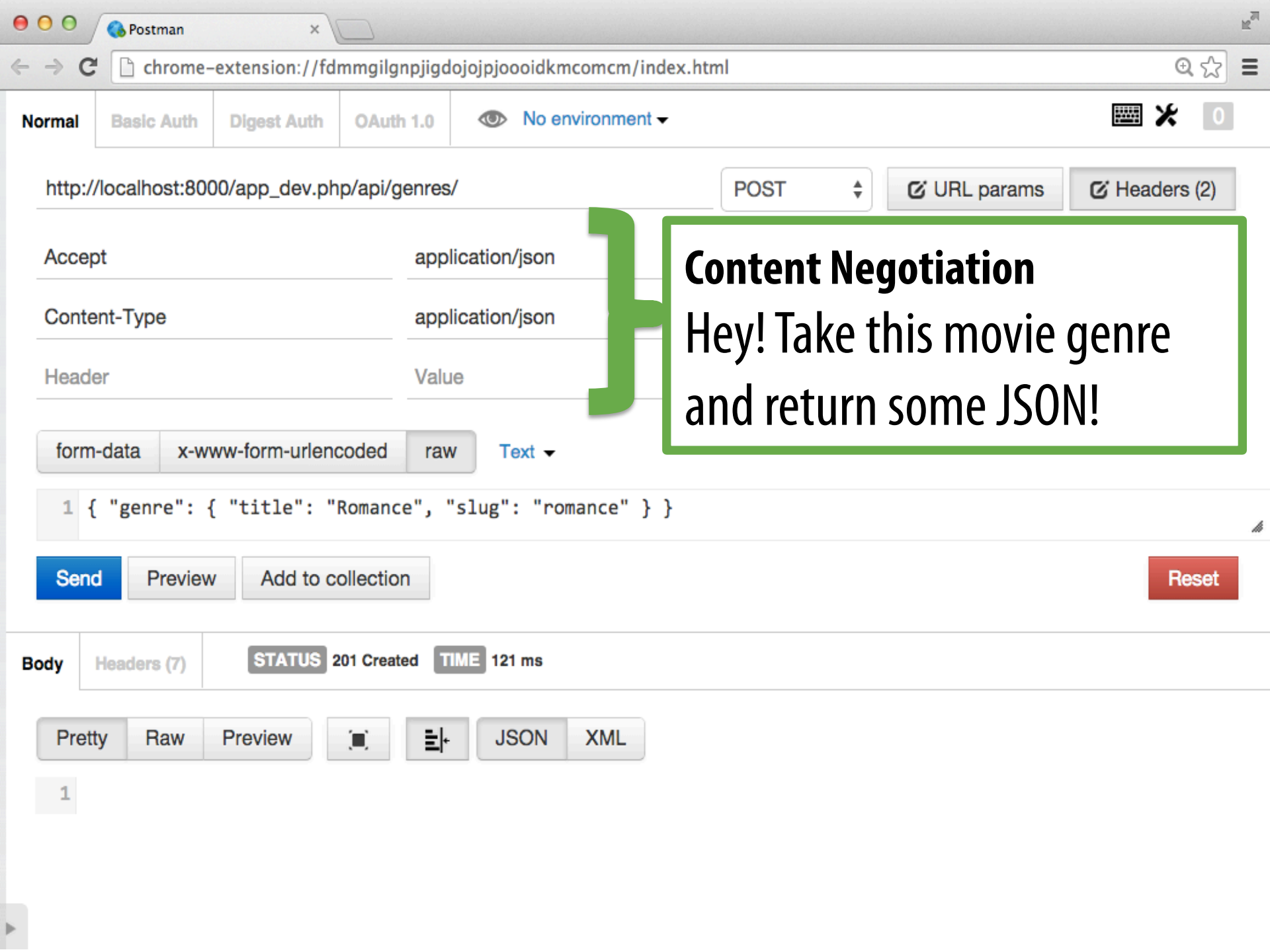
```php
namespace Smoovio\Bundle\ApiBundle\Controller;

use FOS\RestBundle\Controller\FOSRestController;
use FOS\RestBundle\Controller\Annotations as Rest;
// ...

/** @Route("/genres") */
class GenreController extends FOSRestController
{
    /** @Rest\Get("/", name="smoovio_api_genres") */
    public function getGenresAction()
    {
        return $this->get('smoovio_core.repository.genre')->getGenres();
    }
}
```


Postman
chrome-extension://fdmmgilgnpjigdojojpjoooidkmcomcm/index.html
Normal
Basic Auth
Digest Auth
OAuth 1.0
No environment
http://localhost:8000/app_dev.php/api/genres
GET
URL params
Headers (1)
Accept
application/json
Header
Value
Send
Preview
Add to collection
Content Negotiation
Please, give me some JSON if you can!
Body
Headers (7)
STATUS 200 OK
TIME 234 ms
Pretty
Raw
Preview
JSON
XML
[
{
"id": 2,
"title": "Action",
"slug": "action"
},
{
"id": 6,
"title": "Adventure",
"slug": "adventure"
},
{
"id": 9,
"title": "Animation",
"slug": "animation"
},
{

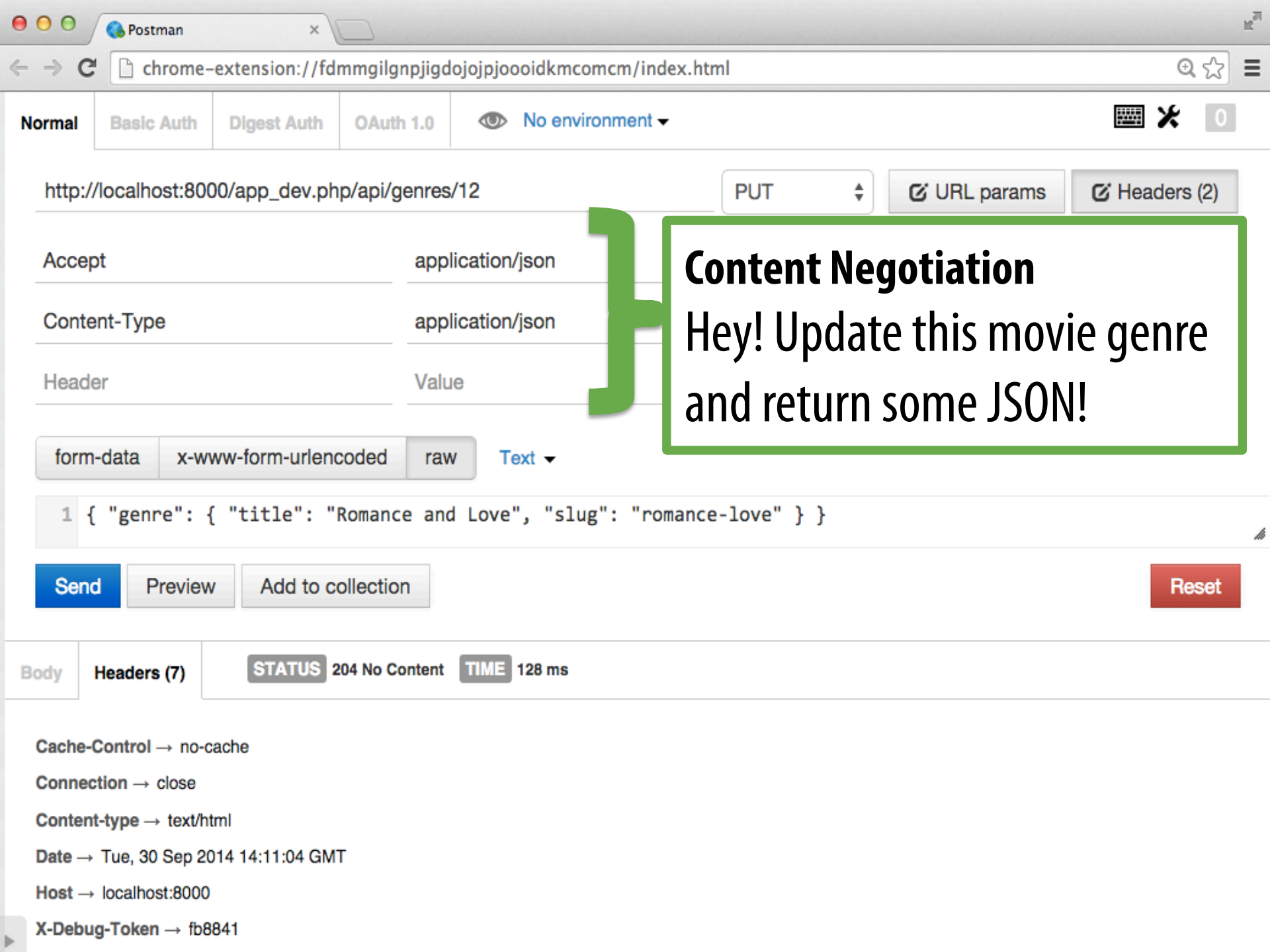
```php
// ...
class GenreController extends FOSRestController
{
    /**
     * @Rest\Get(
     *    path = "/{id}",
     *    name="smoovio_api_genre",
     *    requirements={"id"="\d+"}
     * )
     */
    public function getGenreAction(Genre $genre)
    {
        return $genre;
    }
}
```


Postman
chrome-extension://fdmmgilgnpjigdojojpjoooidkmcomcm/index.html
Normal
Basic Auth
Digest Auth
OAuth 1.0
No environment
http://localhost:8000/app_dev.php/api/genres/6
GET
URL params
Headers (1)
Accept
application/json
Header
Value
Send
Preview
Add to collection
Content Negotiation
Please, give me some JSON if you can!
Body
Headers (7)
STATUS 200 OK
TIME 121 ms
Pretty
Raw
Preview
JSON
XML
{
"id": 6,
"title": "Adventure",
"slug": "adventure"
}

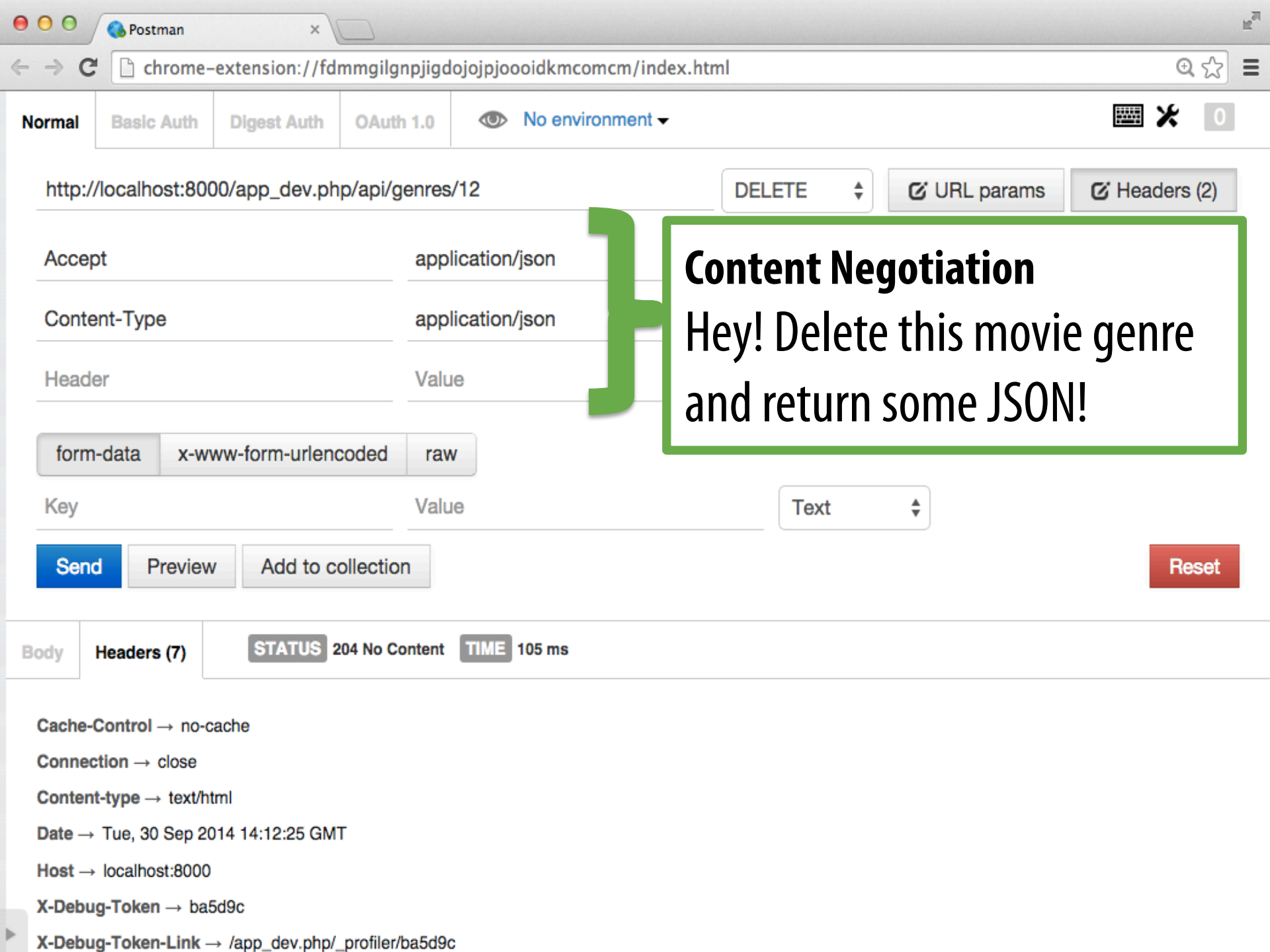
```php
// ...
class GenreController extends FOSRestController
{
    /** @Rest\Post("/", name="smoovio_api_new_genre") */
    public function postGenreAction(Request $request)
    {
        $data = $request->request->get('genre');
        if (empty($data['title']) || empty($data['slug'])) {
            return $this->view([ 'error' => '...'], 400);
        }

        // ...

        return $this->view(null, 201, [ 'Location' => $location ]);
    }
}
```


Postman
chrome-extension://fdmmgilgnpjigdojojpjoooidkmcomcm/index.html
Normal
Basic Auth
Digest Auth
OAuth 1.0
No environment
http://localhost:8000/app_dev.php/api/genres/
POST
URL params
Headers (2)
Accept
application/json
Content-Type
application/json
Header
Value
Content Negotiation
Hey! Take this movie genre and return some JSON!
form-data
x-www-form-urlencoded
raw
Text
1 { "genre": { "title": "Romance", "slug": "romance" } }
Send
Preview
Add to collection
Reset
Body
Headers (7)
STATUS 201 Created
TIME 121 ms
Pretty
Raw
Preview
JSON
XML
1

```php
// ...
class GenreController extends FOSRestController
{
    /**
     * @Rest\Put(
     *     path = "/{id}",
     *     name = "smoovio_api_edit_genre",
     *     requirements = {"id"="\d+"}
     * )
     */
    public function putGenreAction(Genre $genre, Request $request)
    {
        $data = $request->request->get('genre');
        if (empty($data['title']) || empty($data['slug'])) {
            return $this->view([ 'error' => '...'], 400);
        }
        // ...
        return $this->view(null, 204);
    }
}
```

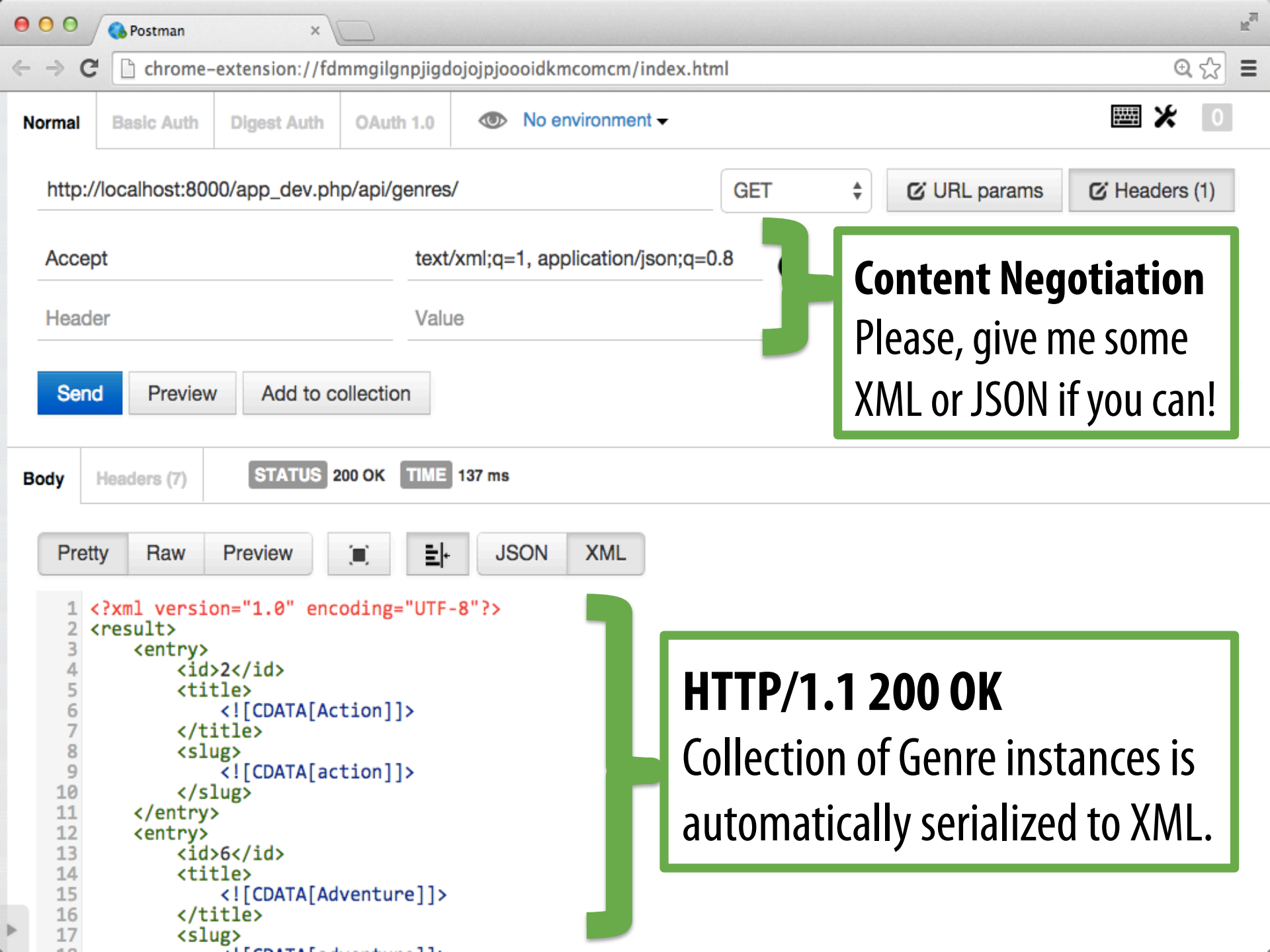

Postman
chrome-extension://fdmmgilgnpjigdojojpjoooidkmcomcm/index.html
Normal
Basic Auth
Digest Auth
OAuth 1.0
No environment
0
http://localhost:8000/app_dev.php/api/genres/12
PUT
URL params
Headers (2)
Accept
application/json
Content-Type
application/json
Header
Value
Content Negotiation
Hey! Update this movie genre and return some JSON!
form-data
x-www-form-urlencoded
raw
Text
1 { "genre": { "title": "Romance and Love", "slug": "romance-love" } }
Send
Preview
Add to collection
Reset
Body
Headers (7)
STATUS 204 No Content
TIME 128 ms
Cache-Control → no-cache
Connection → close
Content-type → text/html
Date → Tue, 30 Sep 2014 14:11:04 GMT
Host → localhost:8000
X-Debug-Token → fb8841

