



Getting Started with DSpace 7: Advanced Training

DSPACE

Andrea Bollini, 4Science

Art Lowel, Atmire

Tim Donohue, DuraSpace



Workshop Schedule

- ❖ DSpace 7 UI deep dive (Angular)
- ❖ Customizing UI (beyond branding)
- ❖ DSpace 7 REST API deep dive
- ❖ Contributing back to DSpace



Hands-on Prerequisites

Instructions on

<https://tinyurl.com/or2019-dspace7-wiki>

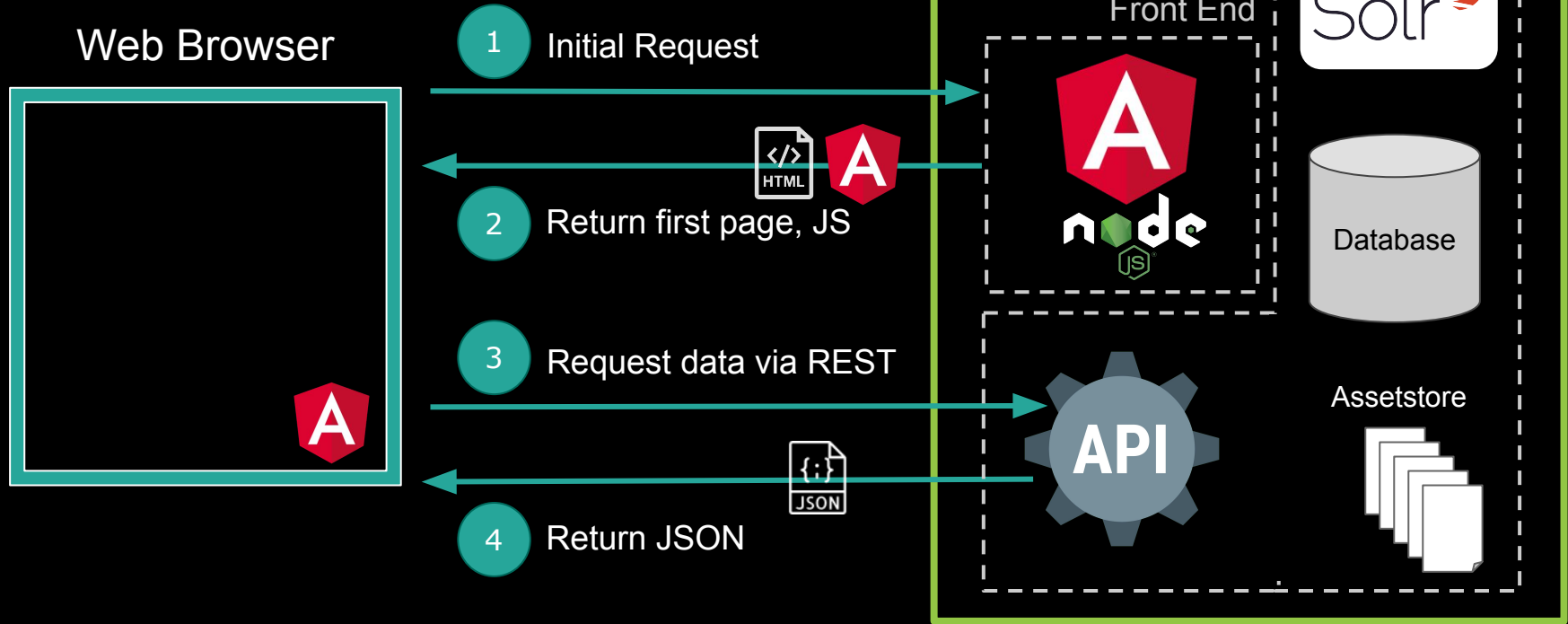


DSPACE



What is Angular?
How does it work?

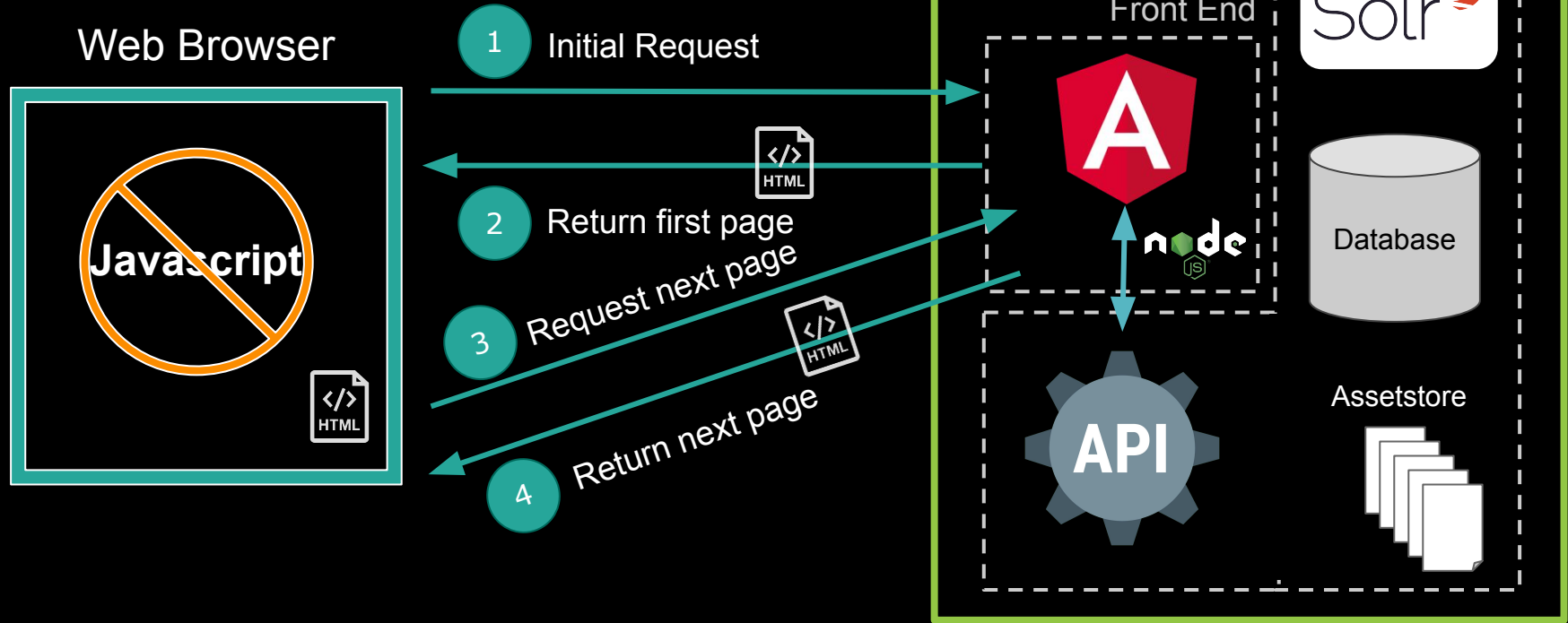
DSpace + Angular Architecture



HTML logo: <https://freeiconshop.com/icon/html-icon-outline/>

JSON logo: http://www.flaticon.com/free-icon/json-file_136443

DSpace + Angular Architecture



Via Angular Universal



Angular Universal

Goal: support server-side rendering for Angular apps ... *using the same code that's used by the client*

- ★ Better perceived performance
 - First page *always* rendered on server
- ★ Search Engine Optimization
- ★ Support other clients lacking Javascript

<https://universal.angular.io/>



Try it out now!

DSpace 7 UI demo

<https://dspace7-demo.atmire.com/>

(uses the REST API demo as backend)

DSpace 7 REST API demo:

<https://dspace7.4science.cloud/dspace-spring-rest/>

Run locally via Docker:

<https://dspace-labs.github.io/DSpace-Docker-Images/>





Intro to Angular

<https://angular.io/tutorial>

Angular Concepts

- ❖ Node / NPM / Yarn
- ❖ TypeScript Language
- ❖ Main architectural elements:
 - Components
 - Templates
 - Services
 - Modules

<https://angular.io/docs/ts/latest/>

Building / Running Angular Apps



: Server side JS platform



: Node's package manager

Pulls in dependencies / third-party tools from registry



yarn : third-party Node package mgr

- Same config as NPM (package.json)
- 3-5 times faster than NPM
- OS collab between Facebook, Google, and Tilde



TypeScript Language



- Extension of ES6 (latest JavaScript)
- Adds types and annotations
 - No more "var"
 - Expandable / sharable (Typings registry)
- Examples:

<code>private title: string;</code>	(String variable)
<code>private myItem: Item;</code>	(Item variable)
<code>private myParam: any;</code>	(any type)

TypeScript Language



- Compiles to regular JavaScript & errors can be detected at compile time
- May look familiar to Java and .NET developers
 - Interfaces, Generics, Decorators, ...
- Much better IDE integration than JS

<https://www.typescriptlang.org/>



TypeScript Example



```
1  import { Metadatum } from "../metadatum.model";
    ...
2  export abstract class DSpaceObject implements CacheableObject {
3      name: string;
4      metadata: Array<Metadatum>;
    ...
5      findMetadata(key: string, language?: string): string {
        const metadatum = this.metadata
6            .find((metadatum: Metadatum) => {
                return metadatum.key === key &&
                    (isEmpty(language) || metadatum.language === language)
            });
7      return metadatum.value;
    }
```



Angular Architecture Overview

“You write Angular applications by composing HTML *templates* with Angularized markup, writing *component* classes to manage those templates, adding application logic in *services*, and boxing components and services in *modules*.”

