# Introduction to Apigility

Rob Allen
October 2014

## APIs are becoming commonplace

## APIs are hard

## API considerations

- Content negotiation
- Error reporting
- Discovery
- Authentication
- Documentation
- HTTP method negotiation
- Versioning
- Validation
- Authorisation

apigility
by zend framework

# An opinionated API builder

(Screencast: Hello World)

## Apigility

- Administration system
- Runtime API engine
- PHP, built on Zend Framework 2

## Versioning by default

**Media type:**

```
GET /albums HTTP/1.1
Accept: application/vnd.music.v1+json
```

**URL-based:**

```
/v1/albums
```

# JSON

Hypermedia Application Language (HAL) - application/hal+json

```
{
    "_links": {
        "self": {
            "href": "http://localhost:8080/albums/1"
        }
    },
    "artist": "Eninem",
    "id": "1",
    "title": "The Marshall Mathers LP 2"
}
```

## Error Reporting

API Problem - application/problem+json

```
{
    "type": "/api/problems/forbidden",
    "title": "Forbidden",
    "detail": "Your API key is missing or invalid.",
    "status": 403,
    "authenticationUrl": "/api/oauth"
}
```

## HTTP features

- Method negotiation
- Accept checking
- Content-Type acceptance
- Automatic OPTIONS

# Validation

```
PATCH /albums/1 HTTP/1.1
Content-Type: application/json

{ "title": "" }


422 Unprocessable Entity
Content-Type: application/problem+json

{
  "type": "w3.org/Protocols/rfc2616/rfc2616-sec10.html",
  "title": "Unprocessable Entity",
  "detail": "Failed validation",
  "status": 422,
  "validation_messages": {
      "title": "Invalid title; must be a non-empty string"
  }
}
```

## Authentication

- HTTP Basic and Digest (for internal APIs)
- OAuth2 (for public APIs)
- Event-driven, to accommodate anything else
- Returns problem response early
- Correct errors: 401, 403, etc.

## Hyperlinking: Pagination

Automatic when you use `Zend\Paginator\Paginator`.

```
{
    _links: {
        self: { href: "/api/albums?page=3" },
        first: { href: "/api/albums" },
        last: { href: "/api/albums?page=14" },
        prev: { href: "/api/albums?page=2" },
        next: { href: "/api/albums?page=4" }
    }
}
```

## Documentation

- Written within admin while setting up API
- Automatically populated via validation admin
- User documentation:
    - apigility/documentation/{API name}/V1
    - JSON or HTMl based on accept header
    - Swagger available too

## Let’s see it in action!

## Getting Started

Install:

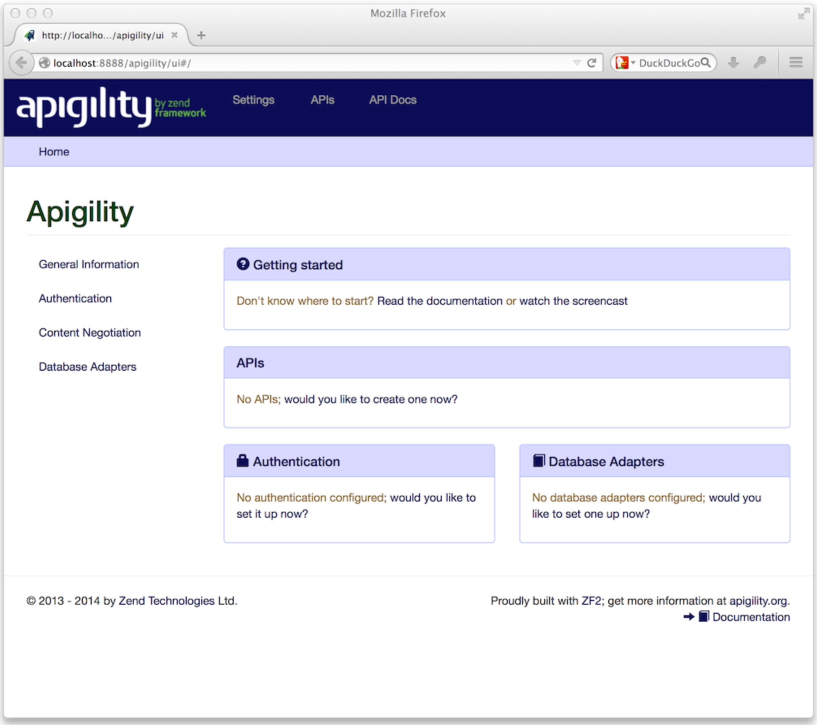
```
$ composer.phar create-project \
  -sdev zfcampus/zf-apigility-skeleton music
```