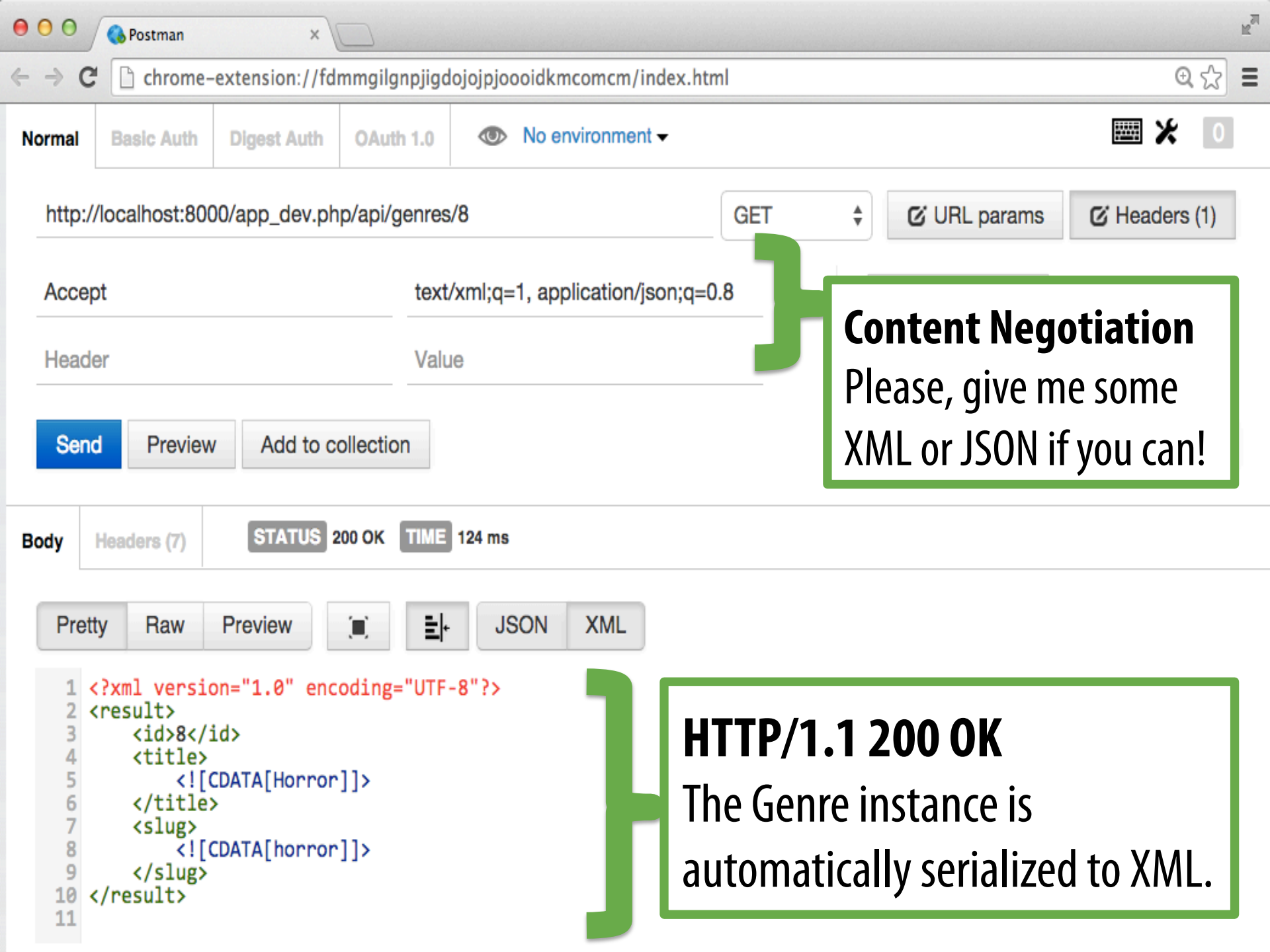
```php
// ...
class GenreController extends FOSRestController
{
    /**
     * @Rest\Delete(
     *     path         = "/{id}",
     *     name         = "smoovio_api_delete_genre",
     *     requirements = {"id"="\d+"}
     * )
     */
    public function deleteGenreAction(Genre $genre)
    {
        $em = $this->getDoctrine()->getManager();
        $em->remove($genre);
        $em->flush();

        return $this->view(null, 204);
    }
}
```


Postman
chrome-extension://fdmmgilgnpjigdojojpjoooidkmcomcm/index.html
Normal
Basic Auth
Digest Auth
OAuth 1.0
No environment
0
http://localhost:8000/app_dev.php/api/genres/12
DELETE
URL params
Headers (2)
Accept
application/json
Content-Type
application/json
Header
Value
Content Negotiation
Hey! Delete this movie genre and return some JSON!
form-data
x-www-form-urlencoded
raw
Key
Value
Text
Send
Preview
Add to collection
Reset
Body
Headers (7)
STATUS 204 No Content
TIME 105 ms
Cache-Control → no-cache
Connection → close
Content-type → text/html
Date → Tue, 30 Sep 2014 14:12:25 GMT
Host → localhost:8000
X-Debug-Token → ba5d9c
X-Debug-Token-Link → /app_dev.php/_profiler/ba5d9c

User Story #7

As a consumer, I want to get movies genres as XML.


Postman
chrome-extension://fdmmgilgnpjigdojojpjoooidkmcomcm/index.html
Normal
Basic Auth
Digest Auth
OAuth 1.0
No environment
http://localhost:8000/app_dev.php/api/genres/
GET
URL params
Headers (1)
Accept
text/xml;q=1, application/json;q=0.8
Header
Value
Send
Preview
Add to collection
Content Negotiation
Please, give me some XML or JSON if you can!
Body
Headers (7)
STATUS 200 OK
TIME 137 ms
Pretty
Raw
Preview
JSON
XML
1 <?xml version="1.0" encoding="UTF-8"?>
2 <result>
3 <entry>
4 <id>2</id>
5 <title>
6 <![CDATA[Action]]>
7 </title>
8 <slug>
9 <![CDATA[action]]>
10 </slug>
11 </entry>
12 <entry>
13 <id>6</id>
14 <title>
15 <![CDATA[Adventure]]>
16 </title>
17 <slug>
HTTP/1.1 200 OK
Collection of Genre instances is automatically serialized to XML.


Postman
chrome-extension://fdmmgilgnpjigdojojpjoooidkmcomcm/index.html
Normal
Basic Auth
Digest Auth
OAuth 1.0
No environment
http://localhost:8000/app_dev.php/api/genres/8
GET
URL params
Headers (1)
Accept
text/xml;q=1, application/json;q=0.8
Header
Value
Send
Preview
Add to collection
Content Negotiation
Please, give me some XML or JSON if you can!
Body
Headers (7)
STATUS 200 OK
TIME 124 ms
Pretty
Raw
Preview
JSON
XML
1 <?xml version="1.0" encoding="UTF-8"?>
2 <result>
3 <id>8</id>
4 <title>
5 <![CDATA[Horror]]>
6 </title>
7 <slug>
8 <![CDATA[horror]]>
9 </slug>
10 </result>
11
HTTP/1.1 200 OK
The Genre instance is automatically serialized to XML.

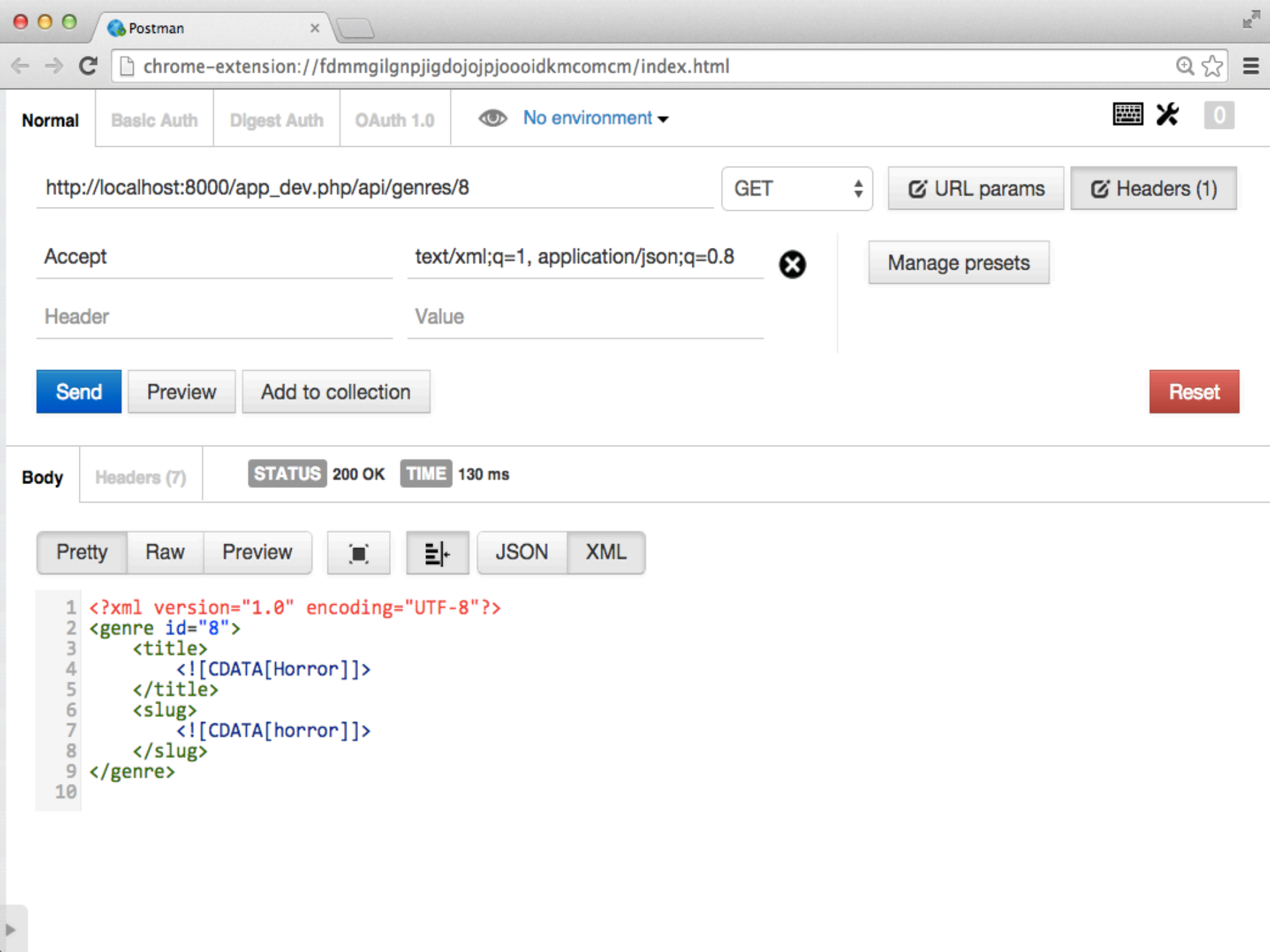
Introduction to JMSSerializerBundle

JMSSerializerBundle > Built-in features

- Encodes objects graphs to standard formats (XML, JSON…),
- Decodes object string representations to object graphs,
- Exposed data are configurable via YAML, XML or annotations,
- Natively integrated with FOSRestBundle.

JMSSerializerBundle > Exposing data as XML

```
# src/Smoovio/Bundle/CoreBundle/Entity/Genre.php

use JMS\Serializer\Annotation as Serializer;

/**
 * @Serializer\ExclusionPolicy("ALL")
 * @Serializer\XmlRoot("genre")
 */
class Genre
{
    /**
     * @Serializer\Expose
     * @Serializer\XmlAttribute
     */
    private $id;

    /** @Serializer\Expose */
    private $title;

    /** @Serializer\Expose */
    private $slug;
}
```


Postman
chrome-extension://fdmmgilgnpjigdojojpjoooidkmcomcm/index.html
Normal
Basic Auth
Digest Auth
OAuth 1.0
No environment
0
http://localhost:8000/app_dev.php/api/genres/8
GET
URL params
Headers (1)
Accept
text/xml;q=1, application/json;q=0.8
Manage presets
Header
Value
Send
Preview
Add to collection
Reset
Body
Headers (7)
STATUS 200 OK
TIME 130 ms
Pretty
Raw
Preview
JSON
XML
1 <?xml version="1.0" encoding="UTF-8"?>
2 <genre id="8">
3 <title>
4 <![CDATA[Horror]]>
5 </title>
6 <slug>
7 <![CDATA[horror]]>
8 </slug>
9 </genre>
10