<https://angular.io/docs/ts/latest/guide/architecture.html>



DSPACE

Angular: Main elements

- ❑ **Templates:** compose HTML
- ❑ **Components:** display data via templates
- ❑ **Services:** retrieve data for components
- ❑ **Modules:** package components & services



Angular: Templates

- HTML-like (almost all HTML is valid)
- Load other Components via their “selector” (i.e. HTML tag)
- Components also have their own templates

<https://angular.io/docs/ts/latest/guide/template-syntax.html>

Example Template

```
1 <div *ngIf="topLevelCommunities.hasSucceeded | async">
2   <h2>{{'home.top-level-communities.head' | translate}}</h2>
   <p class="lead">{{'home.top-level-communities.help' | translate}}</p>
   <ul>
3     <li *ngFor="let community of (topLevelCommunities.payload | async)">
       <p>
4         <span class="lead"><a [routerLink]="['/communities', community.id]">
           {{community.name}}</a></span><br>
5         <span class="text-muted">{{community.shortDescription}}</span>
       </p>
     </li>
    ...
```



Template Syntax Hints

`{{obj.value}}`

Prints value of "obj.value" expression/variable

`<div [class]="obj.class">` [Property binding]

Sets HTML attr "class" to value of "obj.class"

Square brackets set a property

`<button (click)="doThing($event)">` (Event binding)

Call "doThing()" method (in component) when click is triggered

*Parentheses respond to an *event**

`<my-component [(title)]="name">`

Two way binding. Sets property "title" to name, and updates "name" if title is changed (i.e. "titleChange" event is triggered)

`<my-component [title]="name" (titleChange)="name=$event">`



Template Syntax Hints

```
<div *ngIf="showSection">
```

Only display this <div> if "showSection" is true

```
<li *ngFor="let item of list">
```

Display a for every "item" in "list".

```
<div [ngClass]="{'active': isActive}">
```

Set the HTML "class" to "active", if "isActive" is true

<https://angular.io/guide/template-syntax>

<https://angular.io/guide/cheatsheet>



DSPACE

Angular: Main elements

- ☑ Templates: compose HTML
- ☐ Components: display data via templates
- ☐ Services: retrieve data for components
- ☐ Modules: package components & services



Angular: Components

- *The building blocks of an Angular app*
- Allow you to create *new* HTML tags that come with their own code and styling
- Consist of a 'view' and a 'controller' in the traditional MVC sense



Everything's a Component

- Header and footer components
- Thumbnail and file list components
- Collection list component
- Specific metadata field components (extend shared one)
- Entire page too!

The screenshot shows a DSpace interface with a dark blue header containing 'DSpace' and a 'Home' link. The main content area has a title 'Do Open-Access Articles Have a Greater Research Impact?' in a purple box. To the left of the abstract are three stacked components: a 'No Thumbnail' placeholder in a grey box, a 'Files' section in a green box listing 'do_open_access_CRL.pdf (621.71 KiB)', and a metadata section in a purple box with fields for 'Date' (2004-09-01) and 'Author' (Antelman, Kristin). To the right of the abstract is a large rounded rectangle containing the 'Abstract' text. Below the abstract is a 'URI' box with the URL 'http://hdl.handle.net/123456789/8871'. Underneath the URI is a 'Collections' box in a cyan border listing 'A Test Collection' and 'Another Test Collection'. At the bottom of the collections box is a 'Full item page' button. The footer is a grey bar with the text 'DSpace software copyright © 2002-2017 DuraSpace'.

DSpace Home

Do Open-Access Articles Have a Greater Research Impact?

No Thumbnail

Files

[do_open_access_CRL.pdf \(621.71 KiB\)](#)

Date

2004-09-01

Author

Antelman, Kristin

Abstract

Although many authors believe that their work has a greater research impact if it is freely available, studies to demonstrate that impact are few. This study looks at articles in four disciplines at varying stages of adoption of open access—philosophy, political science, electrical and electronic engineering and mathematics—to see whether they have a greater impact as measured by citations in the ISI Web of Science database when their authors make them freely available on the Internet. The finding is that, across all four disciplines, freely available articles do have a greater research impact. Shedding light on this category of open access reveals that scholars in diverse disciplines are adopting open-access practices and being rewarded for it.

URI

<http://hdl.handle.net/123456789/8871>

Collections

[A Test Collection](#)

[Another Test Collection](#)

[Full item page](#)

DSpace software copyright © 2002-2017 DuraSpace

Angular: Components

- Each part of a webpage is a **Component**:
 - ... 'implements' Interface(s), e.g. `onInit`, `onDestroy`
 - ... 'extends' another Component
 - ... has a selector (HTML-like tag)
e.g. `<news>` = `NewsComponent`
 - ... has a constructor (defines its inputs)
 - ... has a template (view) and/or methods (actions)



Calling a Component (from a template)

```
<div class="wrapper">  
  <ds-header></ds-header>  
  
  <main>  
    ...  
  </main>  
  
  <ds-footer></ds-footer>  
</div>
```



Component's Class

```
1  @Component({
2      selector: 'ds-header',
3      styleUrls: ['header.component.css'],
4      templateUrl: 'header.component.html'
5  })
6  export class HeaderComponent implements OnInit {
7      isNavBarCollapsed: boolean;
8      ...
9      ngOnInit(): void {
10         this.isNavBarCollapsed = true;
11     }
12     toggle(): void {
13         this.isNavBarCollapsed = !this.isNavBarCollapsed;
14     }
15 }
```



Component's Template

```
1  <button (click)="toggle()" aria-controls="collapsingNav">
    <i class="fa fa-bars fa-fw" aria-hidden="true"></i>
</button>

2  <div id="collapsingNav" [ngbCollapse]="isNavBarCollapsed">
3    <a class="nav-link" routerLink="/home"
routerLinkActive="active">
        {{ 'header.home' | translate }}
    </a>
</div>
```



Angular: Main elements

- ☑ Templates: compose HTML
- ☑ Components: display data via templates
- ☐ Services: retrieve data for components
- ☐ Modules: package components & services



Angular: Services

Reusable "chunks" of code should be Services

- Singletons (shared/used globally)
- Provide streams of data for components
`this.restService.get('/items')`
- Provide operations to add or modify data
`this.cacheService.add(item)`



Dependency Injection (DI)

Inject Services into Components that need them

```
// (1) Define ItemDataService class as injectable
```

```
@Injectable()
```

```
export class ItemDataService { ... }
```

```
// (2) Then, inject ItemDataService as input to a Component class
```

```
export class MyComponent {
```

```
    constructor(private items: ItemDataService) {}
```

```
}
```



Example Service Class

```
1  @Injectable()
2  export class DSpaceRESTv2Service {
3      constructor(public http: Http) {}

4      get(relativeURL: string, options?: RequestOptionsArgs): Observable<string> {
5          return this.http.get(new RESTURLCombiner(relativeURL).toString(),
                                options)
                                .map(res => res.json())
                                .catch(err => {
6                                  console.log('Error: ', err);
                                  return Observable.throw(err);
                                });
7      }
8  }
```



Angular: Main elements

- ☑ Templates: compose HTML
- ☑ Components: display data via templates
- ☑ Services: retrieve data for components
- ☐ Modules: package components & services



Angular: Modules

- Classes used to simply organize your application into “blocks” of functionality
- Grouping of components and/or services. Import other modules.
- Angular App itself is a Module (`/src/app/app.module.ts`)



Angular: Main elements

- ✓ ☐ Templates: compose HTML
- ✓ ☐ Components: display data via templates
- ✓ ☐ Services: retrieve data for components
- ✓ ☐ Modules: package components & services





The DSpace 7 UI

<https://github.com/DSpace/dspace-angular/>

Hands-on Prerequisites

Instructions on

<https://tinyurl.com/or2019-dspace7-wiki>



DSPACE

DSpace-Angular: Folder structure

/

<code>config/</code>	(configuration files)
<code>resources/</code>	(static files, e.g. i18n, images)
<code>src/app/</code>	(Angular application source code)
<code>src/backend/</code>	(mock REST data)
<code>src/platform/</code>	(root Angular modules for client/server)
<code>dist/</code>	(compiled application created by yarn/npm)



DSpace-Angular: /src/app

/src/app

- Each “feature” is in a separate subfolder
- File naming convention
 - `header.component.ts` (Header component class)
 - `header.component.html` (Header component template)
 - `header.component.scss` (Header component styles)
 - `header.component.spec.ts` (Header comp specs / tests)
 - `community-page.module.ts` (Community Page module definition)
 - `dspace-rest-v2.service.ts` (REST API service definition)



Package.json: build scripts

```
"scripts": {  
  ...  
  "server": "node dist/server/index.js",  
  ...  
  "start": "yarn run server",  
  ...  
},
```

- "scripts" section contains all the build scripts in the project
- `yarn run ${scriptname}`
- scripts can call each other



Creating your branch

- `git remote add workshop`
`https://github.com/Dspace-Labs/dspace-angular-`
`workshops.git`
- `git fetch workshop`
- `git checkout or2019-advanced-start`
- `git checkout -b or2019-advanced`



Yarn commands when switching branch

- Ensure dependencies are up to date
 - `yarn run clean`
 - `yarn install`
- Restart the dev server
 - `yarn run watch`



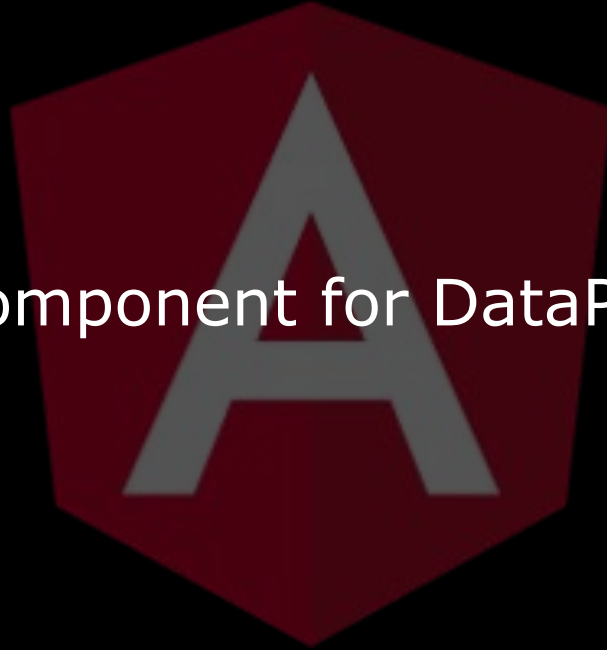
DataPackage and DataType entities

Goal: Create custom item pages for the new
DataPackage and DataType entities



Step 1: DataPackage component

Create a Component for DataPackage pages



Creating the DataPackage component

- Start from the PublicationComponent
 - Go to `src/app/+item-page/simple/item-types`
 - Copy the `publication` folder to `data-package`
 - Rename the files within to `data-package.*`
 - Remove the `.spec.ts` file



Creating the DataPackage component

- in `data-package.component.ts`
 - remove `@rendersItemType(DEFAULT_ITEM_TYPE, ...)`
 - change `@rendersItemType('Publication', ...)`
to `@rendersItemType('DataPackage', ItemViewMode.Full)`
 - Change the selector to `ds-data-package`
 - Update the `styleUrls` and `templateUrl`