Development Mode:

```
$ cd music
$ php public/index.php development enable
```

Run the admin web UI:

```
$ php -S 0:8080 -t public/ public/index.php
```


Mozilla Firefox
http://localho.../apigility/ui
localhost:8888/apigility/ui#/
DuckDuckGo
apigility by zend framework
Settings
APIs
API Docs
Home

Apigility

Getting started

Don't know where to start? Read the documentation or watch the screencast

APIs

No APIs; would you like to create one now?

Authentication

No authentication configured; would you like to set it up now?

Database Adapters

No database adapters configured; would you like to set one up now?

© 2013 - 2014 by Zend Technologies Ltd.

Proudly built with ZF2; get more information at apigility.org.

Documentation

(Screencast: Create a REST API)

## Test with curl

```
$ curl -s -H "Accept: application/vnd.music.v1+json" \
  http://localhost:8080/albums | python -mjson.tool

{
  "detail": "The GET method has not been defined for
             collections",
  "status": 405,
  "title": "Method Not Allowed",
  "type": "w3.org/Protocols/rfc2616/rfc2616-sec10.html"
}
```

# Source code

Apigility has created a module called `Music` for our API.

The `Album` endpoint is in: `src/Music/V1/Rest/Album`

Classes:

| | |
|---|---|
| `AlbumResource` | entry point to service |
| `AlbumCollection` | a collection of albums |
| `AlbumEntity` | a single album |

## AlbumResource class

Methods for the collection: `/albums`

| Class method | HTTP | Notes |
|---|---|---|
| `fetchAll` | GET | retrieve all items |
| `create` | POST | create an item |
| `replaceList` | PUT | replace all items |
| `deleteList` | DELETE | Delete all items |

# AlbumResource class

Methods for a single resource: `/albums/[album_id]`

| Class method | HTTP | Notes |
|---|---|---|
| `fetch` | GET | retrieve an item |
| `patch` | PATCH | update some fields |
| `update` | PUT | replace an item |
| `delete` | DELETE | delete an item |

## The data model

```
class AlbumEntity
{
    protected $id;
    protected $artist;
    protected $title;
}

class AlbumMapper
{
  public function fetchAll($filter) { /* .. */ }
  public function fetchOne($id) { /* .. */ }
  public function save($album) { /* .. */ }
}
```

## Fetching the collection

```
class AlbumResource extends AbstractResourceListener
{
  public function fetchAll($params = array())
  {
    // return an AlbumCollection
    return $this->mapper->fetchAll($params);
  }
}
```

Test:

```
$ curl -s -H "Accept: application/vnd.music.v1+json" \
  http://localhost:8080/albums?title=night \
  | python -mjson.tool
```

# Fetching the collection

```
{
    "_embedded": {
        "album": [
            {
                "_links": {
                    "self": { "href": "http://localhost:8080/albums/4" }
                },
                "artist": "Andre Rieu",
                "id": "4",
                "title": "Music of the Night"
            }
        ]
    },
    "_links": {
        "first": { "href": "http://localhost:8080/albums" },
        "last": { "href": "http://localhost:8080/albums?page=1" },
        "self": { "href": "http://localhost:8080/albums?page=1" }
    },
    "page_count": 1,
    "page_size": 25,
    "total_items": 1
}
```

# Fetching a single resource

```
class AlbumResource extends AbstractResourceListener
{
  public function fetch($id)
  {
    // return an AlbumEntity
    return $this->mapper->fetchOne($id);
  }
}
```

Test:

```
$ curl -s -H "Accept: application/vnd.music.v1+json" \
  http://localhost:8080/albums/1 | python -mjson.tool
```

## Response

```
{
    "_links": {
        "self": {
            "href": "http://localhost:8080/albums/1"
        }
    },
    "artist": "Eninem",
    "id": "1",
    "title": "The Marshall Mathers LP 2"
}
```

## Creating a resource

```
public function create($data)
{
  // return an AlbumEntity
  return $this->mapper->save($data);
}
```