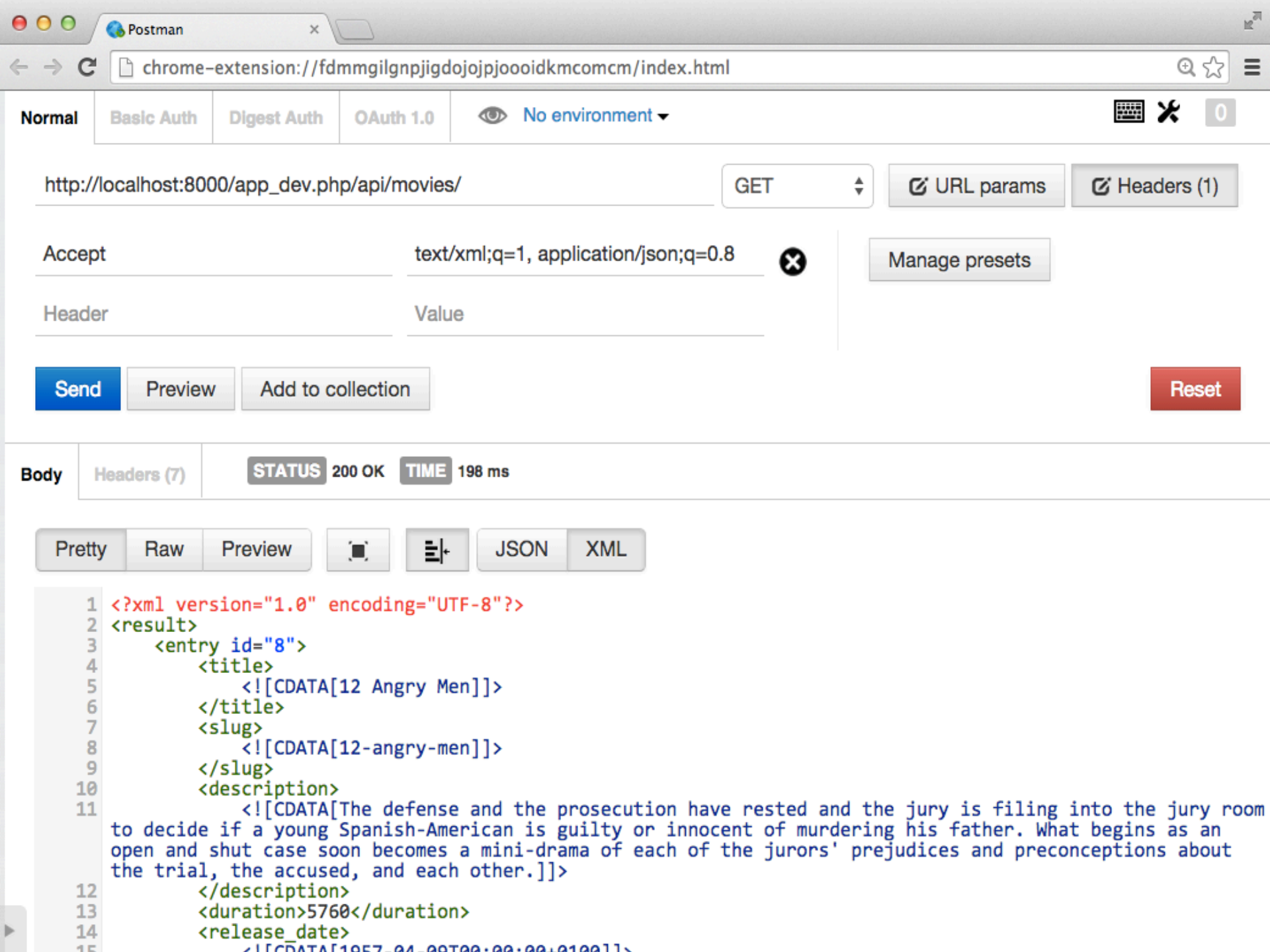
User Story #7

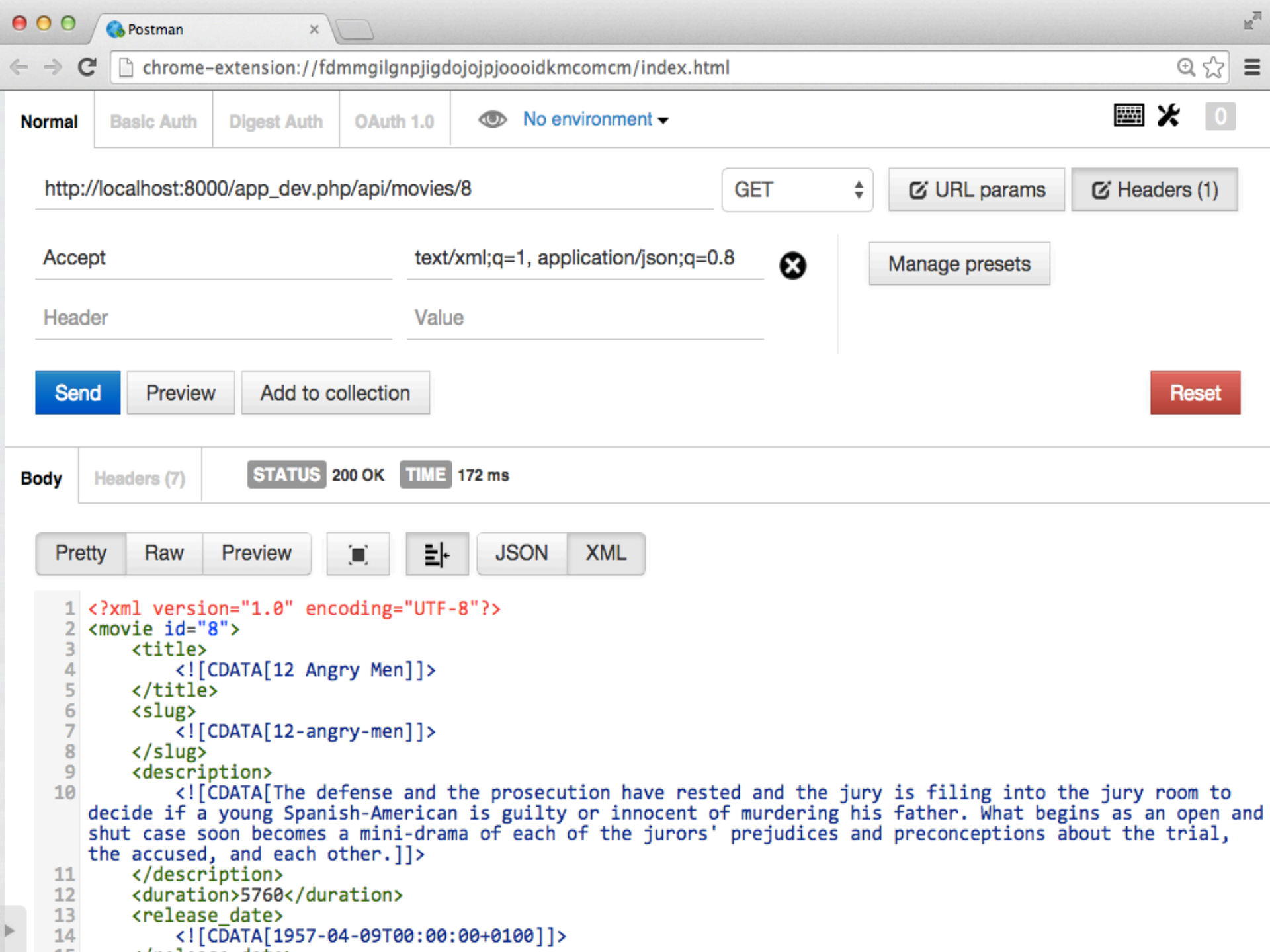
As an API consumer, I want to get movies.

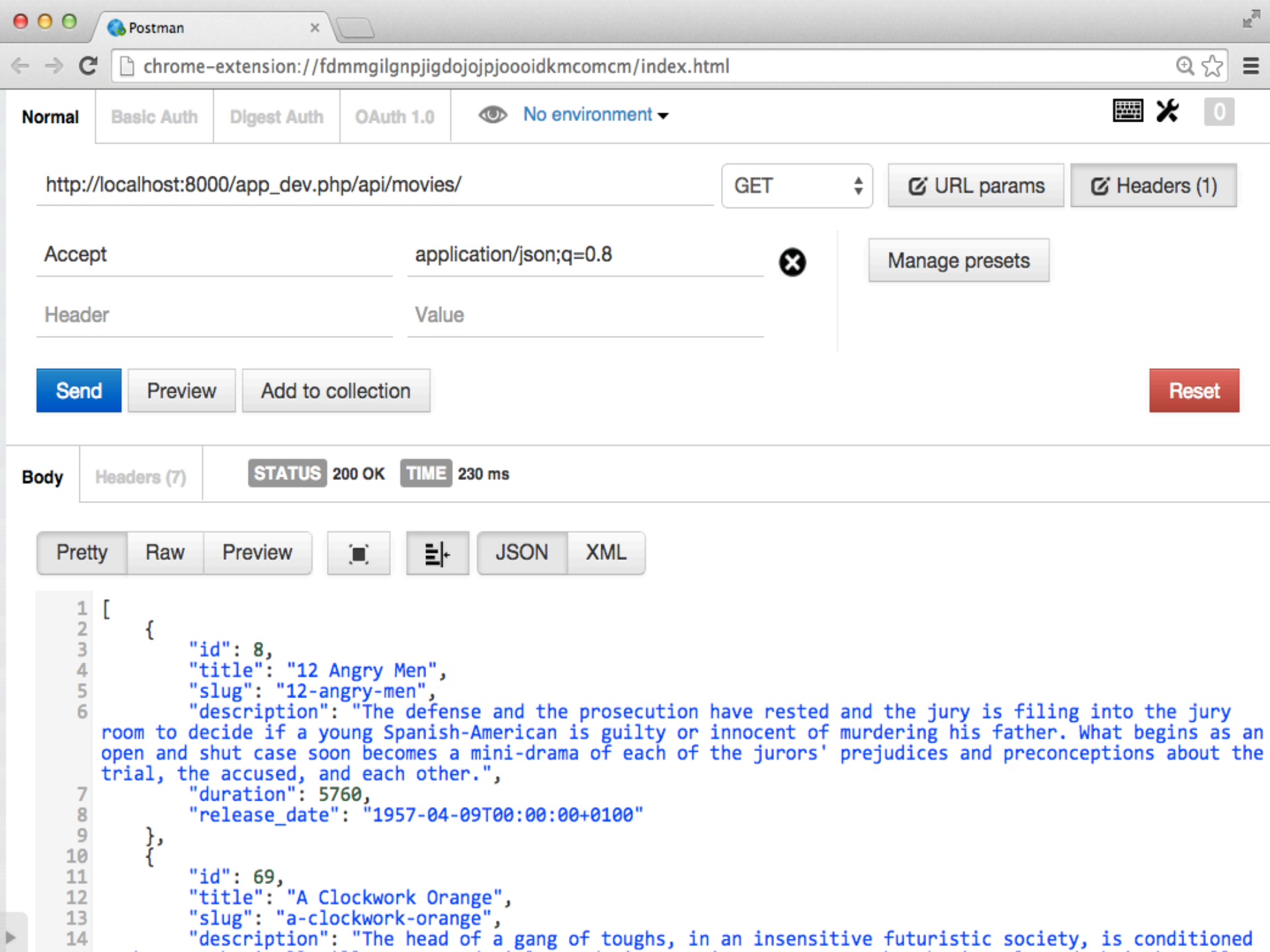
Smoovio > Movies

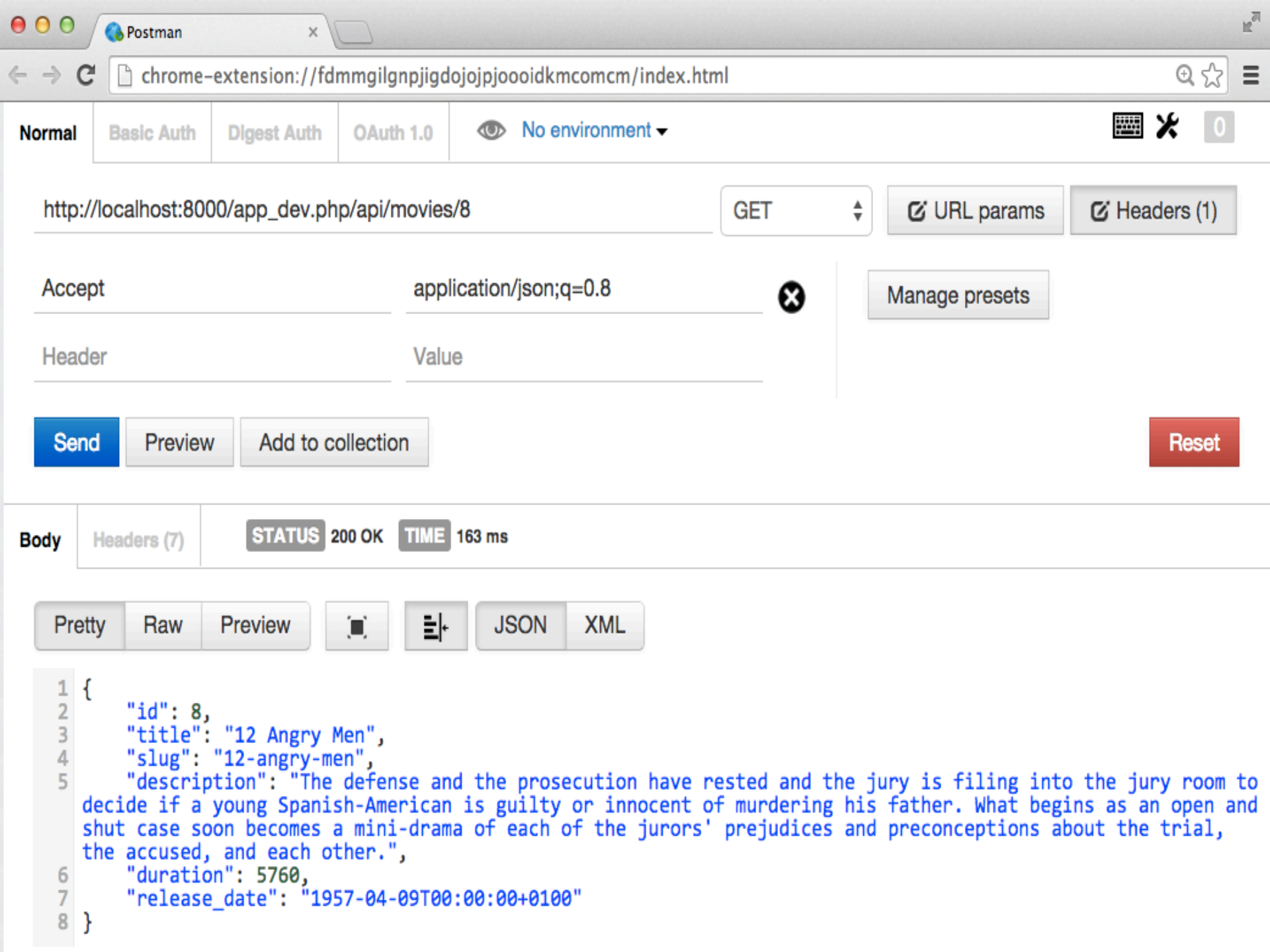
```php
/** @Route("/movies") */
class MovieController extends FOSRestController
{
    /** @Rest\Get("/", name="smoovio_api_movies") */
    public function getMoviesAction()
    {
        return $this->get('smoovio_core.repository.movie')->getMovies();
    }

    /**
     * @Rest\Get(
     *   path = "/{id}",
     *   name="smoovio_api_movie",
     *   requirements={"id"="\d+"}
     * )
     */
    public function getMovieAction(Movie $movie)
    {
        return $movie;
    }
}
```

```php
use JMS\Serializer\Annotation as Serializer;

/**
 * @Serializer\ExclusionPolicy("ALL")
 * @Serializer\XmlRoot("movie")
 */
class Movie
{
    /**
     * @Serializer\Expose
     * @Serializer\XmlAttribute
     */
    private $id;

    /** @Serializer\Expose */
    private $title;

    /** @Serializer\Expose */
    private $slug;

    /** @Serializer\Expose */
    private $description;

    /** @Serializer\Expose */
    private $duration;

    /** @Serializer\Expose */
    private $releaseDate;
}
```


Postman
chrome-extension://fdmmgilgnpjigdojojpjoooidkmcomcm/index.html
Normal
Basic Auth
Digest Auth
OAuth 1.0
No environment
0
http://localhost:8000/app_dev.php/api/movies/
GET
URL params
Headers (1)
Accept
text/xml;q=1, application/json;q=0.8
Manage presets
Header
Value
Send
Preview
Add to collection
Reset
Body
Headers (7)
STATUS 200 OK
TIME 198 ms
Pretty
Raw
Preview
JSON
XML
1 <?xml version="1.0" encoding="UTF-8"?>
2 <result>
3 <entry id="8">
4 <title>
5 <![CDATA[12 Angry Men]]>
6 </title>
7 <slug>
8 <![CDATA[12-angry-men]]>
9 </slug>
10 <description>
11 <![CDATA[The defense and the prosecution have rested and the jury is filing into the jury room to decide if a young Spanish-American is guilty or innocent of murdering his father. What begins as an open and shut case soon becomes a mini-drama of each of the jurors' prejudices and preconceptions about the trial, the accused, and each other.]]>
12 </description>
13 <duration>5760</duration>
14 <release_date>


Postman
chrome-extension://fdmmgilgnpjigdojojpjoooidkmcomcm/index.html
Normal
Basic Auth
Digest Auth
OAuth 1.0
No environment
0
http://localhost:8000/app_dev.php/api/movies/8
GET
URL params
Headers (1)
Accept
text/xml;q=1, application/json;q=0.8
Manage presets
Header
Value
Send
Preview
Add to collection
Reset
Body
Headers (7)
STATUS 200 OK
TIME 172 ms
Pretty
Raw
Preview
JSON
XML
1 <?xml version="1.0" encoding="UTF-8"?>
2 <movie id="8">
3 <title>
4 <![CDATA[12 Angry Men]]>
5 </title>
6 <slug>
7 <![CDATA[12-angry-men]]>
8 </slug>
9 <description>
10 <![CDATA[The defense and the prosecution have rested and the jury is filing into the jury room to decide if a young Spanish-American is guilty or innocent of murdering his father. What begins as an open and shut case soon becomes a mini-drama of each of the jurors' prejudices and preconceptions about the trial, the accused, and each other.]]>
11 </description>
12 <duration>5760</duration>
13 <release_date>
14 <![CDATA[1957-04-09T00:00:00+0100]]>


Postman
chrome-extension://fdmmgilgnpjigdojojpjoooidkmcomcm/index.html
Normal
Basic Auth
Digest Auth
OAuth 1.0
No environment
0
http://localhost:8000/app_dev.php/api/movies/
GET
URL params
Headers (1)
Accept
application/json;q=0.8
Manage presets
Header
Value
Send
Preview
Add to collection
Reset
Body
Headers (7)
STATUS 200 OK
TIME 230 ms
Pretty
Raw
Preview
JSON
XML
[
{
"id": 8,
"title": "12 Angry Men",
"slug": "12-angry-men",
"description": "The defense and the prosecution have rested and the jury is filing into the jury room to decide if a young Spanish-American is guilty or innocent of murdering his father. What begins as an open and shut case soon becomes a mini-drama of each of the jurors' prejudices and preconceptions about the trial, the accused, and each other.",
"duration": 5760,
"release_date": "1957-04-09T00:00:00+0100"
},
{
"id": 69,
"title": "A Clockwork Orange",
"slug": "a-clockwork-orange",
"description": "The head of a gang of toughs, in an insensitive futuristic society, is conditioned


Postman
chrome-extension://fdmmgilgnpjigdojojpjoooidkmcomcm/index.html
Normal
Basic Auth
Digest Auth
OAuth 1.0
No environment
http://localhost:8000/app_dev.php/api/movies/8
GET
URL params
Headers (1)
Accept
application/json;q=0.8
Manage presets
Header
Value
Send
Preview
Add to collection
Reset
Body
Headers (7)
STATUS 200 OK
TIME 163 ms
Pretty
Raw
Preview
JSON
XML
{
"id": 8,
"title": "12 Angry Men",
"slug": "12-angry-men",
"description": "The defense and the prosecution have rested and the jury is filing into the jury room to decide if a young Spanish-American is guilty or innocent of murdering his father. What begins as an open and shut case soon becomes a mini-drama of each of the jurors' prejudices and preconceptions about the trial, the accused, and each other.",
"duration": 5760,
"release_date": "1957-04-09T00:00:00+0100"
}

Smoovio > XML Movies Collection

The XML movies collection view outputs **<results>** & **<result>** tags instead of **<movies>** & **<movie>** tags.