Creating the DataPackage component

- in `data-package.component.html`
 - Replace `{{'publication.page.titleprefix' | translate}}`
With `Data Package:`
 - That way we'll see when it's being used



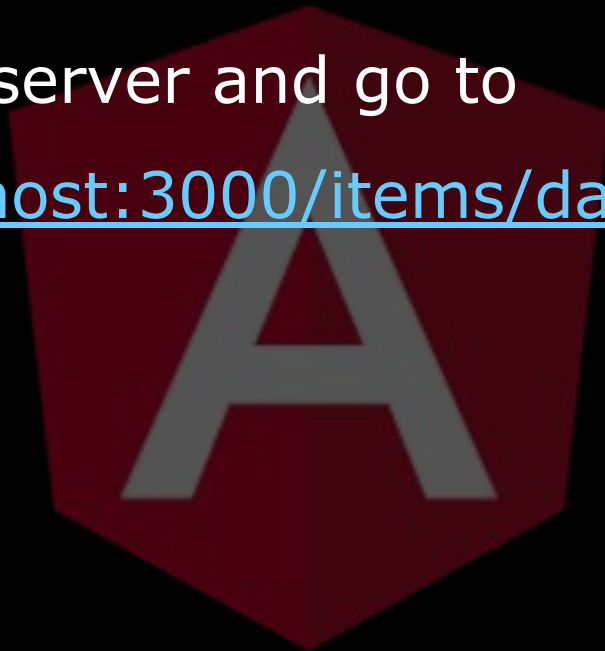
Creating the DataPackage component

- In `src/app/+item-page/item-page.module.ts`
 - Add the new component to the `declarations` and `entryComponents` sections.
 - All new components should be added to `declarations`
 - `entryComponents` is only for components that need to be switched on the fly



Creating the DataPackage component

- Restart the server and go to <http://localhost:3000/items/datapackage>



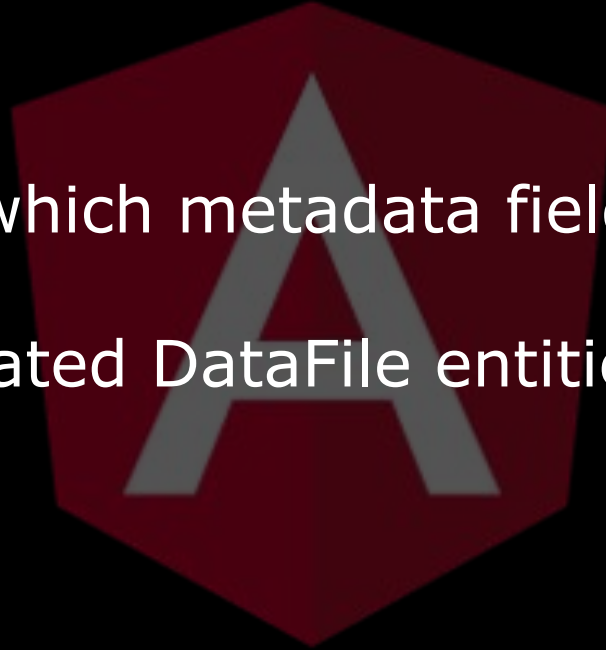
Tag: or2019-advanced-1

- To sync your branch with the solution run:
 - `git reset or2019-advanced-start --hard`
 - `git clean -f -d`
 - `git merge or2019-advanced-1`



Step 2: Configure the DataPackage template

- Goals
 - Change which metadata fields are shown
 - Show related DataFile entities



DataPackage metadata fields

- Fix the journal title
 - It is in `prism.publicationName` instead of `journal.title`
- Remove fields for `issn`, `volume-title` and `citations`
- Replace the `URI` field with `DOI`:
`dc.relation.isreferencedby`



Turn the DOI in to a link

```
<ds-metadata-field-wrapper [label]="DOI"  
*ngVar="item?.firstMetadataValue('dc.relation.isreferencedby') as  
doi">  
  <a href="https://doi.org/{{doi}}" target="_blank">{{doi}}</a>  
</ds-metadata-field-wrapper>
```



DataPackage relations

- Remove relations for projects, org units and journals from both the HTML and the ts
- add a relation for isDataFileOfDataPackage



DataPackage relations

- in `data-package.component.ts`:
 - Add a field: `dataFiles$: Observable<Item[]>;`
 - And populate it:

```
this.dataFiles$ = this.resolvedRelsAndTypes$.pipe(  
  filterRelationsByTypeLabel('isDataFileOfDataPackage'),  
  relationsToItems(this.item.id, this.ids)  
);
```



Observables

- `this.resolvedRelsAndTypes$` is an observable
 - it contains a mapping of relationshipTypes and their relationships
 - “observable” means its value can change over time
 - It will change when the server responds with relationship data.



Piping observables

- the `pipe()` function allows us to run a number of operations on observables.
- The output of each operation is the input for the next.
- The output of `pipe()` is a new observable that will be updated every time the source observable changes



DataPackage relations

- `filterRelationsByTypeLabel('isDataFileOfDataPackage')` will only let through relationship objects of the type `isDataFileOfDataPackage`
- `relationsToItems(this.item.id, this.ids)` will use those relationship objects to retrieve the Items they link to



DataPackage relations

- The async pipe will ensure the template updates when the observable changes

```
<ds-related-items  
  [items]="dataFiles$ | async"  
  [label]="Data Files">  
</ds-related-items>
```



Tag: or2019-advanced-2

- To sync your branch with the solution run:
 - `git reset or2019-advanced-start --hard`
 - `git clean -f -d`
 - `git merge or2019-advanced-2`



Step 3: DataFile Component

- Goals
 - Create the component
 - Add a relation back to the DataPackage



The DataFile component

- Start from `DataPackageComponent`
 - Copy its folder to `data-file`
 - Rename the files within to `data-file.*`



The DataFile component

- in `data-file.component.ts`
 - change `@rendersItemType('DataPackage', ...)`
to `@rendersItemType('DataFile', ItemViewMode.Full)`
 - Change the selector to `ds-data-file`
 - Update the `styleUrls` and `templateUrl`



The DataFile component

- in `data-file.component.ts`
 - rename `dataFiles$` to `dataPackages$`
 - filter by `isDataPackageOfDataFile` instead



The DataFile component

- in `data-file.component.html`
 - Replace `Data Package:` With `Data File:`
 - Remove the `prism.publicationName` field
 - Replace the `dataFiles$` relation with `dataPacakges$` and update the label



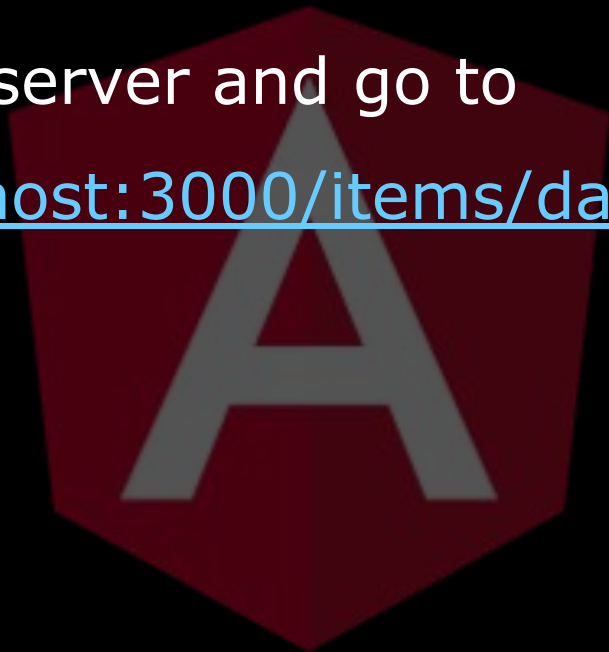
The DataFile component

- In `src/app/+item-page/item-page.module.ts`
 - Add the new component to the `declarations` and `entryComponents` sections.



The DataFile component

- Restart the server and go to <http://localhost:3000/items/datafile1>



Tag: or2019-advanced-3

- To sync your branch with the solution run:
 - `git reset or2019-advanced-start --hard`
 - `git clean -f -d`
 - `git merge or2019-advanced-3`



Debugging

- You can debug using your IDE
 - connect the node debugger to `localhost:5858`
- Or debug using the browser's dev tools
 - disable server side rendering during development:
 - edit (or create) the file `config/environment.dev.js`
 - set `universal.preboot` to `false`



Debugging in the browser

- Source maps ensure you can debug in typescript in the browser.
 - keep in mind that variables can have a slightly different name on a breakpoint
- Useful Chrome shortcut:
 - Fuzzy file search: cmd+O / ctrl+O



Browser Extensions: Augury

- <https://augury.angular.io>
- Component tree
 - see component properties
 - assign a component to a var in the console
 - See dependency injection graphs
- Router tree
 - see route structure as a graph



Browser Extensions: redux-devtools

- <https://github.com/gaearon/redux-devtools>
- See the current state of the store, and the actions that lead to it
- Go forwards and backwards through the actions
- Dispatch custom actions



Learning more

- Learn about Angular, Universal, and other related technologies on the wiki:
<https://tinyurl.com/dspace7-tech-stack>
- Questions? ask on Slack
[#angular-ui on dspace-org.slack.com](https://angular-ui-on-dspace-org.slack.com)





Why a new REST API?

Why a new REST API?

Covers only a
subset of DSpace
functionality

No search

No submit / workflows

Limited admin operations

Limited write / delete

(4.x was read only)

4.x - 6.x



Not based on current
REST best practices
or standards

Handcrafted in Jersey,
while most DSpace code uses
Spring technologies

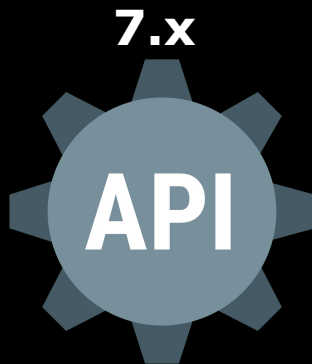


DSPACE

Why a new REST API?

All features **MUST**
be in REST API
(for Angular UI)

Bonus: better third-party
app integration!



Defined **REST Contract**.
HATEOAS, ALPS,
HAL format

Built using Spring technologies
(Spring Boot, MVC, HATEOAS)

<https://github.com/DSpace/DSpace/tree/master/dspace-spring-rest>



DSpace

HATEOAS, HAL, & ALPS, oh my!

HATEOAS = Hypertext As The Engine Of Application State

In each response, include “links” to available next requests.