POST to the collection

```
$ curl -s -X POST -H "Content-type: application/json" \
  -H "Accept: application/vnd.music.v1+json"  \
  -d '{"title":"True", "artist":"Avicii"}' \
  http://localhost:8080/albums |  python -mjson.tool
```

## Response

Header:

```
HTTP/1.1 201 Created
Location: http://localhost:8080/albums/7
```

Body:

```
{
    "_links": {
        "self": {
            "href": "http://localhost:8080/albums/7"
        }
    },
    "artist": "Avicii",
    "id": "7",
    "title": "True"
}
```

# Updating a resource

```
public function update($id, $data)
{
  // return an AlbumEntity
  return $this->mapper->save($data, $id);
}
```

PUT to the resource

```
$ curl -s -X PUT -H "Content-type: application/json" \
  -H "Accept: application/vnd.music.v1+json"  \
  -d '{"title":"True!", "artist":"Avicii"}' \
  http://localhost:8080/albums/7 |  python -mjson.tool
```

## Response

Header:

```
HTTP/1.1 200 OK
```

Body:

```
{
    "_links": {
        "self": {
            "href": "http://localhost:8080/albums/7"
        }
    },
    "artist": "Avicii",
    "id": "7",
    "title": "True!"
}
```

## Validation

- Built into the Apigility admin
- Tested when routing: very fast to fail

(Screencast: Validation)

# Validation

POST with an an empty artist to the collection

```
$ curl -s -X POST -H "Content-type: application/json" \
  -H "Accept: application/vnd.music.v1+json"  \
  -d '{"title":"Greatest Hits", "artist":""}' \
  http://localhost:8080/albums |  python -mjson.tool
```

# Response

Header:

```
HTTP/1.1 422 Unprocessable Entity
```

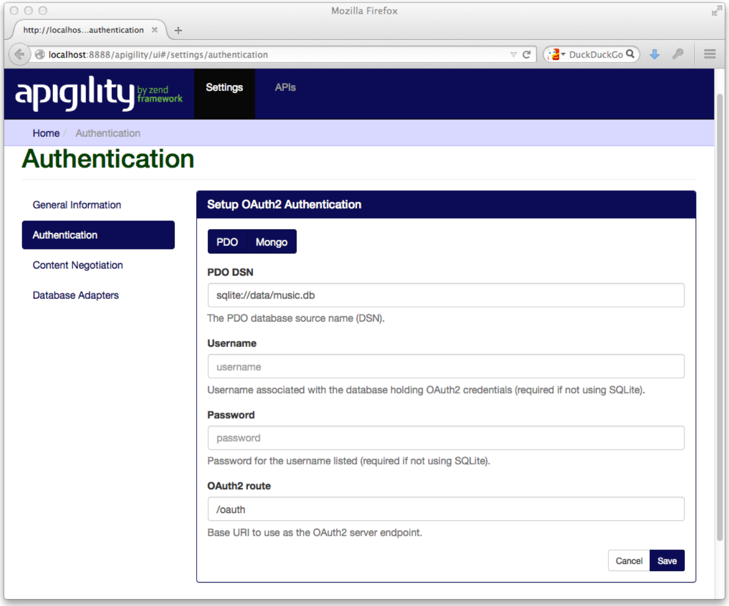
Body:

```
{
    "detail": "Failed Validation",
    "status": 422,
    "title": "Unprocessable Entity",
    "type": "http://www.w3.org/Protocols/rfc2616/rfc2616-sec10.html",
    "validation_messages": {
        "artist": {
            "isEmpty": "Value is required and can't be empty"
        }
    }
}
```

## Authentication

- HTTP Basic uses htpassword file
- HTTP Digest uses htdigest file
- OAuth2 uses database.
    - knpuniversity.com/screencast/oauth/intro
    - bshaffer.github.io/oauth2-server-php-docs/

## OAuth2


Mozilla Firefox
http://localhos...authentication
localhost:8888/apigility/ui#/settings/authentication
DuckDuckGo
apigility by zend framework
Settings
APIs
Home / Authentication

Authentication

Setup OAuth2 Authentication

**PDO DSN**

sqlite://data/music.db

The PDO database source name (DSN).

**Username**

username

Username associated with the database holding OAuth2 credentials (required if not using SQLite).

**Password**

password

Password for the username listed (required if not using SQLite).

**OAuth2 route**

/oauth

Base URI to use as the OAuth2 server endpoint.