Smoovio > Movies > Custom Collection

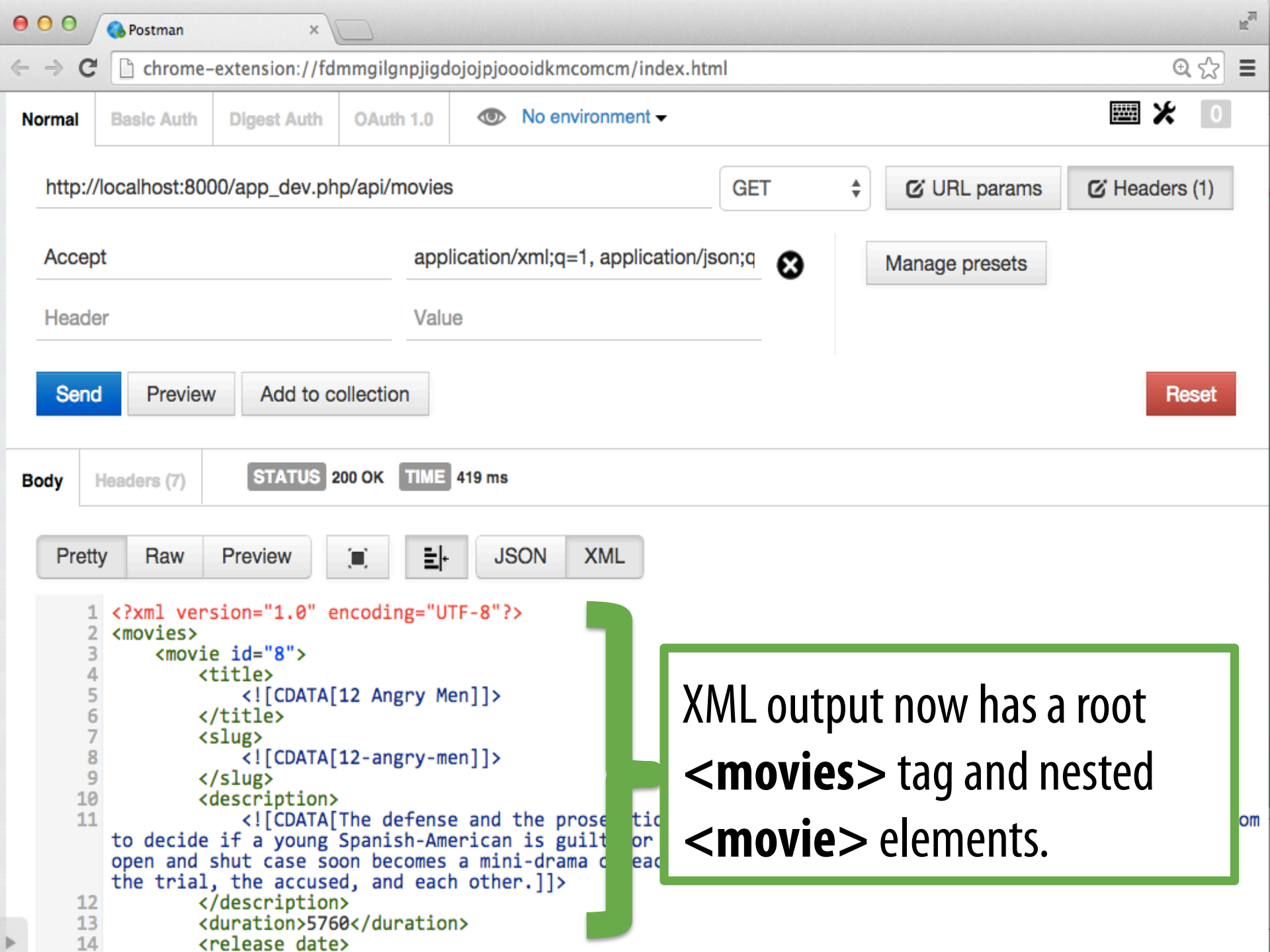
```php
# src/Sensio/Bundle/ApiBundle/Representation/Movies.php
namespace Smoovio\Bundle\ApiBundle\Representation;

class Movies
{
    public $data;

    public function __construct(array $data)
    {
        $this->data = $data;
    }
}
```

Smoovio > Movies > Custom Collection

```php
namespace Smoovio\Bundle\ApiBundle\Representation;

use JMS\Serializer\Annotation as Serializer;

/** @Serializer\XmlRoot("movies") */
class Movies
{
    /**
     * @Serializer\Type("array<Smoovio\Bundle\CoreBundle\Entity\Movie>")
     * @Serializer\XmlList(inline=true, entry = "movie")
     * @Serializer\SerializedName("movies")
     */
    public $data;
}
```

Smoovio > Movies > Custom Collection

```php
// ...
use Smoovio\Bundle\ApiBundle\Representation\Movies;

// ...
class MovieController extends FOSRestController
{
    // ...
    public function getMoviesAction()
    {
        $movies = $this->get('smoovio_core.repository.movie')->getMovies();

        return new Movies($movies);
    }
}
```


Postman
chrome-extension://fdmmgilgnpjigdojojpjoooidkmcomcm/index.html
Normal
Basic Auth
Digest Auth
OAuth 1.0
No environment
http://localhost:8000/app_dev.php/api/movies
GET
URL params
Headers (1)
Accept
application/xml;q=1, application/json;q
Manage presets
Header
Value
Send
Preview
Add to collection
Reset
Body
Headers (7)
STATUS 200 OK
TIME 419 ms
Pretty
Raw
Preview
JSON
XML
1 <?xml version="1.0" encoding="UTF-8"?>
2 <movies>
3 <movie id="8">
4 <title>
5 <![CDATA[12 Angry Men]]>
6 </title>
7 <slug>
8 <![CDATA[12-angry-men]]>
9 </slug>
10 <description>
11 <![CDATA[The defense and the prose tic
to decide if a young Spanish-American is guilt or
open and shut case soon becomes a mini-drama o eac
the trial, the accused, and each other.]]>
12 </description>
13 <duration>5760</duration>
14 <release date>
om
XML output now has a root
<movies> tag and nested
<movie> elements.

User Story #8

As an API consumer, I want to be able to filter and sort the movies list.

Smoovio > Movies > Filtering

Filtering and sorting the collection simply means providing a set of query string parameters to the route.

FOSRestBundle > QueryParam Annotation

- Describes an allowed query string parameter,
- Supports format requirements and default values on query string parameters.

```
/**
 * @QueryParam(
 *    name="",
 *    key=null,
 *    requirements="",
 *    default=null,
 *    description="",
 *    strict=false,
 *    array=false,
 *    nullable=false
 * )
 */
```

FOSRestBundle > RequestParam Annotation

- Describes an allowed request parameter,
- Supports format requirements and default values on request parameters.

```
/**
 * @RequestParam(
 *    name="",
 *    key=null,
 *    requirements="",
 *    default=null,
 *    description="",
 *    strict=false,
 *    array=false,
 *    nullable=false
 * )
 */
```

FOSRestBundle > ParamFetcher

Retrieves and validates a query or request parameter described with an **QueryParam** or **RequestParam** annotation.

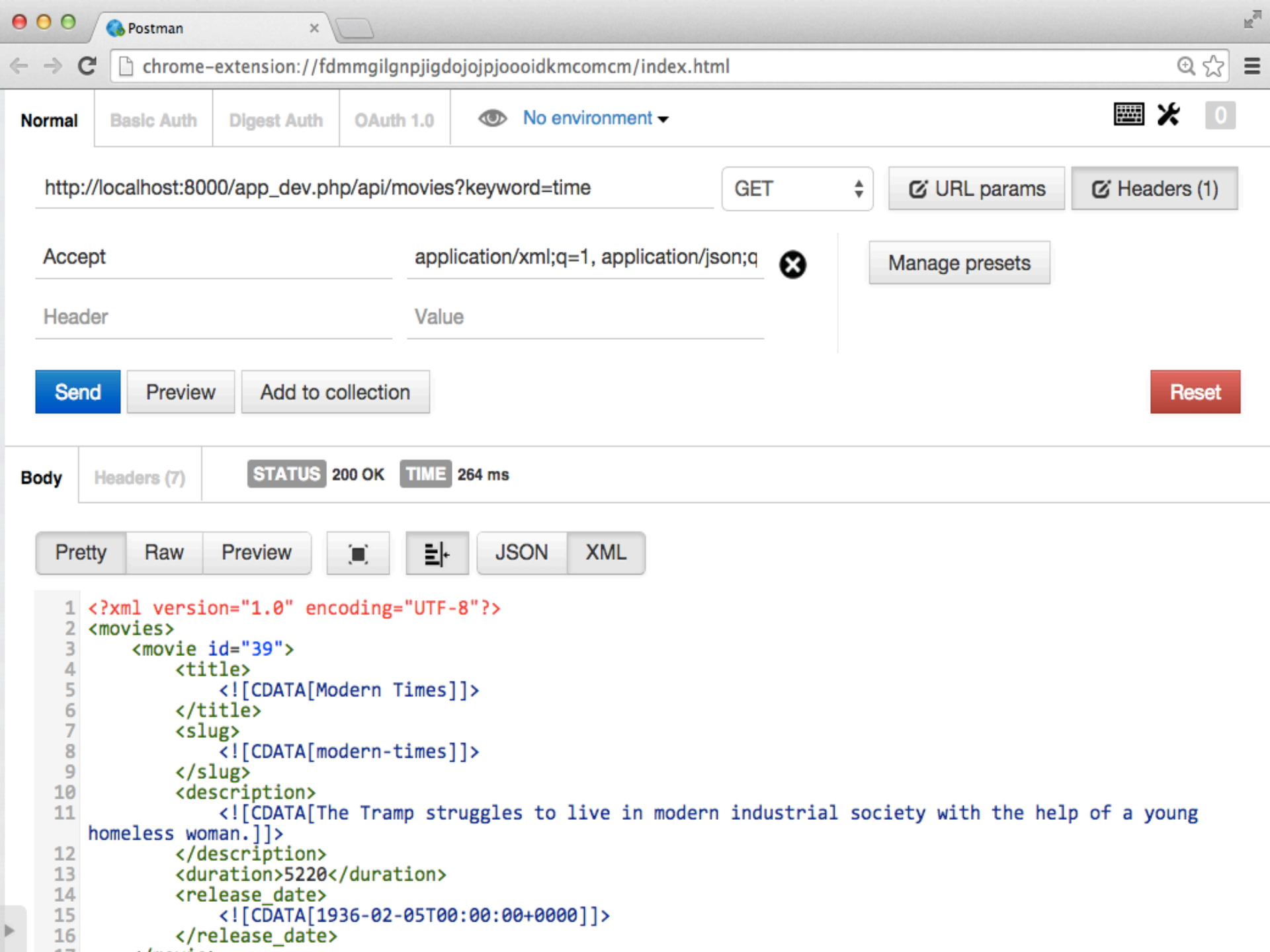
```
public function getMoviesAction(ParamFetcherInterface $paramFetcher)
{
    $keyword = $paramFetcher->get('keyword');
    $order   = $paramFetcher->get('order');

    // …
}
```

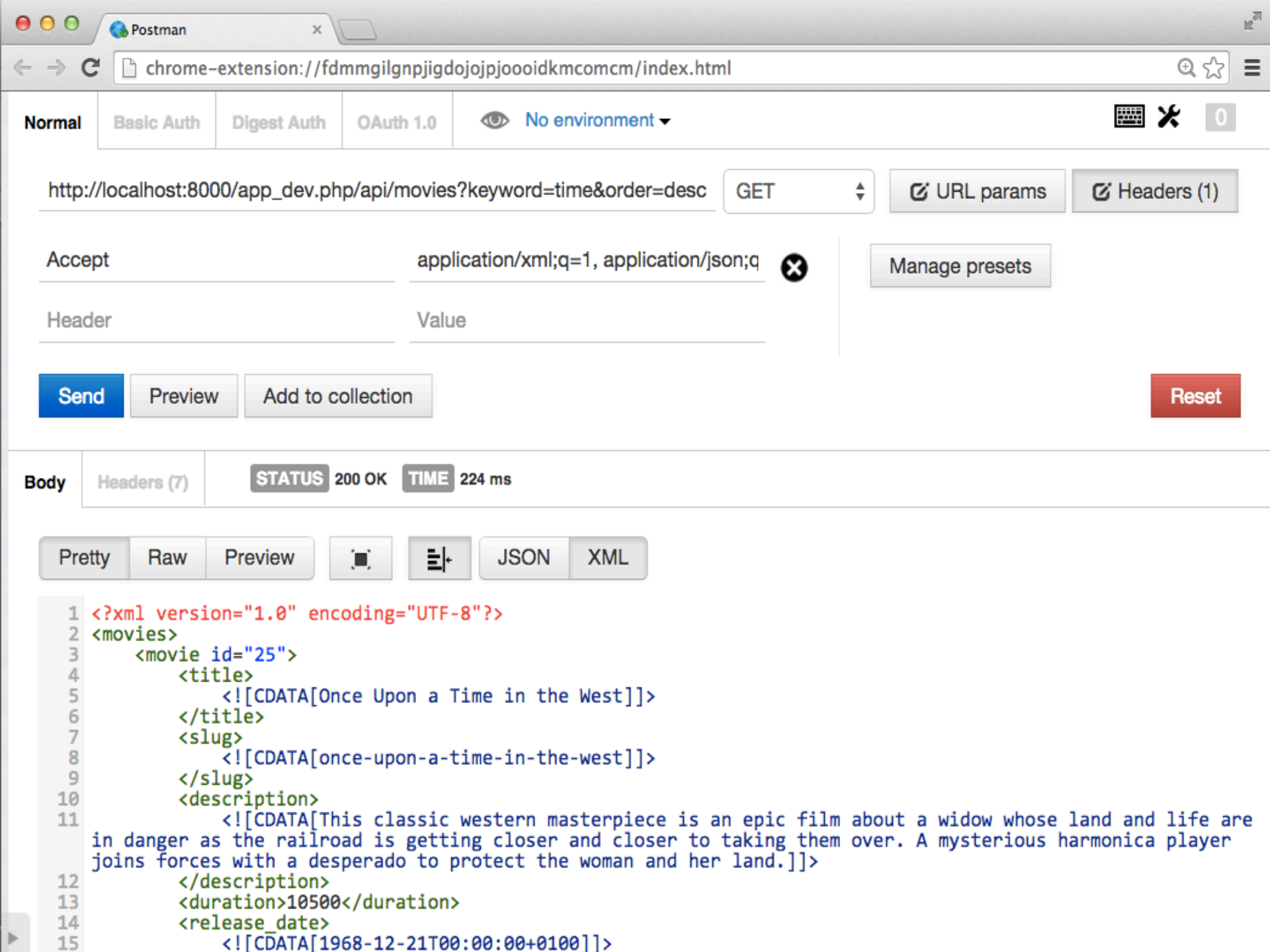
Smoovio > Movies > Filtering

```
class MovieController extends FOSRestController
{
    /**
     * @Rest\Get("/", name="smoovio_api_movies")
     * @Rest\QueryParam(
     *     name="keyword",
     *     requirements="[a-zA-Z0-9]+",
     *     nullable=true,
     *     description="The keyword to search for."
     * )
     * @Rest\QueryParam(
     *     name="order",
     *     requirements="asc|desc",
     *     default="asc",
     *     description="Sort order (asc or desc)."
     * )
     */
    public function getMoviesAction(ParamFetcherInterface $paramFetcher)
```

Smoovio > Movies > Filtering

```php
use FOS\RestBundle\Request\ParamFetcherInterface;

class MovieController extends FOSRestController
{
    // ...
    public function getMoviesAction(ParamFetcherInterface $paramFetcher)
    {
        $repository = $this->get('smoovio_core.repository.movie');

        return new Movies($repository->search(
            $paramFetcher->get('keyword'),
            $paramFetcher->get('order')
        ));
    }
}
```


Postman
chrome-extension://fdmmgilgnpjigdojojpjoooidkmcomcm/index.html
Normal
Basic Auth
Digest Auth
OAuth 1.0
No environment
0
http://localhost:8000/app_dev.php/api/movies?keyword=time
GET
URL params
Headers (1)
Accept
application/xml;q=1, application/json;q
Manage presets
Header
Value
Send
Preview
Add to collection
Reset
Body
Headers (7)
STATUS 200 OK
TIME 264 ms
Pretty
Raw
Preview
JSON
XML
1 <?xml version="1.0" encoding="UTF-8"?>
2 <movies>
3 <movie id="39">
4 <title>
5 <![CDATA[Modern Times]]>
6 </title>
7 <slug>
8 <![CDATA[modern-times]]>
9 </slug>
10 <description>
11 <![CDATA[The Tramp struggles to live in modern industrial society with the help of a young homeless woman.]]>
12 </description>
13 <duration>5220</duration>
14 <release_date>
15 <![CDATA[1936-02-05T00:00:00+0000]]>
16 </release_date>


Postman
chrome-extension://fdmmgilgnpjigdojojpjoooidkmcomcm/index.html
Normal
Basic Auth
Digest Auth
OAuth 1.0
No environment
0
http://localhost:8000/app_dev.php/api/movies?keyword=time&order=desc
GET
URL params
Headers (1)
Accept
application/xml;q=1, application/json;q
Manage presets
Header
Value
Send
Preview
Add to collection
Reset
Body
Headers (7)
STATUS 200 OK
TIME 224 ms
Pretty
Raw
Preview
JSON
XML
1 <?xml version="1.0" encoding="UTF-8"?>
2 <movies>
3 <movie id="25">
4 <title>
5 <![CDATA[Once Upon a Time in the West]]>
6 </title>
7 <slug>
8 <![CDATA[once-upon-a-time-in-the-west]]>
9 </slug>
10 <description>
11 <![CDATA[This classic western masterpiece is an epic film about a widow whose land and life are in danger as the railroad is getting closer and closer to taking them over. A mysterious harmonica player joins forces with a desperado to protect the woman and her land.]]>
12 </description>
13 <duration>10500</duration>
14 <release_date>
15 <![CDATA[1968-12-21T00:00:00+0100]]>

Postman

chrome-extension://fdmmgilgnpjigdojojpjoooidkmcomcm/index.html

Pretty | Raw | Preview | JSON | XML

```xml
<?xml version="1.0" encoding="UTF-8"?>
<movies>
    <movie id="25">
        <title>
            <![CDATA[Once Upon a Time in the West]]>
        </title>
        <slug>
            <![CDATA[once-upon-a-time-in-the-west]]>
        </slug>
        <description>
            <![CDATA[This classic western masterpiece is an epic film about a widow whose land and life are in danger as the railroad is getting closer and closer to taking them over. A mysterious harmonica player joins forces with a desperado to protect the woman and her land.]]>
        </description>
        <duration>10500</duration>
        <release_date>
            <![CDATA[1968-12-21T00:00:00+0100]]>
        </release_date>
    </movie>
    <movie id="76">
        <title>
            <![CDATA[Once Upon a Time in America]]>
        </title>
        <slug>
            <![CDATA[once-upon-a-time-in-america]]>
        </slug>
        <description>
            <![CDATA[Though Sergio Leone is primarily known for his westerns, his final film is a sweeping gangster epic with meditations on friendship, loyalty, and the passage of time. Spanning decades, the film follows a group of Jewish gangsters from childhood into their glory years of prohibition, and their eventual reunion in later years.]]>
        </description>
        <duration>13740</duration>
        <release_date>
            <![CDATA[1984-02-17T00:00:00+0100]]>
        </release_date>
    </movie>
    <movie id="39">
        <title>
            <![CDATA[Modern Times]]>
        </title>
        <slug>
            <![CDATA[modern-times]]>
        </slug>
        <description>
            <![CDATA[The Tramp struggles to live in modern industrial society with
        </description>
        <duration>5220</duration>
        <release_date>
            <![CDATA[1936-02-05T00:00:00+0000]]>
        </release_date>
    </movie>
</movies>
```

Collection has three movies matching the « *time* » word in their titles.