Results in better decoupling, as API is self-describing.

HAL Format = Hypertext Application Language (JSON or XML)

A standard format for making REST APIs browseable
(think HTML for machines). Open source HAL Browser available.

ALPS = Application Level Profile Semantics*

Describes the operations (actions) available for all REST endpoints.
(Machine-readable) metadata about how to interact with the API.

RESULT: A standards-based, browseable, self-describing REST API

REST Terminology

Stateless (no session)

Specifications

- JSON web tokens ([JWT](#))
- Cross-Origin Resource Sharing ([CORS](#)) Headers

HTTP [Resources](#)

- URIs reference individual [resources](#) or [collections](#) (of resources)
- `/items/{uuid}` and `/items`

Formats: **JSON***

HTTP methods

GET (*read*), HEAD (*read headers only*)
POST (*create*)
PUT (*replace*), PATCH (*partial update*)
DELETE (*remove*)
OPTIONS (*verify access, e.g. via CORS*)

HTTP return codes

2xx (Success)
3xx (Redirect)
4xx (Client error)
5xx (Server error)



Hypermedia as the Engine of Application State - HATEAOS

HAL Format

Links are expressed in a standard/well-known way in the json response (it is also suitable for xml but DSpace7 will focus only on JSON)

- enable the interactive navigation
- abstract routing (URL can change without break the client)
- support documentation

ALPS

Available methods, semantics of request and response objects discoverable in a standard way (/profile) and expressed in multiple standards format (Alps JSON, JSON Schema, XML, etc)

- enable to expose only available methods (i.e. GET on resourcepolicies is disallowed)
- document in a machine-readable way the request and response semantics



Interacting with the REST API

The slides in this section were originally designed by Terry Brady, errors are mine

Interacting with a REST API

You can use the command line...

```
curl "https://dspace7.4science.cloud/dspace-spring-rest/api"
```

But, there are *better* ways... using a REST client

HAL Browser

- Development public instance (provided by 4Science)
 - <https://dspace7.4science.it/dspace-spring-rest/>
- Navigate to the link above to explore





Explorer

Custom Request Headers

Properties

```
{
  "dspaceURL": "https://dspace7-demo.atmire.com",
  "dspaceName": "DSpace7 Demo provided by 4Science",
  "dspaceRest": "https://dspace7.4science.cloud/dspace-spring-rest",
  "type": "root"
}
```

Links

rel	title	name / index	docs	GET	NON-GET
authn					
authorities					
bitstreamformats					
bitstreams					
browses					
claimedtasks					
collections					
communities					
discover					

Inspector

Response Headers

200 200

```
pragma: no-cache
date: Fri, 07 Jun 2019 14:59:53 GMT
x-content-type-options: nosniff
server: Apache/2.4.29 (Ubuntu)
x-frame-options: SAMEORIGIN, DENY
strict-transport-security: max-age=31536000 ; includeSubDomains, max-age=31536000
content-type: application/hal+json;charset=UTF-8
cache-control: no-cache, no-store, max-age=0, must-revalidate
transfer-encoding: chunked
connection: Keep-Alive
keep-alive: timeout=5, max=100
x-xss-protection: 1; mode=block
expires: 0
```

Response Body

```
{
  "dspaceURL": "https://dspace7-demo.atmire.com",
  "dspaceName": "DSpace7 Demo provided by 4Science",
  "dspaceRest": "https://dspace7.4science.cloud/dspace-spring-rest",
  "type": "root",
  "_links": {
    "authn": {
      "href": "https://dspace7.4science.cloud/dspace-spring-rest/api/authn"
    },
    "authorities": {
      "href": "https://dspace7.4science.cloud/dspace-spring-rest/api/integration/authorities"
    },
    "bitstreamformats": {
      "href": "https://dspace7.4science.cloud/dspace-spring-rest/api/core/bitstreamformats"
    },
    "bitstreams": {
      "href": "https://dspace7.4science.cloud/dspace-spring-rest/api/core/bitstreams"
    }
  },
}
```

Request + Headers

Explorer

/dspace-spring-rest/api

Go!

Custom Request Headers



DSpace

Response Headers

Inspector

Response Headers

200 200



HTTP Status code

```
pragma: no-cache
date: Fri, 07 Jun 2019 14:59:53 GMT
x-content-type-options: nosniff
server: Apache/2.4.29 (Ubuntu)
x-frame-options: SAMEORIGIN, DENY
strict-transport-security: max-age=31536000 ; includeSubDomains, max-age=31536000
content-type: application/hal+json;charset=UTF-8
cache-control: no-cache, no-store, max-age=0, must-revalidate
transfer-encoding: chunked
connection: Keep-Alive
keep-alive: timeout=5, max=100
x-xss-protection: 1; mode=block
expires: 0
```



Response Body

Response Body

```
{
  "dspaceURL": "https://dspace7-demo.atmire.com",
  "dspaceName": "DSpace7 Demo provided by 4Science",
  "dspaceRest": "https://dspace7.4science.cloud/dspace-spring-rest",
  "type": "root",
  "_links": {
    "authn": {
      "href": "https://dspace7.4science.cloud/dspace-spring-rest/api/core/authn"
    },
    "authorities": {
      "href": "https://dspace7.4science.cloud/dspace-spring-rest/api/core/authorities"
    },
    "bitstreamformats": {
      "href": "https://dspace7.4science.cloud/dspace-spring-rest/api/core/bitstreamformats"
    },
    "bitstreams": {
      "href": "https://dspace7.4science.cloud/dspace-spring-rest/api/core/bitstreams"
    },
    "browses": {
```

RAW JSON Response

The HAL Browser will parse the key elements:

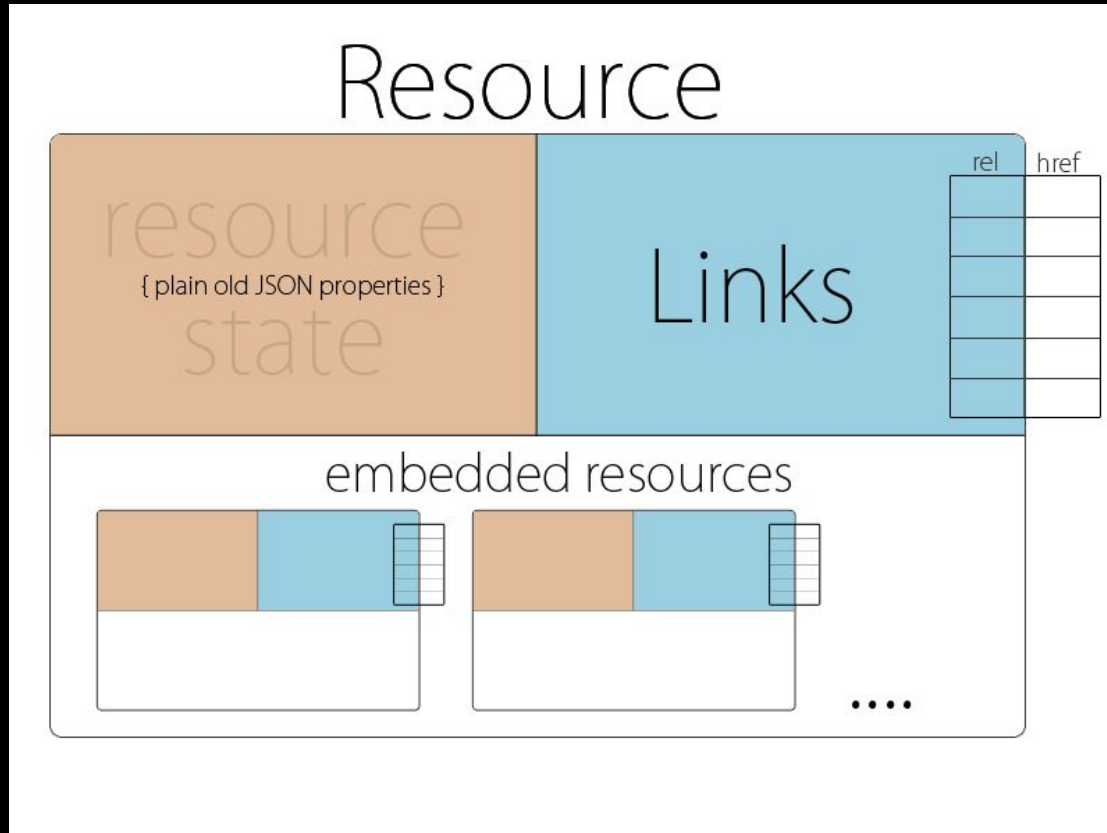
_links

_embedded

Everything else



HAL Document



Response Object Components

- Response Properties
 - Often pagination details
- Links
 - What you can do next
- Embedded Resources
 - List of objects which may contain
 - Properties
 - Links



Response Properties











Response Body

```
{
  "dspaceURL": "https://dspace7-demo.atmire.com",
  "dspaceName": "DSpace7 Demo provided by 4Science",
  "dspaceRest": "https://dspace7.4science.cloud/dspace-spring-rest",
  "type": "root",
  "_links": {
    "authn": {
      "href": "https://dspace7.4science.cloud/dspace-spring-rest/api/authn"
    },
    "authorities": {
      "href": "https://dspace7.4science.cloud/dspace-spring-rest/api/integration/authorities"
    },
    "bitstreamformats": {
      "href": "https://dspace7.4science.cloud/dspace-spring-rest/api/core/bitstreamformats"
    },
    "bitstreams": {
      "href": "https://dspace7.4science.cloud/dspace-spring-rest/api/core/bitstreams"
    },
    "browses": {
```



Links: All endpoints are available from the Entry Point

Links

rel	title	name / index	docs	GET	NON-GET
authn					
authorities					
bitstreamformats					
bitstreams					
browses					
claimedtasks					
collections					
communities					
discover					
dso					



Communities Endpoint Response

Explorer

<https://dspace7.4science.cloud/dspace-spring-rest/api/core/communities>

Go!