## OAuth2 process

1. Get an access token.
2. Send it on all subsequent requests:

```
Authorization: Bearer 5ce33e13e66c5ff723f997387e183c
```

## Password grant type

```
POST /oauth

{
  "grant_type": "password",
  "client_id" : "testclient",
  "username": "rob@akrabat.com",
  "password": "password"
}
```

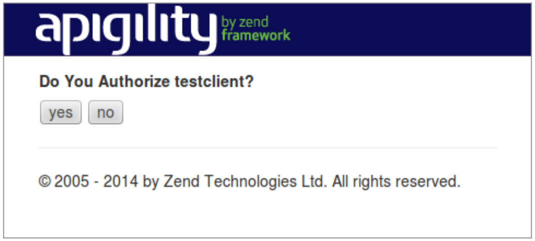
Returns:

```
{
  "access_token": "7f4ac44eb70616204748c41c457b8867e",
  "expires_in": 3600,
  "token_type": "Bearer",
  "scope": null,
  "refresh_token": "3f0d94d87dd891813feddcb4b24f0963"
}
```

# Authorization code (1)

Request authorisation code

http://localhost:8888/oauth/authorize?response_type=code
&client_id=testclient&redirect_uri=/oauth/receivecode



Do You Authorize testclient?

© 2005 - 2014 by Zend Technologies Ltd. All rights reserved.

Clicking Yes will redirect to ‘redirect_uri’ with the *authorization code* in the query string.

## Authorization code (2)

Request access token using authorisation code

```
POST /oauth
```

```
{
  "grant_type": "authorization_code",
  "client_id" : "testclient",
  "client_secret": "testpass",
  "code": "a4dd64ffb43e6bfe16d47acfab1e68d9c"
}
```

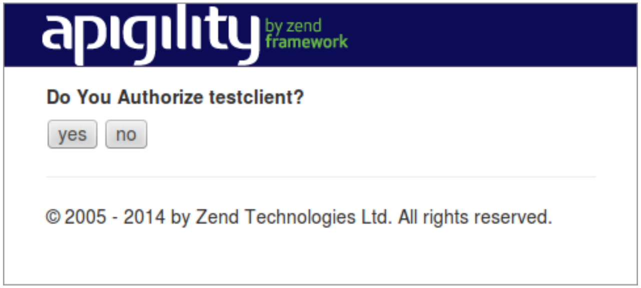
Returns:

```
{
  "access_token": "907c762e069589c2cd2a229cdae7b8778",
  "expires_in": 3600,
  "token_type": "Bearer",
  "refresh_token": "43018382188f462f6b0e5784dd44c36f"
}
```

# Authorization code

Request authorisation code

http://localhost:8888/oauth/authorize?response_type=code
&client_id=testclient&redirect_uri=/oauth/receivecode



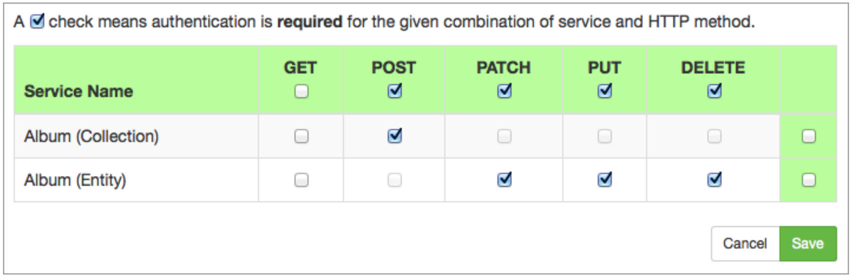
Do You Authorize testclient?

© 2005 - 2014 by Zend Technologies Ltd. All rights reserved.

(Yes, you can customise it!)

# Protect your API

via configuration; fails early.



A ✓ check means authentication is **required** for the given combination of service and HTTP method.

| Service Name | GET ☐ | POST ✓ | PATCH ✓ | PUT ✓ | DELETE ✓ | |
|---|---|---|---|---|---|---|
| Album (Collection) | ☐ | ✓ | ☐ | ☐ | ☐ | ☐ |
| Album (Entity) | ☐ | ☐ | ✓ | ✓ | ✓ | ☐ |

# Protect your API

via code in your Resource class.

```
use ZF\MvcAuth\Identity\AuthenticatedIdentity as Identity;

if ($this->getIdentity() instanceof Identity) {
    $identity = $this->getIdentity()
    $user = $identity->getAuthenticationIdentity();
}
```