User Story #9

As an API consumer, I want to see movies list paginated.

Smoovio > Movies > Pagination

Paginating a list of records is easy using the famous **Pagerfanta** Composer package.

Smoovio > Movies > Pager Fanta

Pagerfanta provides a simple pager interface and **multiple adapters** to paginate arrays, Doctrine queries, Propel queries...

```php
$adapter = new \Pagerfanta\Adapter\ArrayAdapter($array);
$pager = new \Pagerfanta\Pagerfanta($adapter);

$pager->setMaxPerPage($maxPerPage);
$pager->getMaxPerPage();

$pager->setCurrentPage($currentPage);
$pager->getCurrentPage();

$pager->getNbResults();
$pager->getCurrentPageResults();

$pager->haveToPaginate();

$pager->getNbPages();
$pager->hasPreviousPage();
$pager->getPreviousPage();
$pager->hasNextPage();
$pager->getNextPage();
```

The **MovieRepository** object already leverages the **Pagerfanta** object to paginate the search query builder.

```php
namespace Smoovio\Bundle\CoreBundle\Repository;

use Doctrine\ORM\EntityRepository;
use Doctrine\ORM\QueryBuilder;
use Pagerfanta\Adapter\DoctrineORMAdapter;
use Pagerfanta\Pagerfanta;

abstract class AbstractRepository extends EntityRepository
{
    protected function paginate(QueryBuilder $qb, $limit = 20, $offset = 0)
    {
        if (0 == $limit || 0 == $offset) {
            throw new \LogicException('$limit & $offset must be greater than 0.');
        }

        $pager = new Pagerfanta(new DoctrineORMAdapter($qb));
        $pager->setCurrentPage(ceil(($offset + 1) / $limit));
        $pager->setMaxPerPage((int) $limit);

        return $pager;
    }
}
```

```php
namespace Smoovio\Bundle\CoreBundle\Repository;

class MovieRepository extends AbstractRepository
{
    public function search($term, $order = 'asc', $limit = 20, $offset = 0)
    {
        $qb = $this
            ->createQueryBuilder('m')
            ->select('m')
            ->orderBy('m.title', $order);

        if ($term) {
            $qb
                ->where('m.title LIKE ?1')
                ->setParameter(1, '%' . $term . '%');
        }

        return $this->paginate($qb, $limit, $offset);
    }
}
```

Smoovio > Movies > Pagination

Paginating the list of Movies is as simple as introducing two new **offset** & **limit** query string parameters.

```php
class MovieController extends FOSRestController
{
    /**
     * //...
     * @Rest\QueryParam(
     *     name="limit",
     *     requirements="\d+",
     *     default="15",
     *     description="Max number of movies per page."
     * )
     * @Rest\QueryParam(
     *     name="offset",
     *     requirements="\d+",
     *     default="0",
     *     description="The pagination offset."
     * )
     */
    public function getMoviesAction(ParamFetcherInterface $paramFetcher)
    {
    }
}
```

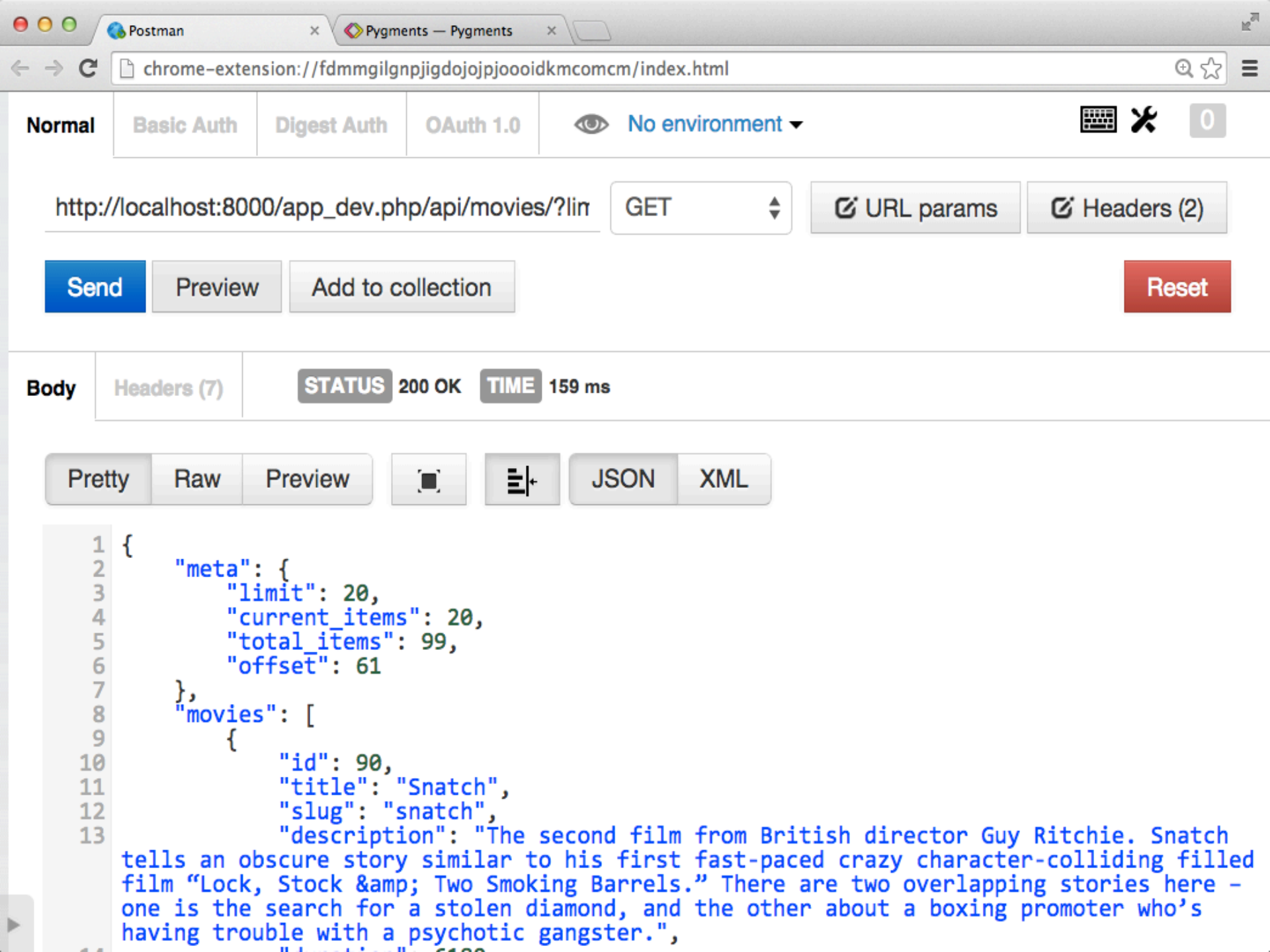
```php
class MovieController extends FOSRestController
{
    // ...
    function getMoviesAction(ParamFetcherInterface $paramFetcher)
    {
        $movies = $this->get('smoovio_core.repository.movie')->search(
            $paramFetcher->get('keyword'),
            $paramFetcher->get('order'),
            $paramFetcher->get('limit'),
            $paramFetcher->get('offset')
        );

        return new Movies($movies);
    }
}
```

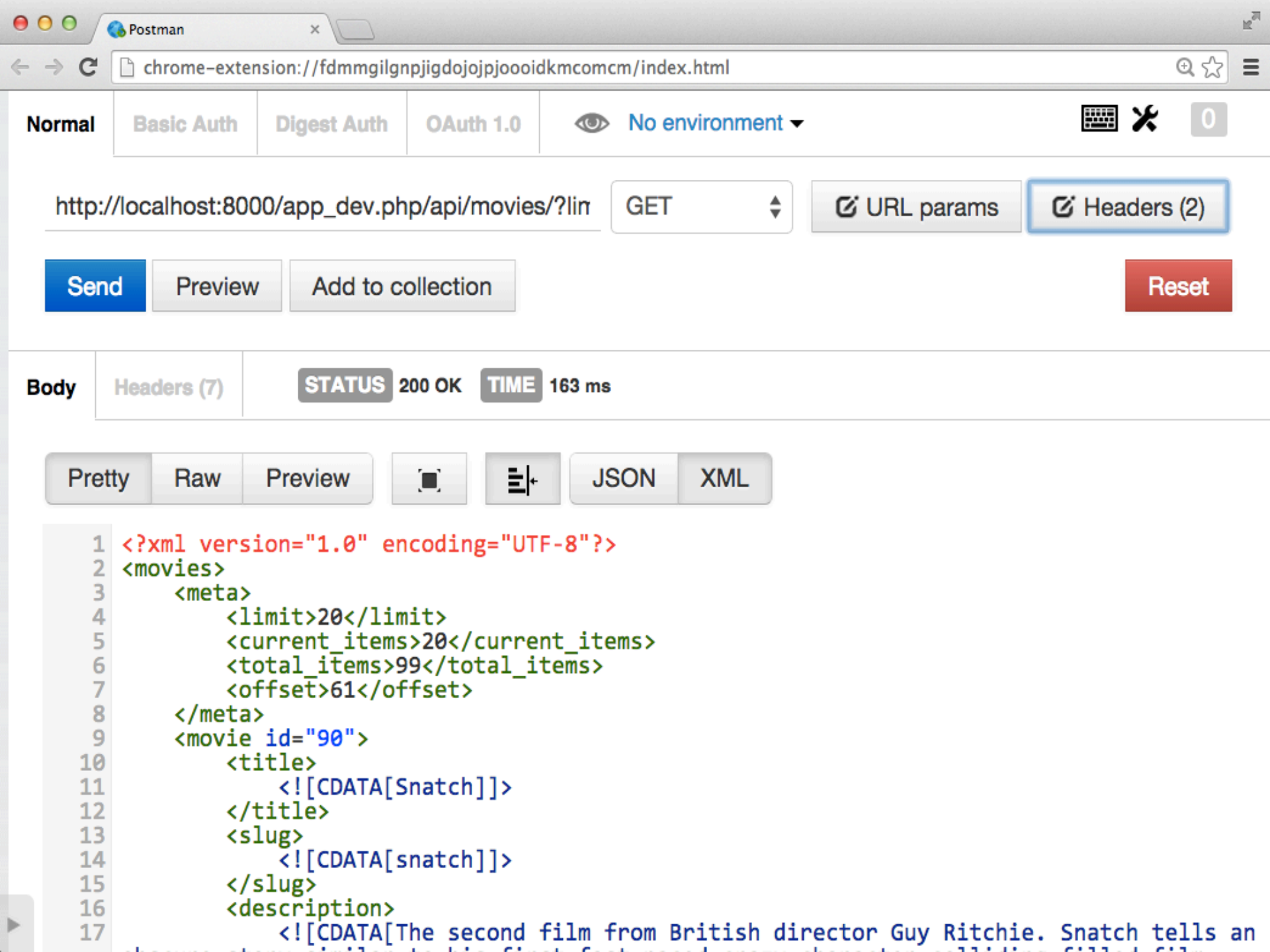
Smoovio > Movies > Pagination Metadata

The **Pagerfanta** object encapsulates all pagination information. To help browsing the movies list, these information can be embedded in the serialized output.

Smoovio > Movies > Pagination Metadata

```php
// ...
use Pagerfanta\Pagerfanta;

class Movies
{
    // ...
    public function __construct(Pagerfanta $data)
    {
        $this->data = $data;

        $this->addMeta('limit', $data->getMaxPerPage());
        $this->addMeta('current_items', count($data->getCurrentPageResults()));
        $this->addMeta('total_items', $data->getNbResults());
        $this->addMeta('offset', $data->getCurrentPageOffsetStart());
    }
}
```


Postman
Pygments — Pygments
chrome-extension://fdmmgilgnpjigdojojpjoooidkmcomcm/index.html
Normal
Basic Auth
Digest Auth
OAuth 1.0
No environment
0
http://localhost:8000/app_dev.php/api/movies/?lin
GET
URL params
Headers (2)
Send
Preview
Add to collection
Reset
Body
Headers (7)
STATUS 200 OK
TIME 159 ms
Pretty
Raw
Preview
JSON
XML
{
"meta": {
"limit": 20,
"current_items": 20,
"total_items": 99,
"offset": 61
},
"movies": [
{
"id": 90,
"title": "Snatch",
"slug": "snatch",
"description": "The second film from British director Guy Ritchie. Snatch tells an obscure story similar to his first fast-paced crazy character-colliding filled film “Lock, Stock & Two Smoking Barrels.” There are two overlapping stories here – one is the search for a stolen diamond, and the other about a boxing promoter who’s having trouble with a psychotic gangster.",


Postman
chrome-extension://fdmmgilgnpjigdojojpjoooidkmcomcm/index.html
Normal
Basic Auth
Digest Auth
OAuth 1.0
No environment
0
http://localhost:8000/app_dev.php/api/movies/?lin
GET
URL params
Headers (2)
Send
Preview
Add to collection
Reset
Body
Headers (7)
STATUS 200 OK
TIME 163 ms
Pretty
Raw
Preview
JSON
XML
1 <?xml version="1.0" encoding="UTF-8"?>
2 <movies>
3 <meta>
4 <limit>20</limit>
5 <current_items>20</current_items>
6 <total_items>99</total_items>
7 <offset>61</offset>
8 </meta>
9 <movie id="90">
10 <title>
11 <![CDATA[Snatch]]>
12 </title>
13 <slug>
14 <![CDATA[snatch]]>
15 </slug>
16 <description>
17 <![CDATA[The second film from British director Guy Ritchie. Snatch tells an

Smoovio > Movies > Pagination Links

To help a client browsing the API, it's also a good practice to add navigation links alongside with other pagination data.

Smoovio > Movies > Pagination Links

The response should include the total number of items and the links to browse the other paginated items.