Communities
endpoint

Custom Request Headers

Properties

```
{
  "page": {
    "size": 20,
    "totalElements": 87,
    "totalPages": 5,
    "number": 0
  }
}
```

Pagination
properties



DSpace

Links - A generic approach to paginated results

Links

rel	title	name / index	docs	GET	NON-GET
first					
self					
next					
last					
search					



Pagination

<https://github.com/DSpace/Rest7Contract#pagination>

Resource Collections are always paginated

→ pagination information are returned in a consistent way across all the endpoints

→ pagination parameters are expected in the same form in all the endpoints

→ common JAVA API to deal with page jump and offset



Embedded Community Properties

Embedded Resources

[communities\[0\]](#)

[communities\[1\]](#)

[communities\[2\]](#)

Properties

```
{
  "id": "56a74672-fbb5-4911-9f16-b340a67ab587",
  "uuid": "56a74672-fbb5-4911-9f16-b340a67ab587",
  "name": "Botany department",
  "handle": "10673/645",
  "metadata": {
    "dc.description": [
      {
        "value": "Botany...a study of plants",
        "language": null,
        "authority": null,
        "confidence": -1,
        "place": 0
      }
    ],
    "dc.description.abstract": [
      {
        "value": "Botanical garden",
        "language": null,
        "authority": null,
        "confidence": -1,
        "place": 0
      }
    ]
  }
}
```



Embedded Community Links

Links

rel	title	name / index	docs	GET	NON-GET
collections					
logo					
subcommunities					
self					



Right use of the HTTP Verbs: collection

- **POST** Adds a new element to the collection.
- **GET** Returns the first page of the resources in the collection



Right use of the HTTP Verbs: element

GET Returns a single entity.

HEAD Returns whether the item resource is available.

PUT Replaces the state

PATCH Similar to PUT but partially updating the resources state

DELETE Deletes the resource exposed.



ALPS will enable...

In this case, if you executed `curl -H 'Accept:application/schema+json' http://localhost:8080/profile/persons`, you would see something like this:

```
{  
  "title" : "org.springframework.data  
  "properties" : { 2  
    "firstName" : {  
      "readOnly" : false,  
      "type" : "string"  
    },  
    "lastName" : {  
      "readOnly" : false,  
      "type" : "string"  
    },  
    "siblings" : {
```

Create/Update

Person

First name

Last name

Action:

POST

<http://localhost:8080/persons>

Make Request



DSpace

HTTP status codes - 2xx, 3xx

200 Ok - Normal success state

201 Created - Returned when a resource is created

204 No content - Returned when the operation succeed but no content is available (i.e. hit the logo endpoint of a community without logo)

206 Partial Content - DSpace 7 provides range support for bitstream download allowing streaming

302 Found - the PID endpoint redirect to the target resource



HTTP status codes - 4xx

400 Bad Request - if the request is invalid (not a json, etc.)

401 Unauthorized - if the request require a logged-in user

403 Forbidden - if the requester doesn't have enough privilege to execute the request

404 Not found - if the requested resource doesn't exists

405 Method Not Allowed - if the requested verb is not supported for the resource

422 Unprocessable Entity - if the request is semantically invalid (i.e. attempt to add undefined metadata, deposit an invalid workspace)



REST Maturity level

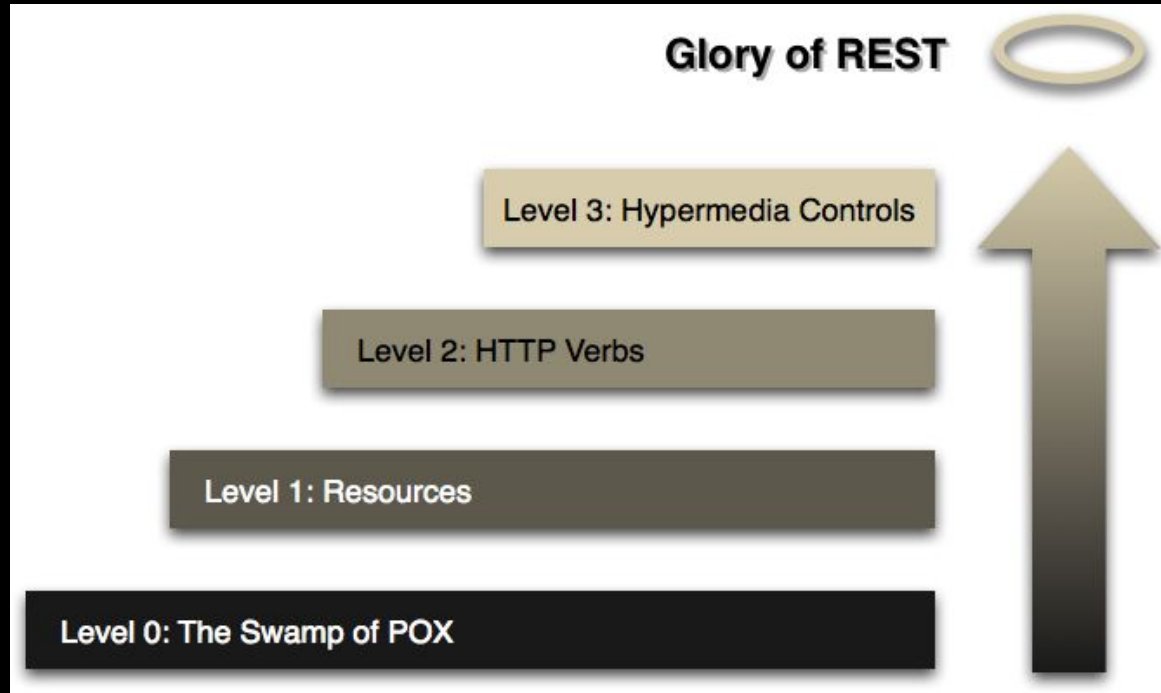


Image from: <https://martinfowler.com/articles/richardsonMaturityModel.html>



DSpace

The REST contract

- <https://github.com/DSpace/Rest7Contract>
 - Explore the list of endpoints
 - Implemented
 - Unimplemented
 - Under discussion
 - Use GitHub pull requests to suggest changes



The future of REST API Documentation

/dspace7-rest/api





































Go!

Custom Request Headers

Properties

{ }

Links

rel	title	name / index	docs	GET	NON-GET
c:bitstreamformats					
c:bitstreams					
d:browses					
c:collections					
c:communities					
p:persons					
p:groups					
c:items					
c:metadatafields					
c:metadataschemas					
c:sites					
c:users					

Item list REST endpoint

The item endpoint lists all the items within the repository with pagination.

Response structure

Path	Type	Description
_embedded	Object	The items that are in the repository, displayed with pagination
_links	Object	The links that are displayed on this page, mostly with regards to pagination if applicable
page	Object	A page object that contains information about the amount of items shown and in total

Links

Relation	Description
first	Link to the first page of items
self	Link to this page
next	Link to the next page
last	Link to the last page

[PR#1915](#)



DSpace

Exercise 1: Explore HAL Browser

Exercise 1 - Exploring Endpoints in HAL Browser

Short Link To Exercises

<http://bit.ly/dspace7-rest>



DSPACE

REST Authentication

- The existing DSpace authentication methods will be supported
- The REST API returns an authentication token
- This token is passed to subsequent REST requests as a header



JWT Token

A JWT Token contains user data “signed” by the Server so that they cannot be modified

<https://jwt.io/>



ALGORITHM

HS256



Encoded

PASTE A TOKEN HERE

```
eyJhbGciOiJIUzI1NiJ9.eyJlaWQiOiIzMzU2NDd  
iNi04YTUyLTRlY2ItYThjMS03ZWJhYmIxOTliZGE  
iLCJzZyI6W10sImV4cCI6MTU1OTkxMzg3M30.ajU  
M3LrmijAs5thAH-k48ICZ5wmY6UCiLbo2-K4g8NA
```

Decoded

EDIT THE PAYLOAD AND SECRET

HEADER: ALGORITHM & TOKEN TYPE

```
{  
  "alg": "HS256"  
}
```

PAYLOAD: DATA

```
{  
  "eid": "335647b6-8a52-4ecb-a8c1-7ebabb199bda",  
  "sg": [],  
  "exp": 1559913813  
}
```

VERIFY SIGNATURE

```
HMACSHA256(  
  base64UrlEncode(header) + "." +  
  base64UrlEncode(payload),  
  your-256-bit-secret  
) ☐ secret base64 encoded
```

eid → EPerson uuid

sg → Special group

exp → Expiration
time

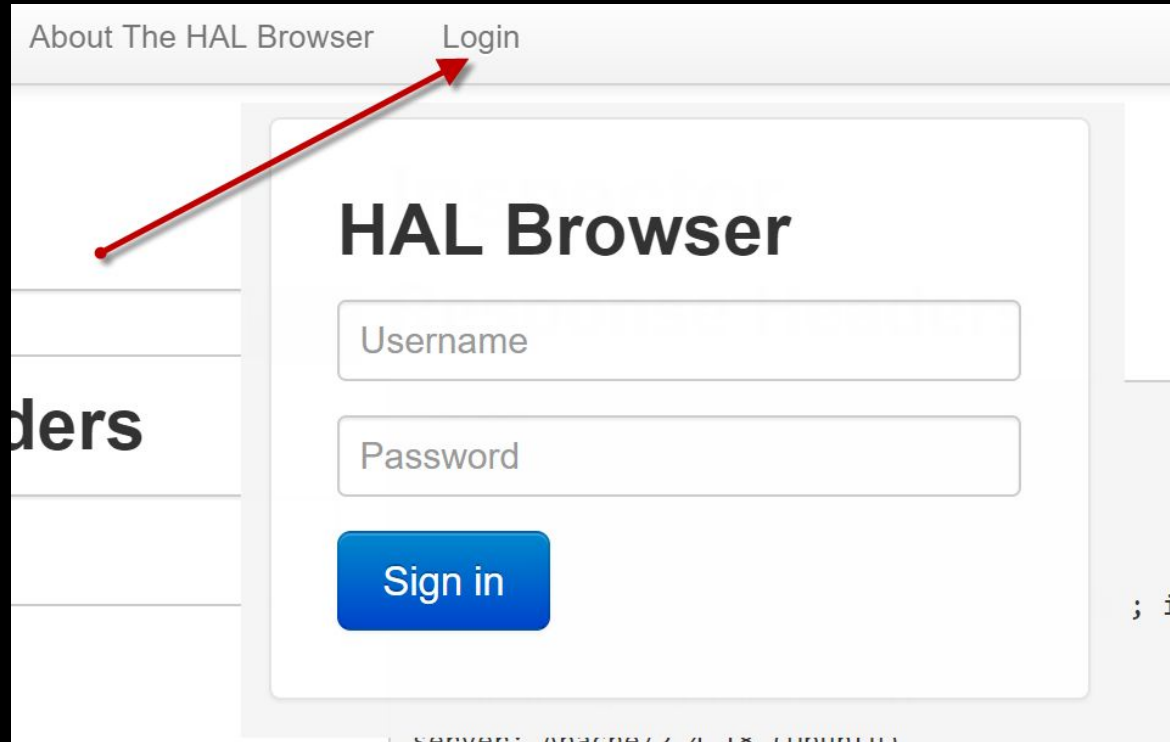
Why Authenticate?

- View restricted items/bitstreams
- Create/Update/Delete Objects
 - Submit items
- Perform admin operations*
- View restricted metadata (provenance)*

Note: some of these actions are not yet supported



A subset of these operations are available in HAL



The screenshot shows a web interface for the HAL Browser. At the top, there are two links: "About The HAL Browser" and "Login". A red arrow points from the "Login" link to the "HAL Browser" title of the main content area. The main content area contains a login form with two input fields: "Username" and "Password", and a blue "Sign in" button. The page is partially obscured by a sidebar on the left with the word "ders" visible.

About The HAL Browser Login

HAL Browser

Username

Password

Sign in



Exercise 2: Authenticating in HAL

Exercise 2 - Authentication in HAL Browser

- For this workshop, we will use password authentication

<http://bit.ly/dspace7-rest>



Limitations of HAL AuthN

- Special chars not handled correctly ([DS-4259](#))*
- Multiple AuthN providers are not supported ([DS-3814](#))
- Credentials only passed for GET operations
- File uploads not supported

* it will be merged in the next few days



Postman - A development client for REST API's

<https://www.getpostman.com/downloads/>



Postman - A tool for sending REST requests

- Postman is a tool for interacting with various REST services
 - Even ones without a HAL Browser
- Using a web browser, it is difficult to construct complex requests



Postman

The screenshot displays the Postman application interface. At the top, there is a menu bar with 'File', 'Edit', 'View', and 'Help'. Below the menu bar, a toolbar contains buttons for 'New', 'Import', 'Runner', and a settings icon. The main interface is divided into several sections:

- Left Sidebar:** Contains a 'Filter' search bar, 'History', 'Collections' (with a '+ New Collection' button), and a 'Trash' section. A list of collections is shown, including 'DSpace 7 Tutorial' (11 requests) and various API endpoints like 'API Entry point', 'Metadata Schema Registry', 'Metadata Fields Registry', 'Browses', 'Bitstream Formats Registry', 'Communities list', 'Collections list', 'Items list', 'EPersons list', 'Not existing endpoint', and 'Authentication Status'.
- Top Bar:** Displays the current collection 'DSpace 7 Tutorial' and an 'Invite' button. On the right, there are icons for a refresh, share, sync, notifications, heart, and a user profile, along with an 'Upgrade' button.
- Request Editor:** The main area for editing the request. It shows a 'GET' request to the endpoint '((dSPACEresturl))/api'. Below the URL bar, there are tabs for 'Params', 'Authorization', 'Headers (7)', 'Body', 'Pre-request Script', and 'Tests'. The 'Params' tab is active, showing a table of query parameters.
- Query Params Table:**

KEY	VALUE	DESCRIPTION
Key	Value	Description
- Body Tab:** The 'Body' tab is selected, showing the request body in JSON format. The status bar indicates 'Status: 200 200', 'Time: 394 ms', and 'Size: 4.19 KB'. There are 'Save' and 'Download' buttons.
- Response View:** The response is displayed in the bottom section, showing a JSON object with various API endpoints and their descriptions. The response is formatted in 'Pretty' view.

Collections, Tabs, Workspace and Environments help you organize and re-use requests

The screenshot displays the DSpace 7 Tutorial interface. On the left, a sidebar titled 'Collections' lists various request collections, including 'DSpace 7 Tutorial' (11 requests) and 'API Entry point'. The main area shows the 'API Entry point' collection selected, with a 'GET' method and a URL template '({dSPACEURL})/api'. The 'Query Params' section is empty. The 'Body' tab is active, showing a JSON response. The 'Pretty' view is selected, displaying the JSON structure with line numbers. The response includes fields like 'dSPACEURL', 'dSPACEName', 'dSPACERest', 'type', 'links', 'authn', 'authorities', 'bitstreamformats', 'bitstreams', 'browse', 'clainedtasks', 'collections', 'communities', 'discover', 'dso', 'entitytypes', 'epersons', 'filtered-discovery-pages', and 'groups'.

File Edit View Help

New Import Runner

DSpace 7 Tutorial Invite

Filter

History Collections

+ New Collection

DSpace 7 Tutorial

11 requests

GET API Entry point

GET Metadata Schema Registry

GET Metadata Fields Registry

GET Browse

GET Bitstream Formats Registry

GET Communities list

GET Collections list

GET Items list

GET EPersons list

GET Not existing endpoint

GET Authentication Status

API Entry point

GET ((dSPACEURL))/api

Send Save

Params Authorization Headers (7) Body Pre-request Script Tests

Query Params

KEY	VALUE	DESCRIPTION
Key	Value	Description

Body Cookies Headers (15) Test Results

Status: 200 200 Time: 394 ms Size: 4.19 KB Save Download

Pretty Raw Preview JSON

```
1 {
2   "dSPACEURL": "https://dspace7-demo.atnire.com",
3   "dSPACEName": "DSpace7 Demo provided by 4Science",
4   "dSPACERest": "https://dspace7.4science.cloud/dspace-spring-rest",
5   "type": "root",
6   "links": {
7     "authn": {
8       "href": "https://dspace7.4science.cloud/dspace-spring-rest/api/authn"
9     },
10    "authorities": {
11      "href": "https://dspace7.4science.cloud/dspace-spring-rest/api/integration/authorities"
12    },
13    "bitstreamformats": {
14      "href": "https://dspace7.4science.cloud/dspace-spring-rest/api/core/bitstreamformats"
15    },
16    "bitstreams": {
17      "href": "https://dspace7.4science.cloud/dspace-spring-rest/api/core/bitstreams"
18    },
19    "browse": {
20      "href": "https://dspace7.4science.cloud/dspace-spring-rest/api/discover/browse"
21    },
22    "clainedtasks": {
23      "href": "https://dspace7.4science.cloud/dspace-spring-rest/api/workflow/clainedtasks"
24    },
25    "collections": {
26      "href": "https://dspace7.4science.cloud/dspace-spring-rest/api/core/collections"
27    },
28    "communities": {
29      "href": "https://dspace7.4science.cloud/dspace-spring-rest/api/core/communities"
30    },
31    "discover": {
32      "href": "https://dspace7.4science.cloud/dspace-spring-rest/api/discover"
33    },
34    "dso": {
35      "href": "https://dspace7.4science.cloud/dspace-spring-rest/api/dso/find?{uid}",
36      "templated": true
37    },
38    "entitytypes": {
39      "href": "https://dspace7.4science.cloud/dspace-spring-rest/api/core/entitytypes"
40    },
41    "epersons": {
42      "href": "https://dspace7.4science.cloud/dspace-spring-rest/api/eperson/epersons"
43    },
44    "filtered-discovery-pages": {
45      "href": "https://dspace7.4science.cloud/dspace-spring-rest/api/config/filtered-discovery-pages"
46    },
47    "groups": {
```



Collections, Tabs, **Workspace** and Environments help you organize and re-use requests

The screenshot displays the DSpace 7 Tutorial interface. On the left, a sidebar shows a collection named "DSpace 7 Tutorial" containing 11 requests. The main panel shows the "API Entry point" request selected, with tabs for Params, Authorization, Headers (7), Body, and Preview. The "Body" tab is active, showing a JSON response. A dropdown menu is open, showing the "Personal" tab with a list of workspaces: "DSpace 7 Tutorial" (checked) and "My Workspace".

File Edit View Help

New Import Runner

DSpace 7 Tutorial Invite

Filter

History Collections APIs BETA

+ New Collection Trash

DSpace 7 Tutorial
11 requests

- GET API Entry point
- GET Metadata Schema Registry
- GET Metadata Fields Registry
- GET Browsers
- GET Bitstream Formats Registry
- GET Communities list
- GET Collections list
- GET Items list
- GET EPersons list
- GET Not existing endpoint
- GET Authentication Status

GET API Entry point X GET Communities list

API Entry point

GET {{(dSPACEresturl)}/api}

Params Authorization Headers (7) Body Preview

Query Params

KEY	VALUE
Key	

Body Cookies Headers (15) Test Results

Pretty Raw Preview JSON

```
1 {
2   "dSPACEURL": "https://dspace7-demo.atmire.com",
3   "dSPACEName": "DSpace7 Demo provided by 4Science",
4   "dSPACERest": "https://dspace7.4science.cloud/dspace-spring-rest",
5   "type": "root",
6   "links": {
7     "authn": {
8       "href": "https://dspace7.4science.cloud/dspace-spring-rest/api/authn"
9     },
10    "authorities": {
```



Collections, Tabs, Workspace and Environments help you organize and re-use requests

DSpace 7 Tutorial

Invite

GET Communities list

GET Not existing endpoint

GET Browsers

GET Metadata Schema Registry

GET Metadata Fields Registry

+

...

Official DSpace 7 Demo

Official DSpace 7 Demo

Edit

VARIABLE	INITIAL VALUE	CURRENT VALUE
dspaceresturl	https://dspace7.4science.cloud/dspace-spring-rest	https://dspace7.4science.cloud/dspace-spring-rest

Globals

Edit

No global variables

Global variables are a set of variables that are always available in a workspace.

Learn more about globals

Use variables to reuse values in different places. The current value is used while sending a request and is never synced to Postman's servers. The initial value is auto-updated to reflect the current value. Change this behaviour from Settings. Learn more about variable values

X

GET Metadata Schema Registry

GET Metadata Fields Registry

+

...

Official DSpace 7 Demo

Official DSpace 7 Demo

Edit

VARIABLE	INITIAL VALUE	CURRENT VALUE
dspaceresturl	https://dspace7.4science.cloud/dspace-spring-rest	https://dspace7.4science.cloud/dspace-spring-rest

Globals

Edit

No global variables

Global variables are a set of variables that are always available in a workspace.