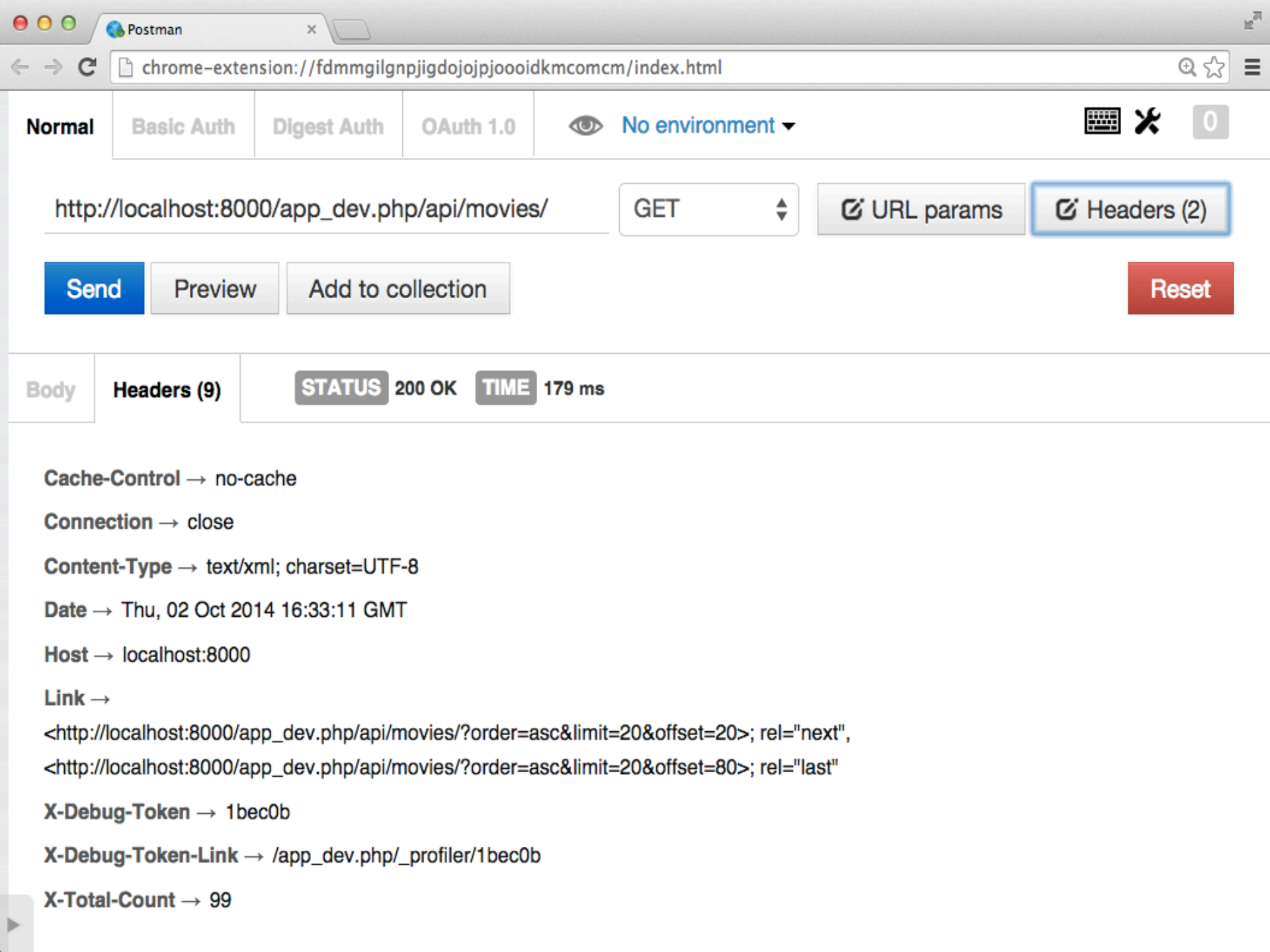
```
HTTP/1.1 200 OK
Content-Type: application/json
Host: localhost:8000
X-Total-Count: 99
Link: <http://…/movies?limit=20&offset=0>; rel="first",
<http://…/movies?limit=20&offset=21>; rel="prev",
<http://…/movies?limit=20&offset=61>; rel="next",
<http://…/movies?limit=20&offset=81>; rel="last"
```

Smoovio > Movies > Pagination Links

The **MoviesViewHandler** service renders the representation of the movies list and appends the **X-Total-Count** and **Link** HTTP headers to the response.

Smoovio > Movies > Pagination Links

```
class MovieController extends FOSRestController
{
    public function getMoviesAction(ParamFetcherInterface $paramFetcher)
    {
        $movies = $this->get('smoovio_core.repository.movie')->search(
            $paramFetcher->get('keyword'),
            $paramFetcher->get('order'),
            $paramFetcher->get('limit'),
            $paramFetcher->get('offset')
        );

        return $this
            ->get('smoovio_api.movies_view_handler')
            ->handleRepresentation(new Movies($movies), $paramFetcher->all())
        ;
    }
}
```


Postman
chrome-extension://fdmmgilgnpjigdojojpjoooidkmcomcm/index.html
Normal
Basic Auth
Digest Auth
OAuth 1.0
No environment
0
http://localhost:8000/app_dev.php/api/movies/
GET
URL params
Headers (2)
Send
Preview
Add to collection
Reset
Body
Headers (9)
STATUS 200 OK
TIME 179 ms
Cache-Control → no-cache
Connection → close
Content-Type → text/xml; charset=UTF-8
Date → Thu, 02 Oct 2014 16:33:11 GMT
Host → localhost:8000
Link →
<http://localhost:8000/app_dev.php/api/movies/?order=asc&limit=20&offset=20>; rel="next",
<http://localhost:8000/app_dev.php/api/movies/?order=asc&limit=20&offset=80>; rel="last"
X-Debug-Token → 1bec0b
X-Debug-Token-Link → /app_dev.php/_profiler/1bec0b
X-Total-Count → 99

User Story #10

As an API provider, I want to validate input movies genres data.

Smoovio > Genre > Data Validation

To ensure client's submitted data are not corrupted or dangerous for the application, they must be validated.

Smoovio > Genre > Data Validation

The easiest way is to use the Form and Validator components to map request data to an object and validate it.

Smoovio > Genre > Data Validation

The other way is to use FOSRestBundle and its built-in listeners to hydrate an object and validate it on the fly.

Smoovio > FOSRestBundle > Enable validation

```
fos_rest:
    # ...
    body_converter:
        enabled: true
        validate: true
        validation_errors_argument: violations
```

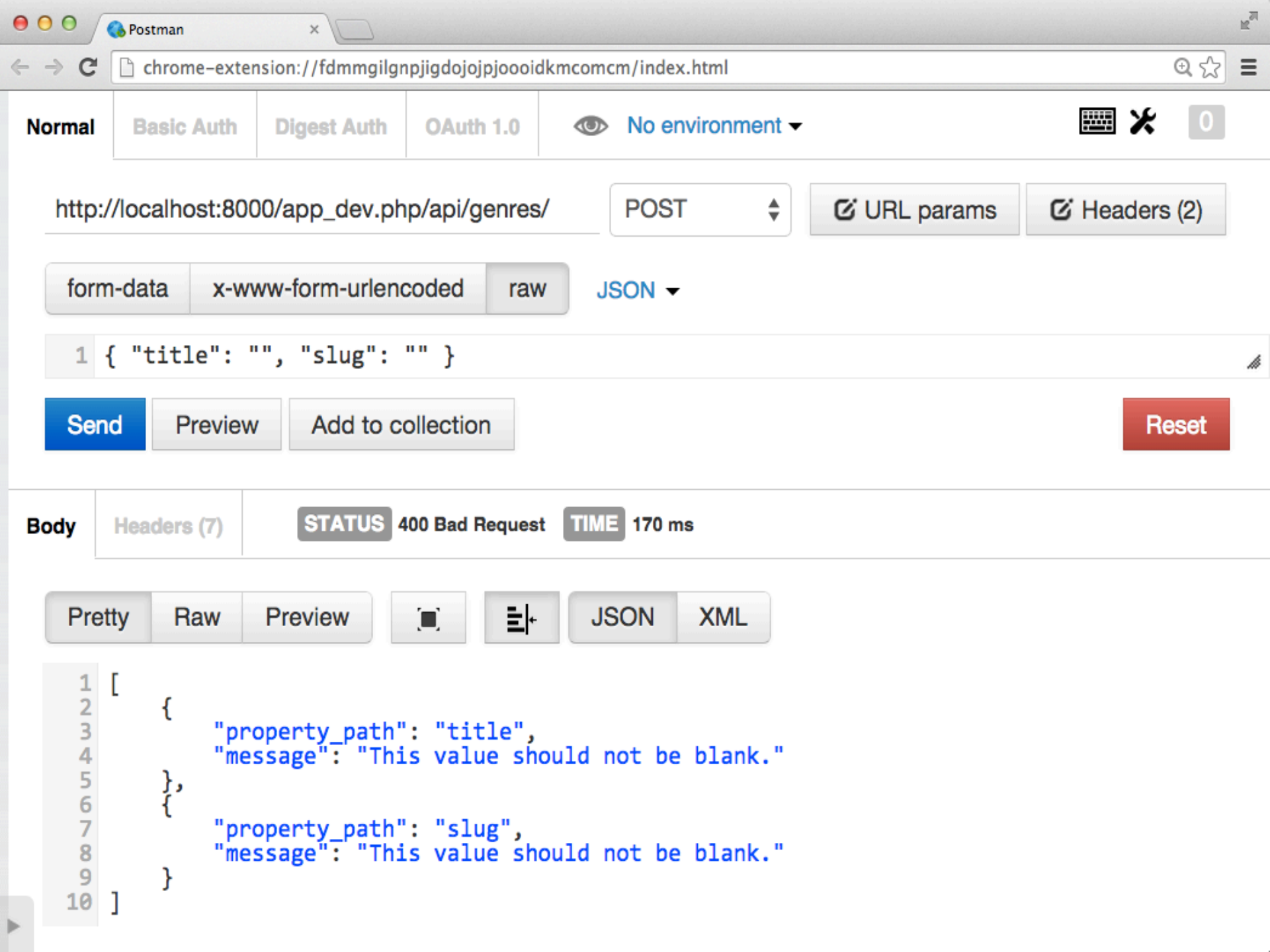
The « **validation_errors_arguments** » option defines the received action's parameter that contains validation errors.

Smoovio > FOSRestBundle > Param Converter

```
/**
 * @Rest\Post("/", name="smoovio_api_new_genre »)
 * @Rest\View(statusCode=201)
 * @ParamConverter("genre", converter="fos_rest.request_body")
 */
public function postGenreAction(
    Genre $genre,
    ConstraintViolationListInterface $violations
)
{
    if (count($violations)) {
        return $this->view($violations, 400);
    }
    // ...
}
```

Smoovio > FOSRestBundle > Param Converter

The built-in « **fos_rest.request_body** » param converter converts the request JSON/XML body to a populated object, and validates it. Validation errors are passed to the action in the « **violations** » argument.


Postman
chrome-extension://fdmmgilgnpjigdojojpjoooidkmcomcm/index.html
Normal
Basic Auth
Digest Auth
OAuth 1.0
No environment
0
http://localhost:8000/app_dev.php/api/genres/
POST
URL params
Headers (2)
form-data
x-www-form-urlencoded
raw
JSON
1 { "title": "", "slug": "" }
Send
Preview
Add to collection
Reset
Body
Headers (7)
STATUS 400 Bad Request
TIME 170 ms
Pretty
Raw
Preview
JSON
XML
1 [
2 {
3 "property_path": "title",
4 "message": "This value should not be blank."
5 },
6 {
7 "property_path": "slug",
8 "message": "This value should not be blank."
9 }
10]

```php
// ...
use Symfony\Component\Validator\Constraints as Assert;

class Genre
{
    // ...
    /**
     * ...
     * @Assert\NotBlank
     * @Assert\Length(min=2, max=100)
     * @Assert\Regex("/^[a-z]+$/")
     */
    private $title;

    /**
     * ...
     * @Assert\NotBlank
     */
    private $slug;
}
```

Smoovio > Genre > Refactoring

```php
class GenreController extends FOSRestController
{
    /**
     * @Rest\Post("/", name="smoovio_api_new_genre")
     * @ParamConverter("genre", converter="fos_rest.request_body")
     * @Rest\View(statusCode=201)
     */
    function postGenreAction(Genre $genre, ConstraintViolationListInterface $violations)
    {
        if (count($violations)) {
            return $this->view($violations, 400);
        }

        $em = $this->getDoctrine()->getManager();
        $em->persist($genre);
        $em->flush();
    }
}
```

```php
class GenreController extends FOSRestController
{
    /**
     * ...
     * @ParamConverter("apiGenre", converter="fos_rest.request_body")
     */
    function putGenreAction(
        Genre $genre,
        Genre $apiGenre,
        ConstraintViolationListInterface $violations
    )
    {
        if (count($violations)) {
            return $this->view($violations, 400);
        }
        $genre->update($apiGenre);
        $em = $this->getDoctrine()->getManager();
        $em->flush($genre);
    }
}
```

```php
class GenreController extends FOSRestController
{
    /**
     * ...
     * @Rest\View(statusCode=204)
     */
    public function deleteGenreAction(Genre $genre)
    {
        $em = $this->getDoctrine()->getManager();
        $em->remove($genre);
        $em->flush();
    }
}
```

Smoovio > Genres > Validation Groups

Validation groups are also supported by **FOSRestBundle**. To validate a deserialized object against one or more **validation groups**, they must be given in the @**ParamConverter** annotation.

Smoovio > Genres > Validation Groups

```
/**
 * @ParamConverter(
 *      name"genre",
 *      converter="fos_rest.request_body",
 *      options={
 *          "validator"={
 *              "groups"="Create"
 *          }
 *      }
 * )
 */
```

User Story #11

As an API provider, I want to version my API.

Smoovio > API Versioning > Strategies

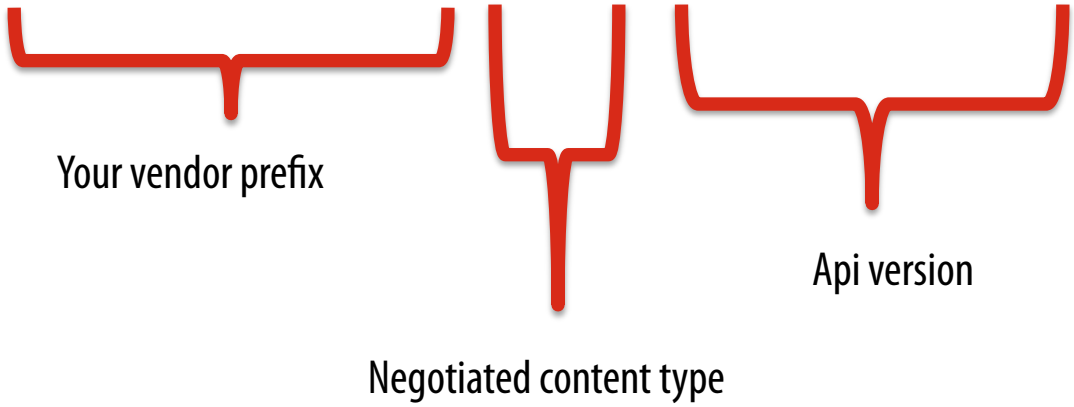
Strategy	Example
Domain Name	`http://v2.domain.tld/api`
Prefix Path	`http://domain.tld/api/v2`
Query String	`http://domain.tld/api?v=v2`
Custom Header	`X-API-Version: 2.0.0`
Accept Header	`Accept: application/vnd.smoovio.movies-v2+json`

Cool URIs don't change!

Keep Consumers Backward Compatibility

Smoovio > API Versioning > Accept Header

```
application/vnd.smoovio.movies+json; version=2.0.0
```



Smoovio > JMSSerializerBundle > Version

The **JMSSerializerBundl**e allows to serialize data only if a required version number is configured and met.

```
@Serializer\Since("2.0.0")
@Serializer\Until("1.9.99")
```

Smoovio > FOSRestBundle > Version

```
fos_rest:
    view:
        mime_types:
            json:
                - application/json
                - application/x-json
                - application/vnd.smoovio.movies+json
                - application/vnd.smoovio.movies+json;v=1.0
                - application/vnd.smoovio.movies+json;v=2.0
            xml:
                - text/xml
                - application/vnd.smoovio.movies+xml
                - application/vnd.smoovio.movies+xml;v=1.0
                - application/vnd.smoovio.movies+xml;v=2.0
```

Smoovio > Movies > Serialization

```
// ...
class Movie
{
    // ...

    /**
     * The date the movie has been released.
     *
     * @var \DateTime
     * @Serializer\Expose
     * @Serializer\Since("2.0")
     */
    private $releaseDate;
}
```

Accept: application/vnd.smoovio.movies+xml;v=1.0

```
<movies>
    <meta>
        <limit>20</limit>
        <current_items>20</current_items>
        <total_items>99</total_items>
        <offset>1</offset>
        <next_page_link>
            <![CDATA[http://localhost:8000/app_dev.php/api/movies/?order=asc&limit=20&offset=20]]>
        </next_page_link>
        <last_page_link>
            <![CDATA[http://localhost:8000/app_dev.php/api/movies/?order=asc&limit=20&offset=80]]>
        </last_page_link>
    </meta>
    <movie id="8">
        <title>
            <![CDATA[12 Angry Men]]>
        </title>
        <slug>
            <![CDATA[12-angry-men]]>
        </slug>
        <description>
            <![CDATA[The defense and the prosecution have rested and the jury is filing into the jury room to decide if a young Spanish-American is guilty or innocent of murdering his father. What begins as an open and shut case soon becomes a mini-drama of each of the jurors' prejudices and preconceptions about the trial, the accused, and each other.]]>
        </description>
        <duration>5760</duration>
    </movie>
    <movie id="69">
```

Accept: application/vnd.smoovio.movies+xml;v=2.0

```
    <meta>
        <limit>20</limit>
        <current_items>20</current_items>
        <total_items>99</total_items>
        <offset>1</offset>
        <next_page_link>
            <![CDATA[http://localhost:8000/app_dev.php/api/movies/?order=asc&limit=20&offset=20]]>
        </next_page_link>
        <last_page_link>
            <![CDATA[http://localhost:8000/app_dev.php/api/movies/?order=asc&limit=20&offset=80]]>
        </last_page_link>
    </meta>
    <movie id="8">
        <title>
            <![CDATA[12 Angry Men]]>
        </title>
        <slug>
            <![CDATA[12-angry-men]]>
        </slug>
        <description>
            <![CDATA[The defense and the prosecution have rested and the jury is filing into the jury room to decide if a young Spanish-American is guilty or innocent of murdering his father. What begins as an open and shut case soon becomes a mini-drama of each of the jurors' prejudices and preconceptions about the trial, the accused, and each other.]]>
        </description>
        <duration>5760</duration>
        <release_date>
            <![CDATA[1957-04-09T00:00:00+0100]]>
        </release_date>
    </movie>
```

User Story #12

As an API provider, I want to document my API.

Why NelmioApiDocBundle?

NelmioApiDocBundle

The **NelmioApiDocBundle** bundle allows you to generate a decent documentation for your APIs.