Learn more about globals

Use variables to reuse values in different places. The current value is used while sending a request and is never synced to Postman's servers. The initial value is auto-updated to reflect the current value. Change this behaviour from Settings. Learn more about variable values

X



DSPACE

Select the right
HTTP Method

Request Area

Endpoint URL (can contains
variable from the Env)

Tabs allow access to more
settings (parameters, auth,
content-type, request body)

The screenshot shows the DSpace API client interface. The top bar includes a dropdown menu for 'DSpace 7 Demo (Admin)' and a 'Send' button. The main area is divided into tabs: 'API Entry point', 'Params', 'Authorization', 'Headers (8)', 'Body', 'Pre-request Script', and 'Tests'. The 'API Entry point' tab is active, showing a 'GET' method and an endpoint URL '{(dSPACEURL)}/api'. The 'Params' tab is also visible, showing a table with 'KEY' and 'VALUE' columns. The 'Body' tab is active, showing a JSON response. The JSON response is displayed in a 'Pretty' view, showing a list of API endpoints and their descriptions. The interface also includes a 'Status' bar at the bottom right showing 'Status: 200 200', 'Time: 95 ms', and 'Size: 4.16 KB'. There are also buttons for 'Save' and 'Download'.

API Entry point

GET {(dSPACEURL)}/api

Params Authorization Headers (8) Body Pre-request Script Tests

Query

KEY	VALUE	DESCRIPTION
Key	Value	Description

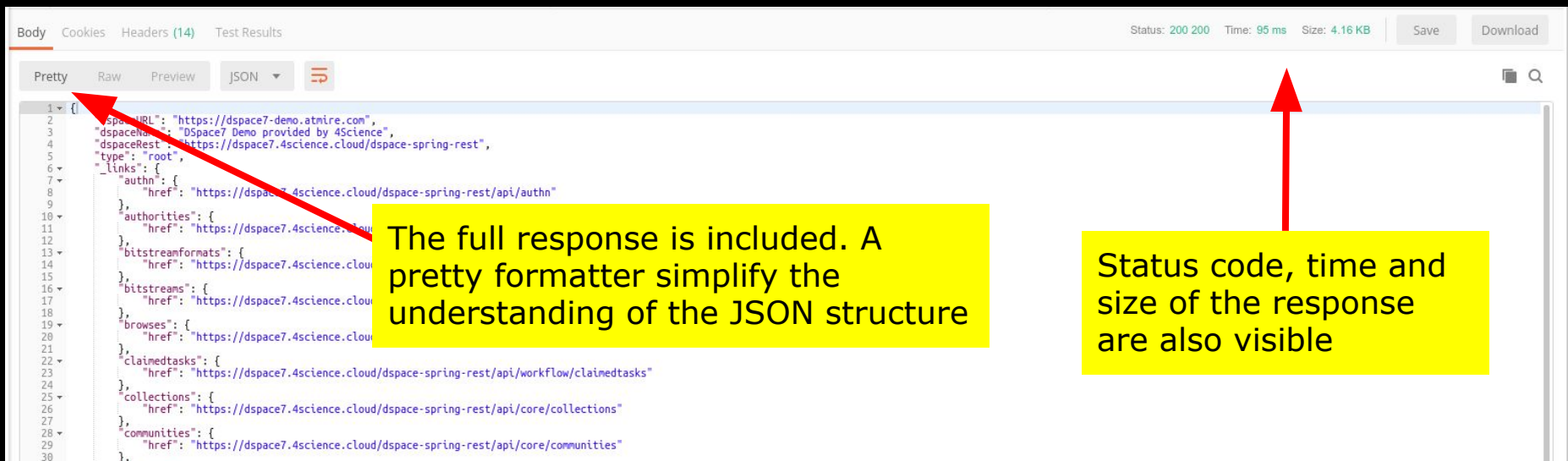
Body Cookies Headers (14) Test Results

Status: 200 200 Time: 95 ms Size: 4.16 KB Save Download

```
{
  "dSPACEURL": "https://dspace7-demo.atnre.com",
  "dSPACEName": "DSpace7 Demo provided by 4Science",
  "dSPACERest": "https://dspace7.4science.cloud/dspace-",
  "type": "root",
  "_links": {
    "authn": {
      "href": "https://dspace7.4science.cloud/dspace-spring-rest/api/core/authn"
    },
    "authorities": {
      "href": "https://dspace7.4science.cloud/dspace-spring-rest/api/core/authorities"
    },
    "bitstreamformats": {
      "href": "https://dspace7.4science.cloud/dspace-spring-rest/api/core/bitstreamformats"
    },
    "bitstreams": {
      "href": "https://dspace7.4science.cloud/dspace-spring-rest/api/core/bitstreams"
    },
    "browses": {
      "href": "https://dspace7.4science.cloud/dspace-spring-rest/api/discover/browses"
    },
    "claimedtasks": {
      "href": "https://dspace7.4science.cloud/dspace-spring-rest/api/workflow/claimedtasks"
    },
    "collections": {
      "href": "https://dspace7.4science.cloud/dspace-spring-rest/api/core/collections"
    },
    "communities": {
      "href": "https://dspace7.4science.cloud/dspace-spring-rest/api/core/communities"
    },
    "discover": {
      "href": "https://dspace7.4science.cloud/dspace-spring-rest/api/discover"
    },
    "dso": {
      "href": "https://dspace7.4science.cloud/dspace-spring-rest/api/dso/find?uid",
      "templated": true
    },
    "entitytypes": {
      "href": "https://dspace7.4science.cloud/dspace-spring-rest/api/core/entitytypes"
    },
    "persons": {
      "href": "https://dspace7.4science.cloud/dspace-spring-rest/api/core/persons"
    }
  }
}
```



Response Panel



The screenshot shows the 'Response' panel in a web browser's developer tools. The 'Pretty' button is selected, and the JSON response is displayed in a formatted manner. The status bar at the top right shows 'Status: 200 200', 'Time: 95 ms', and 'Size: 4.16 KB'. Two yellow callout boxes provide additional information:

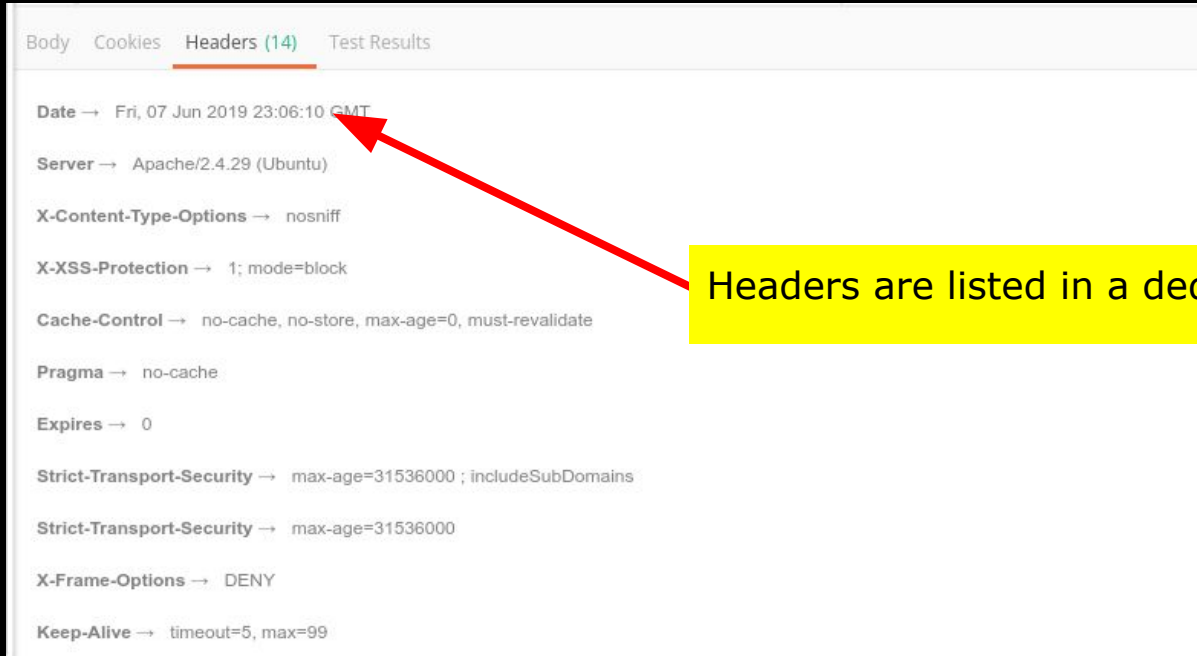
The full response is included. A pretty formatter simplify the understanding of the JSON structure

Status code, time and size of the response are also visible

```
1 {
2   "dspaceURL": "https://dspace7-demo.atmire.com",
3   "dspaceName": "DSpace7 Demo provided by 4Science",
4   "dspaceRest": "https://dspace7.4science.cloud/dspace-spring-rest",
5   "type": "root",
6   "_links": {
7     "authn": {
8       "href": "https://dspace7.4science.cloud/dspace-spring-rest/api/authn"
9     },
10    "authorities": {
11      "href": "https://dspace7.4science.cloud/dspace-spring-rest/api/authorities"
12    },
13    "bitstreamformats": {
14      "href": "https://dspace7.4science.cloud/dspace-spring-rest/api/core/bitstreamformats"
15    },
16    "bitstreams": {
17      "href": "https://dspace7.4science.cloud/dspace-spring-rest/api/core/bitstreams"
18    },
19    "browses": {
20      "href": "https://dspace7.4science.cloud/dspace-spring-rest/api/core/browses"
21    },
22    "claimedtasks": {
23      "href": "https://dspace7.4science.cloud/dspace-spring-rest/api/workflow/claimedtasks"
24    },
25    "collections": {
26      "href": "https://dspace7.4science.cloud/dspace-spring-rest/api/core/collections"
27    },
28    "communities": {
29      "href": "https://dspace7.4science.cloud/dspace-spring-rest/api/core/communities"
30    }
31  }
```



Response Headers



The screenshot shows the 'Headers' tab in a web browser's developer tools. The 'Date' header is highlighted with a red arrow pointing to a yellow callout box. The 'Date' header value is 'Fri, 07 Jun 2019 23:06:10 GMT'. Other headers listed include 'Server', 'X-Content-Type-Options', 'X-XSS-Protection', 'Cache-Control', 'Pragma', 'Expires', 'Strict-Transport-Security', 'X-Frame-Options', and 'Keep-Alive'.

Header	Value
Date	Fri, 07 Jun 2019 23:06:10 GMT
Server	Apache/2.4.29 (Ubuntu)
X-Content-Type-Options	nosniff
X-XSS-Protection	1; mode=block
Cache-Control	no-cache, no-store, max-age=0, must-revalidate
Pragma	no-cache
Expires	0
Strict-Transport-Security	max-age=31536000 ; includeSubDomains
Strict-Transport-Security	max-age=31536000
X-Frame-Options	DENY
Keep-Alive	timeout=5, max=99

Headers are listed in a dedicated tab



Browsing with Postman

- View all communities
 - `/api/core/communities`
- Change page size
 - `/api/core/communities?size=5`
- Change starting page
 - `/api/core/communities?size=5&page=2`



Exercise 3: Browsing with Postman

Exercise 3 - Browsing with Postman

<http://bit.ly/dspace7-rest>



DSpace

Postman - Authenticating as a User

AuthN Status in Postman (no AuthN token)

► Status

Examples (0) ▼

GET

{{dspaceresturl}}/api/authn/status

Send

Save

Params

Authorization

Headers (7)

Body

Pre-request Script

Tests

Cookies Code Comments (0)

Query Params

KEY	VALUE	DESCRIPTION	...	Bulk Edit
Key	Value	Description		

Body

Cookies

Headers (14)

Test Results

Status: 200 200

Time: 75 ms

Size: 765 B

Save

Download

Pretty

Raw

Preview

JSON



```
1 {
2   "id": null,
3   "okay": true,
4   "authenticated": false,
5   "type": "status",
6   "_links": {
7     "self": {
8       "href": "https://dspace7.4science.cloud/dspace-spring-rest/api/authn/status"
9     }
10  },
11  "_embedded": {
12    "eperson": null
13  }
14 }
```

authenticated: false

Authenticating in Postman

POST x-www-form-urlencoded

User credentials

Authentication Token

AuthN Status in Postman (passing Bearer token)

GET `{{dspaceurl}}/api/authn/status`

Send

Save

Params Authorization Headers (8) Body Pre-request Script Tests

Cookies Code Comments (0)

TYPE

Bearer Token

Token

eyJhbGciOiJIUzI1NiJ9.eyJlaWQiOiIzMTU2NDdiNi04YTUyLTRYIiwiaXNjaWkiOiJMS03ZWJhYmIiLCJzZyI6IjE0smV4cCMTU1OTk1...

The authorization header will be automatically generated when you send the request. [Learn more about authorization](#)

Preview Request

Authentication Token retrieved from the login endpoint

Body Cookies Headers (14) Test Results

Status: 200 200 Time: 177 ms Size: 13.56 KB

Save

Download

Pretty Raw Preview

JSON

IE

authenticated: true

```
1 {
2   "id": null,
3   "okay": true,
4   "authenticated": true,
5   "type": "status",
6   "_links": {
7     "eperson": {
8       "href": "https://dspace7.4science.cloud/dspace-spring-rest/api/eperson/epersons/335647b6-8a52-4ecb-a8c1-7ebabb199bda"
9     },
10    "self": {
11      "href": "https://dspace7.4science.cloud/dspace-spring-rest/api/authn/status"
12    }
13  },
14  "embedded": {
15    "eperson": {
16      "id": "335647b6-8a52-4ecb-a8c1-7ebabb199bda",
17      "uuid": "335647b6-8a52-4ecb-a8c1-7ebabb199bda",
18      "name": "dspaceadmin@gmail.com",
19      "handle": null,
20      "metadata": {
21        "eperson.firstname": [
22          {
23            "value": "Demo",
24            "language": null,
25            "authority": null,
26            "confidence": -1,
27            "place": 0
28          }
29        ],
30        "eperson.language": [
31          {
32            "value": "en",
33            "language": null,
34            "authority": null,
35            "confidence": -1,
36            "place": 0
37          }
38        ]
39      }
40    }
41  }
42 }
```


Let's Attempt to change data

- The current REST API allows a user to start a submission
- POST
 - /api/submission/workspaceitems
 - Body: {}
- The response will return an object with an id



Retrieving the Item that was created

- GET
 - /api/submission/workspaceitems/[id]
- DELETE
 - /api/submission/workspaceitems/[id]
- GET
 - /api/submission/workspaceitems/[id]

The second GET request will fail



Exercise 4: Authenticating with Postman

Exercise 4 - Authenticating with Postman

<http://bit.ly/dspace7-rest>



DSpace

How to deal with PATCH

[JSON Patch specification RFC6902](#)

Express change to a JSON Object in JSON Array of Operations executed in an atomic transaction

Each successful Patch operation will return a HTTP 200 CODE with the new state of the patched resource as body similar to what is returned for a GET request.



How to deal with PATCH

ADD / REMOVE / REPLACE / MOVE

```
[  
  { "op": "test", "path": "/a/b/c", "value": "foo" },  
  { "op": "remove", "path": "/a/b/c" },  
  { "op": "add", "path": "/a/b/c", "value": [ "foo", "bar" ] },  
  { "op": "replace", "path": "/a/b/c", "value": 42 },  
  { "op": "move", "from": "/a/b/c", "path": "/a/b/d" },  
  { "op": "copy", "from": "/a/b/d", "path": "/a/b/e" }  
]
```

TEST & COPY (no plan to support them)



```
{
  id: 1,
  type: "workspaceitem",
  sections:
  {
    "traditionalpageone": {
      "dc.title": [
        {
          "value": "Sample Submission Item",
          "language": "en_US",
          "authority": null,
          "confidence": -1,
          "place": 0
        }
      ],
      "dc.contributor.author": [
        {
          "value": "Bollini, Andrea",
          "language": null,
          "authority": "rp00001",
          "confidence": 600,
          "place": 0
        }
      ]
    }
  },
  ...
},
```



```
{
  id: 1,
  type: "workspaceitem",
  sections:
  {
    "traditionalpageone": {
      "dc.title": [
        {
          "value": "Sample Submission Item",
          "language": "en_US",
          "authority": null,
          "confidence": -1,
          "place": 0
        }
      ],
      "dc.contributor.author": [
        {
          "value": "Bollini, Andrea",
          "language": null,
          "authority": "rp00001",
          "confidence": 600,
          "place": 0
        },
        {
          "value": "Another, Author",
          "language": null,
          "authority": null,
          "confidence": -1,
          "place": 1
        }
      ]
    }
  }
}
```

PATCH (More examples: <https://goo.gl/G84oRQ>)

```
{
  "op": "add",
  "path": "/sections/traditionalpageone/dc.contributor.author/-",
  "value": {"value": "Another, Author"}
}
```

Exercise 5: Interact with the submission

Exercise 5 - Interact with the submission

<http://bit.ly/dspace7-rest>



Managing Postman

- Postman allows you to use variables and scripts to manage credentials
- Environments allow you to switch easily from one installation: test, staging, prod (!) to another - can be also used to switch between user

[Tips on using postman](#)



A look to the JAVA backend

Spring technologies

Spring Bootstrap: pre-configured managed Spring platform

Spring MVC: Front Controller, data binding, error handling

Spring REST: MVC extension to easily support Content-Negotiation, Response in JSON format



Spring technologies

Spring HATEOAS: library to deal with HAL Document (Resources, Resource, Links, Curie)

Spring Data REST: only for inspiration (consistent design choices, HTTP error handling, support classes, etc.)



Spring technologies

Spring Data REST Hal Browser: WebJar of the HAL browser customized to work at best with Spring Data REST

Spring REST Docs: Self contained documentation with snippets from Integration Tests



How URL are translated to JAVA code

Where is my web.xml?

It is a spring boot application with web support (MVC)

Controller → are the “new” Servlet
@RestController



What about the lovable spring xml files?

sorry now we use annotations :)

Component scanning enabled on
repository, converter, utils packages

@Component

@Service



Walkthrough the EPerson endpoint

org.dspace.app.rest.repository.EPersonRestRepository

- List EPersons → findAll
- Get EPerson → findOne

Rest Model Class

[org.dspace.app.rest.model.EPersonRest](#)

- POJO used as DTO between the REST layer and the services.
- Clean and stable version of our data model
- The converter class transcode the DSpace-API objects in REST objects



Repository Class

- It is defined in the [Repository design pattern](#)
- It implements the Data Access Layer in DDD
- Close to the domain model, abstracting access to a collection



Resource Class

[org.dspace.app.rest.model.hateoas.EPersonResource](#)

- Wrap the Domain Class inside an HAL document
- Provide supports to add links
- Manage embedded objects



Integration Test

org.dspace.app.rest.EPersonRestRepositoryIT

Builder and Matcher helper classes to keep code compact & cleanup the test database between runs

One or more test for each repository methods

- Separate test for different scenarios (exception, limit case, normal case)
- Multiple calls in the same test to cross check the intent changes



@PreAuthorize

[org.dspace.app.rest.security.EPersonRestPermissionEvaluatorPlugin](#)

- Security checks isolated in dedicated classes, more cohesive and reusable
- Pluggable implementations to approve incoming requests



Walkthrough - Search Methods

<https://goo.gl/NSju3q>

Collection endpoints that allowing filtering, search or precise lookup of resources MUST provide a search link listing all the available methods

- findByName
- findByEmail



Contributing Back



How can I help, technically?

Join Community Sprints (*coming again soon*)

Join DSpace 7 Working Group

- Next meeting: Thurs, June 20 at 14UTC

Join DSpace 7 Entities Working Group

- Next meeting, Tues, June 18 at 15UTC



More info at OR2019

- ★ DSpace 7 - Creating High Quality Software: Update to Development Practices (Developer Track)
 - **Weds, 9:00am (Lecture Hall M)**
 - Andrea Bollini, Terry Brady, Giuseppe Digilio, Tim Donohue



How can I help, non-technically?

Join [DSpace 7 Marketing Working Group!](#)

- Marketing both DSpace 7 and DSpace (in general). *Thank them for t-shirts & buttons!*

Join [DSpace Community Advisory Team \(DCAT\)](#)

- Repository Manager interest group
- Advises on community needs



Help by becoming a member!

*DSpace is funded / developed / supported
by its community.*

- ★ Become a member and influence product roadmap, governance and member benefits.
- ★ Membership also funds coordination (Tech Coordinator, Product Coordinator?)



Try it out now!

DSpace 7 UI demo

<https://dspace7-demo.atmire.com/>

(uses the REST API demo as backend)

DSpace 7 REST API demo:

<https://dspace7.4science.cloud/dspace-spring-rest/>

Run locally via Docker:

<https://dspace-labs.github.io/DSpace-Docker-Images/>



Questions?

Slides available at

<https://tinyurl.com/or2019-dspace7-advanced>