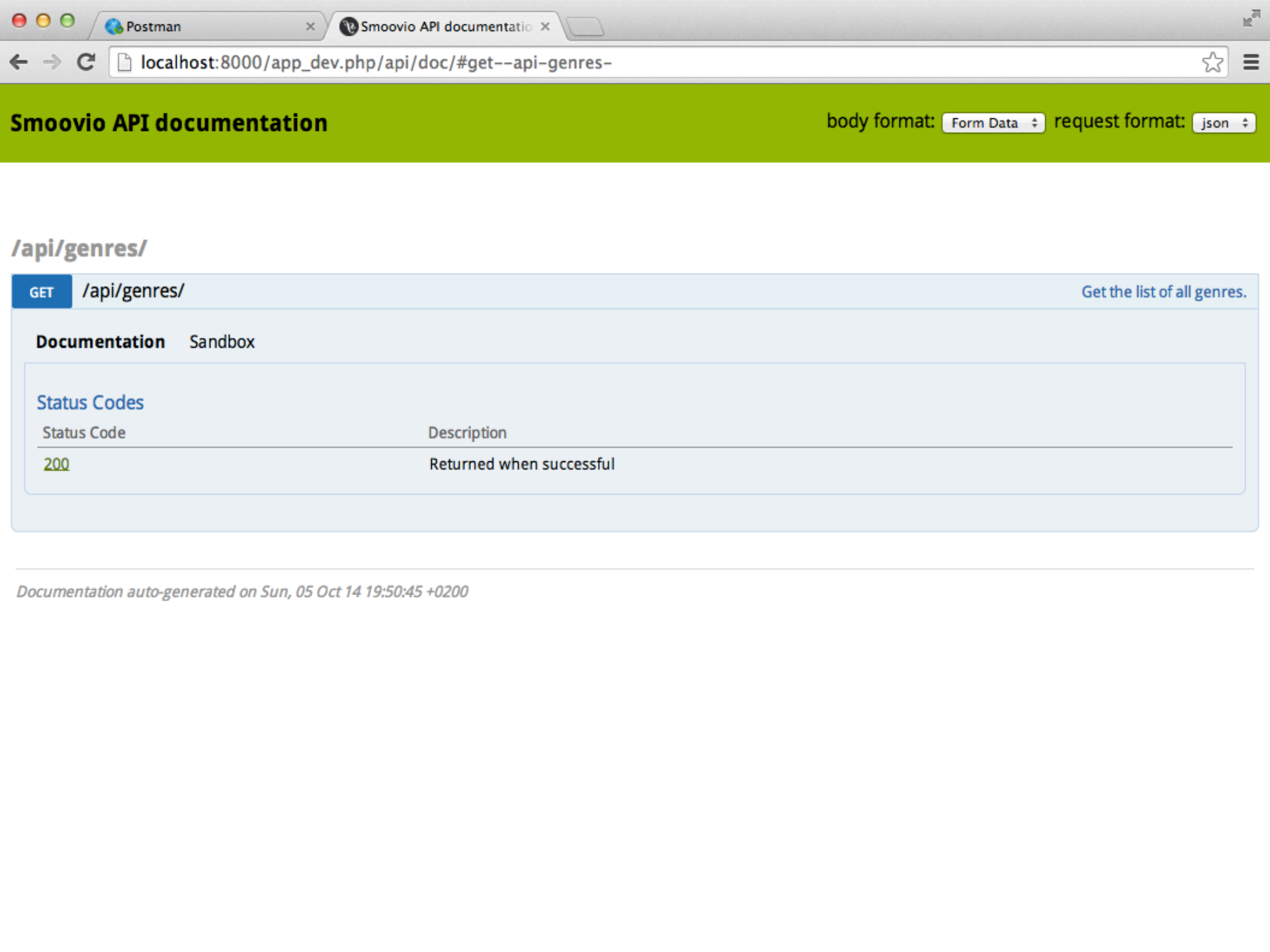
NelmioApiDocBundle

The **NelmioApiDocBundle** bundle also guesses documentation based on **FOSRestBundle** and **JMSSerializerBundle** annotations.

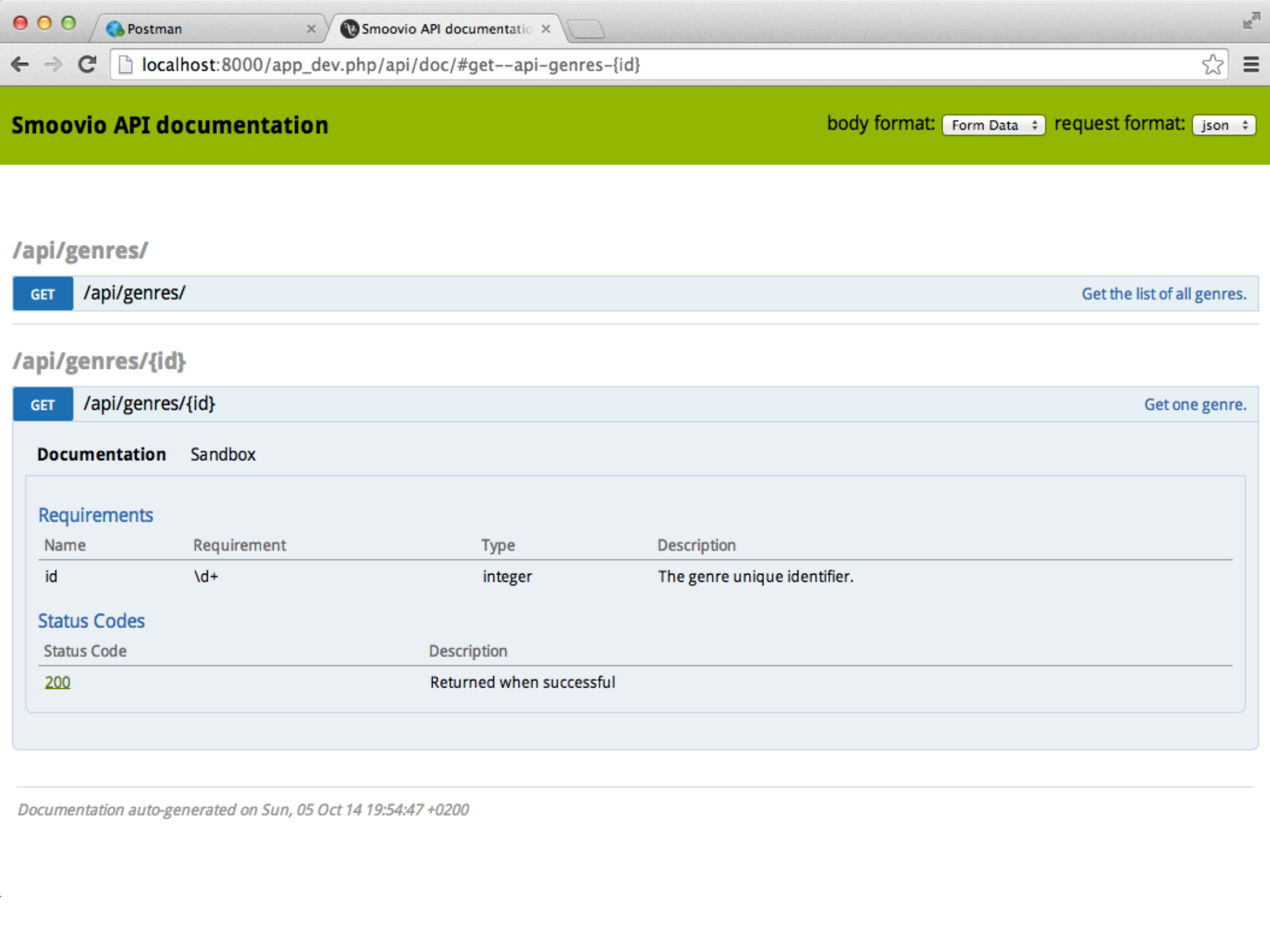
The **ApiDoc** annotation provides an easy way to document your controllers and automatically generate the corresponding documentation.

Smoov.io > NelmioApiDocBundle

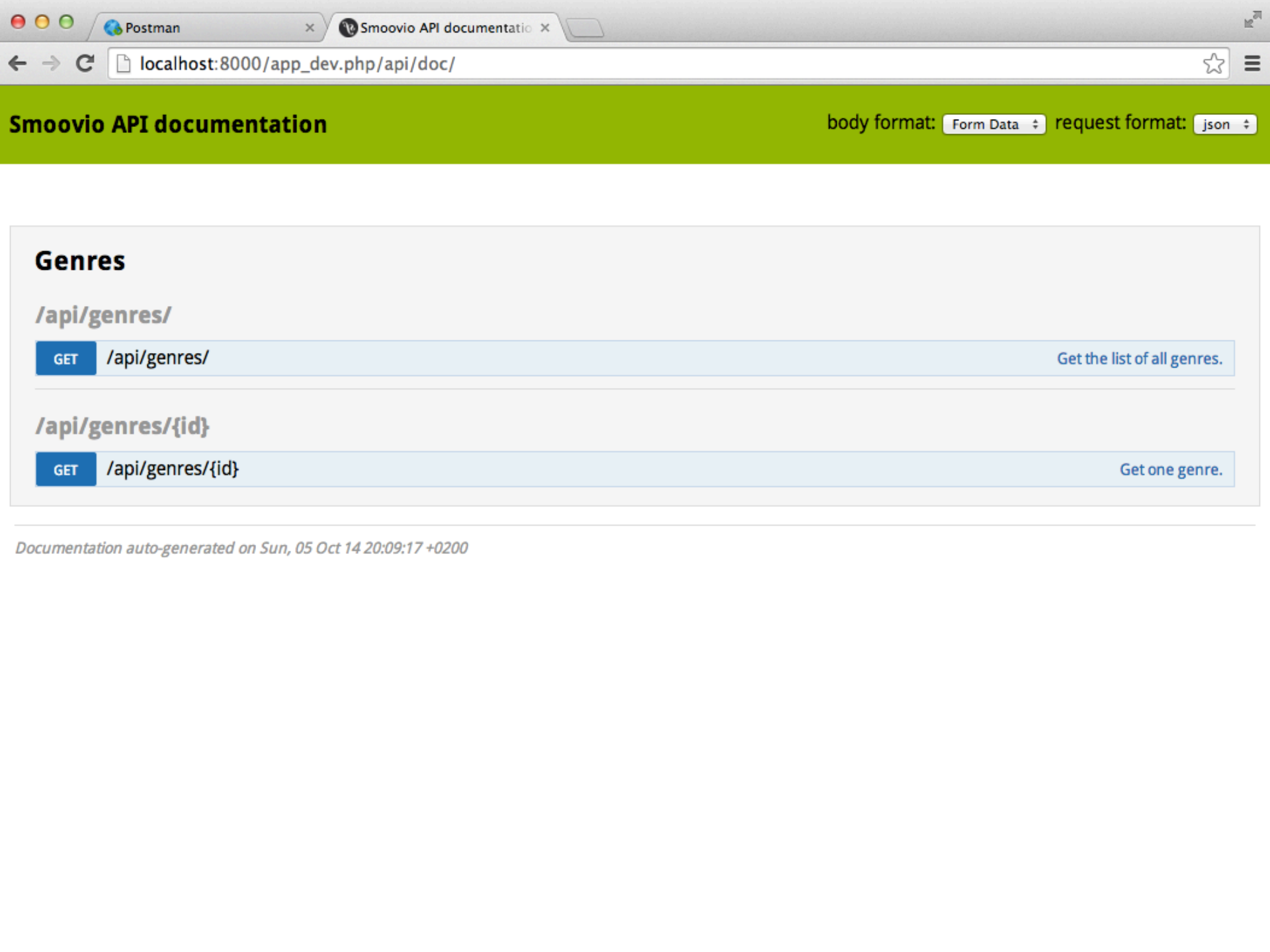
```php
use Nelmio\ApiDocBundle\Annotation as Doc;
// ...
class GenreController extends FOSRestController
{
    /**
     * ...
     * @Doc\ApiDoc(
     *     resource=true,
     *     description="Get the list of all genres.",
     *     statusCodes={
     *         200="Returned when successful",
     *     }
     * )
     */
    public function getGenresAction()
    {
        // ...
    }
}
```


Postman
Smoovio API documentatio
localhost:8000/app_dev.php/api/doc/#get--api-genres-
Smoovio API documentation
body format: Form Data
request format: json
/api/genres/
GET
/api/genres/
Get the list of all genres.
Documentation
Sandbox
Status Codes
Status Code
Description
200
Returned when successful
Documentation auto-generated on Sun, 05 Oct 14 19:50:45 +0200

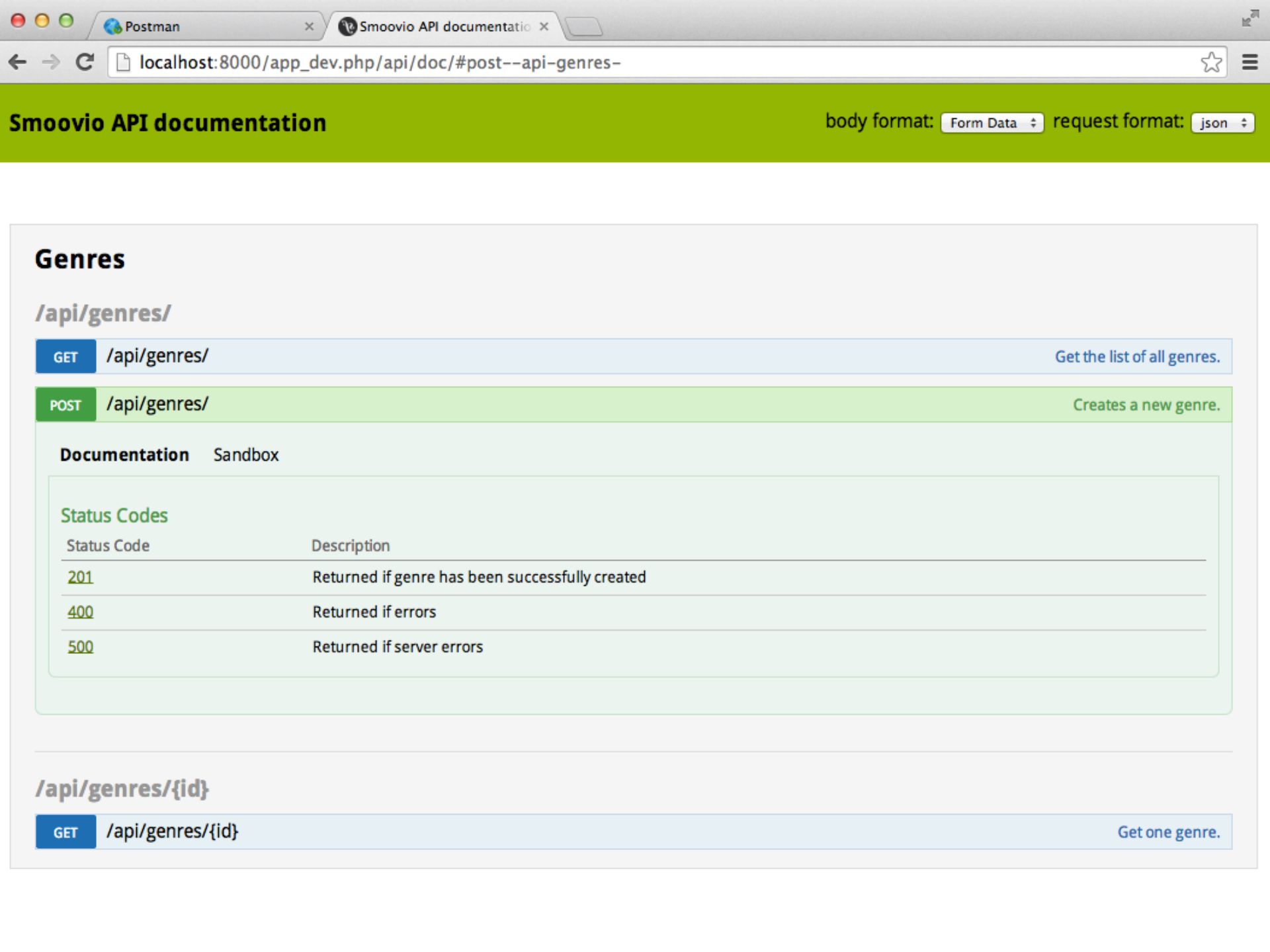
```php
/**
 * ...
 * @Doc\ApiDoc(
 *     resource=true,
 *     description="Get one genre.",
 *     requirements={
 *         {
 *             "name"="id",
 *             "dataType"="integer",
 *             "requirement"="\d+",
 *             "description"="The genre unique identifier."
 *         }
 *     },
 *     statusCodes={
 *         200="Returned when successful",
 *     }
 * )
 */
public function getGenreAction(Genre $genre)
{
    return $genre;
}
```


Postman
Smoovio API documentatio
localhost:8000/app_dev.php/api/doc/#get--api-genres-{id}
Smoovio API documentation
body format: Form Data
request format: json
/api/genres/
GET
/api/genres/
Get the list of all genres.
/api/genres/{id}
GET
/api/genres/{id}
Get one genre.
Documentation
Sandbox
Requirements
Name
Requirement
Type
Description
id
\d+
integer
The genre unique identifier.
Status Codes
Status Code
Description
200
Returned when successful
Documentation auto-generated on Sun, 05 Oct 14 19:54:47 +0200

```
/**
 * ...
 * @Doc\ApiDoc(
 *     section="Genres",
 *     ...
 * )
 */
```


Postman
Smoovio API documentatio
localhost:8000/app_dev.php/api/doc/
Smoovio API documentation
body format: Form Data
request format: json
Genres
/api/genres/
GET
/api/genres/
Get the list of all genres.
/api/genres/{id}
GET
/api/genres/{id}
Get one genre.
Documentation auto-generated on Sun, 05 Oct 14 20:09:17 +0200

```
/**
 * ...
 * @Doc\ApiDoc(
 *     section="Genres",
 *     description="Creates a new genre.",
 *     statusCodes={
 *         201="Returned if genre has been successfully created",
 *         400="Returned if errors",
 *         500="Returned if server errors"
 *     }
 * )
 */
public function postGenreAction(...)
{
}
```


Postman
Smoovio API documentatio
localhost:8000/app_dev.php/api/doc/#post--api-genres-
Smoovio API documentation
body format: Form Data
request format: json
Genres
/api/genres/
GET /api/genres/ Get the list of all genres.
POST /api/genres/ Creates a new genre.
Documentation Sandbox
Status Codes
Status Code Description
201 Returned if genre has been successfully created
400 Returned if errors
500 Returned if server errors
/api/genres/{id}
GET /api/genres/{id} Get one genre.

```
/**
 * ...
 * @Doc\ApiDoc(
 *     section="Genres",
 *     description="Edit an existing genre.",
 *     statusCodes={
 *         200="Returned if genre has been successfully edited",
 *         400="Returned if errors",
 *         500="Returned if server errors"
 *     },
 *     requirements={
 *         {
 *             "name"="id",
 *             "dataType"="integer",
 *             "requirement"="\d+",
 *             "description"="The genre unique identifier."
 *         }
 *     }
 * )
 */
public function putGenreAction(...)
```

/api/genres/

GET /api/genres/ — Get the list of all genres.

POST /api/genres/ — Creates a new genre.

/api/genres/{id}

GET /api/genres/{id} — Get one genre.

PUT /api/genres/{id} — Edit an existing genre.

Documentation Sandbox

Requirements

Name	Requirement	Type	Description
id	\d+	integer	The genre unique identifier.

Status Codes

Status Code	Description
200	Returned if genre has been successfully edited
400	Returned if errors
500	Returned if server errors

```
/**
 * ...
 * @Doc\ApiDoc(
 *     section="Genres",
 *     description="Delete an existing genre.",
 *     statusCodes={
 *         204="Returned if genre has been successfully deleted",
 *         404="Returned if genre does not exist",
 *         500="Returned if server error"
 *     },
 *     requirements={
 *         {
 *             "name"="id",
 *             "dataType"="integer",
 *             "requirement"="\d+",
 *             "description"="The genre unique identifier."
 *         }
 *     }
 * )
 */
public function deleteGenreAction(...)
```

GET /api/genres/ — Get the list of all genres.

POST /api/genres/ — Creates a new genre.

/api/genres/{id}

GET /api/genres/{id} — Get one genre.

PUT /api/genres/{id} — Edit an existing genre.

DELETE /api/genres/{id} — Delete an existing genre.

Documentation Sandbox

Requirements

Name	Requirement	Type	Description
id	\d+	integer	The genre unique identifier.

Status Codes

Status Code	Description
204	Returned if genre has been successfully deleted
404	Returned if genre does not exist
500	Returned if server error

```
/**
 * ...
 * @Doc\ApiDoc(
 *     section="Movies",
 *     resource=true,
 *     description="Get the list of all movies.",
 *     statusCodes={
 *         200="Returned if successful",
 *     }
 * )
 */
public function getMoviesAction(...)
```

Movies

/api/movies/

GET /api/movies/ — Get the list of all movies.

Documentation Sandbox

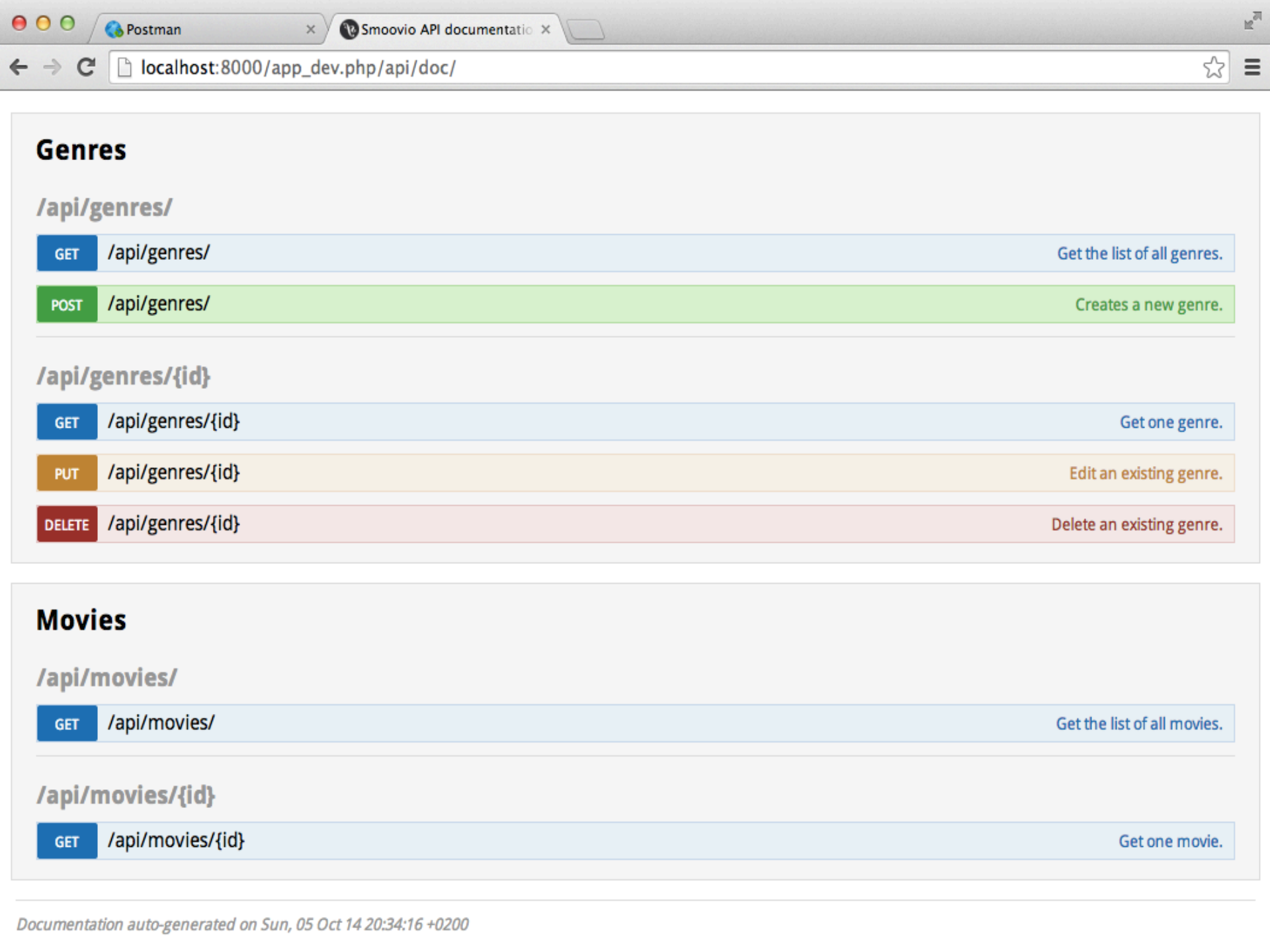
Filters

Name	Information	
keyword	Requirement	[a-zA-Z0-9]+
	Description	The keyword to search for.
order	Requirement	asc|desc
	Description	Sort order (asc or desc).
	Default	asc
limit	Requirement	\d+
	Description	Max number of movies per page.
	Default	20
offset	Requirement	\d+
	Description	The paginated items offset.
	Default	0

Status Codes

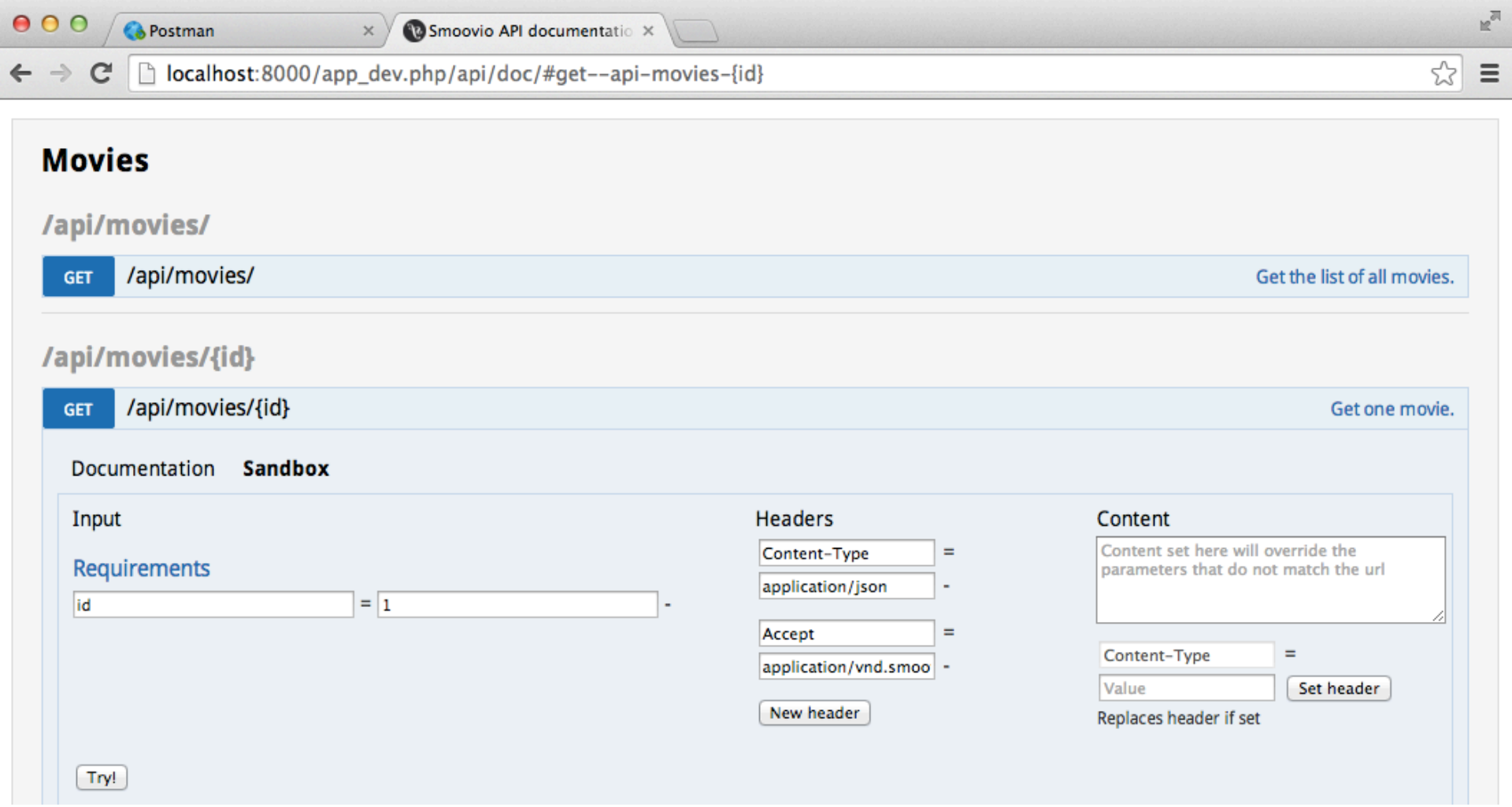
Status Code	Description
200	Returned if successful

```
/**
 * ...
 * @Doc\ApiDoc(
 *     section="Movies",
 *     resource=true,
 *     description="Get one movie.",
 *     statusCodes={
 *         200="Returned if successful",
 *     },
 *     requirements={
 *         {
 *             "name"="id",
 *             "dataType"="integer",
 *             "requirement"="\d+",
 *             "description"="The genre unique identifier."
 *         }
 *     }
 * )
 */
public function getMovieAction(...)
```


Postman
Smoovio API documentatio
localhost:8000/app_dev.php/api/doc/
Genres
/api/genres/
GET /api/genres/ Get the list of all genres.
POST /api/genres/ Creates a new genre.
/api/genres/{id}
GET /api/genres/{id} Get one genre.
PUT /api/genres/{id} Edit an existing genre.
DELETE /api/genres/{id} Delete an existing genre.
Movies
/api/movies/
GET /api/movies/ Get the list of all movies.
/api/movies/{id}
GET /api/movies/{id} Get one movie.
Documentation auto-generated on Sun, 05 Oct 14 20:34:16 +0200

NelmioApiDocBundle > Sandbox

The generated documentation also embeds a sandbox mode to test the API.



Set **Accept and Content-Type** HTTP headers like in Postman Chrome extension.

Request URL

```
GET /api/movies/1
```

Response Headers [Expand] [Profiler]

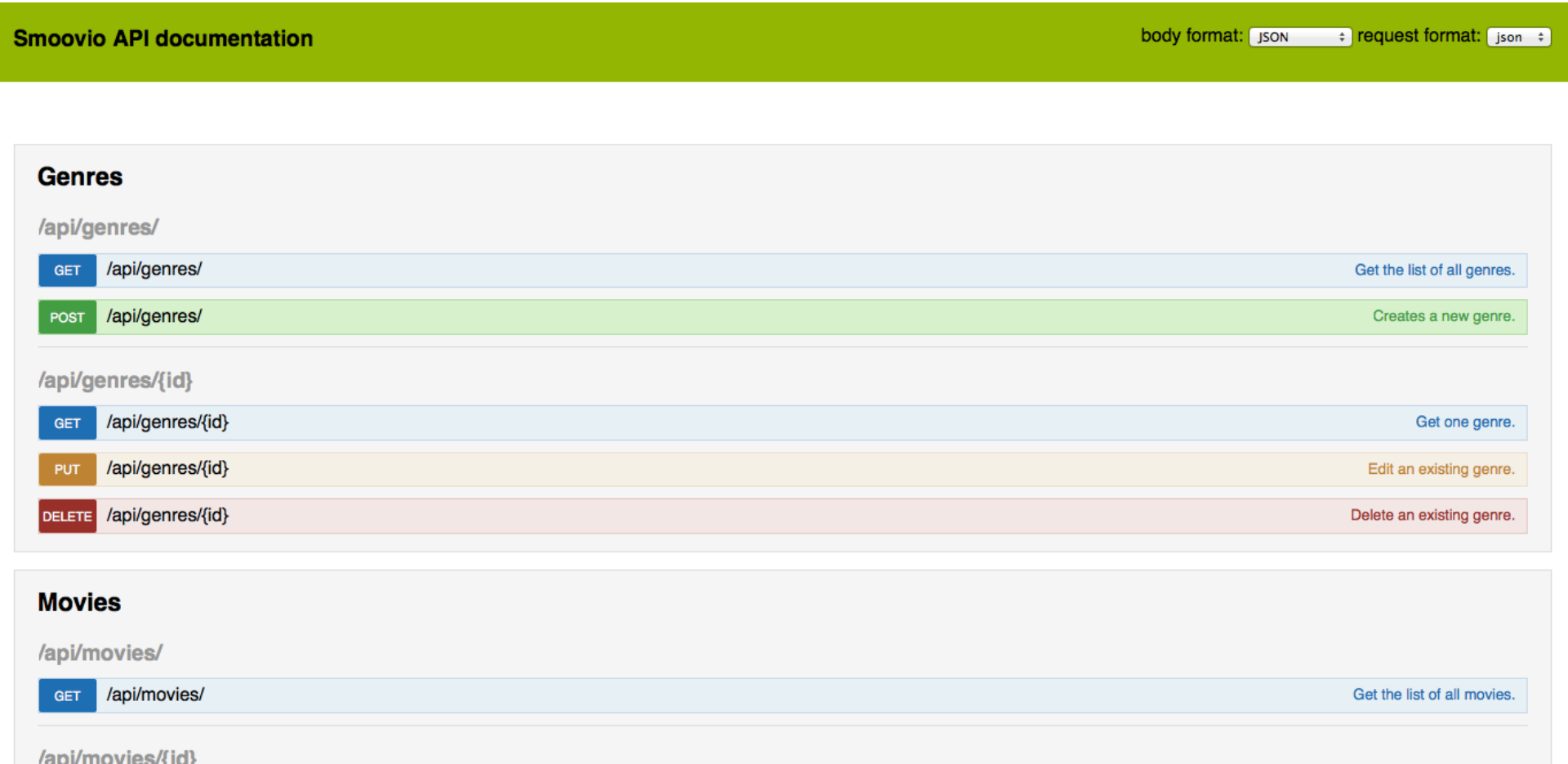
```
200 OK
```

Response Body [Raw]


```
{
  "id": 1,
  "title": "The Shawshank Redemption",
  "slug": "the-shawshank-redemption",
  "description": "Framed in the 1940s for the double murder of his wife and her lover, upstanding banker Andy Dufresne beg
  "duration": 8520,
  "release_date": "1994-09-14T00:00:00+0200",
  "_links": {
    "self": {
      "href": "http://localhost:8000/app_dev.php/api/movies/1"
    }
  },
  "_embedded": {
    "genre": {
      "id": 1,
      "title": "Crime",
      "slug": "crime",
      "_links": {
        "self": {
          "href": "http://localhost:8000/app_dev.php/api/genres/1"
        },
        "edit_genre": {
          "href": "http://localhost:8000/app_dev.php/api/genres/1"
        },
        "delete_genre": {
          "href": "http://localhost:8000/app_dev.php/api/genres/1"
        }
      }
    },
    "roles": [
      {
```

The built-in **doc:api:dump** command exports the documentation as JSON, Markdown or HTML format.

```
php app/console api:doc:dump --format=json > doc.json
php app/console api:doc:dump --format=markdown > doc.md
php app/console api:doc:dump --format=html > doc.html
```



SensioLabs Training Department

Address

92-98 Boulevard Victor Hugo

92 115 Clichy Cedex

France

Phone

+33 140 998 205

Email

training@sensiolabs.com

training.sensiolabs.com

SensioLabs