



DEVELOPERS LIVE

Headless GraphQL with Content Fragments

Jabran Asghar | Senior Software Engineer



Agenda

Content delivery & management

Content Fragment Models & GraphQL types

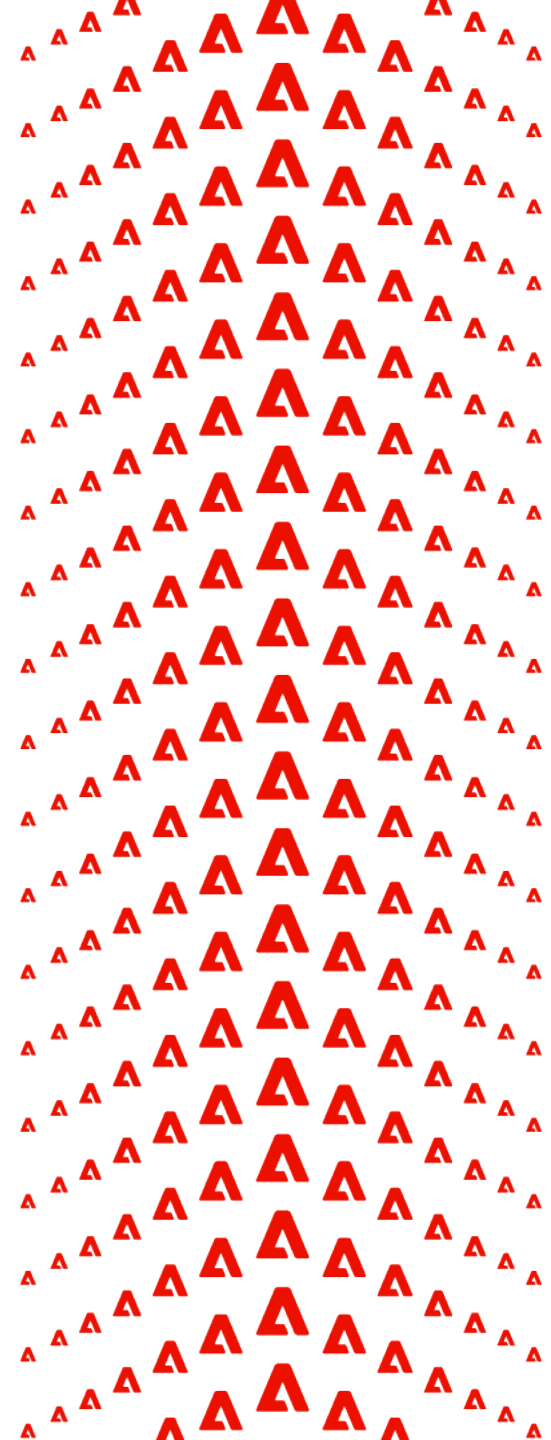
Content Fragment Modeling enhancements

Content Fragment editor enhancements

Headless – AEM GraphQL API Demo

Wrap up

What's next?

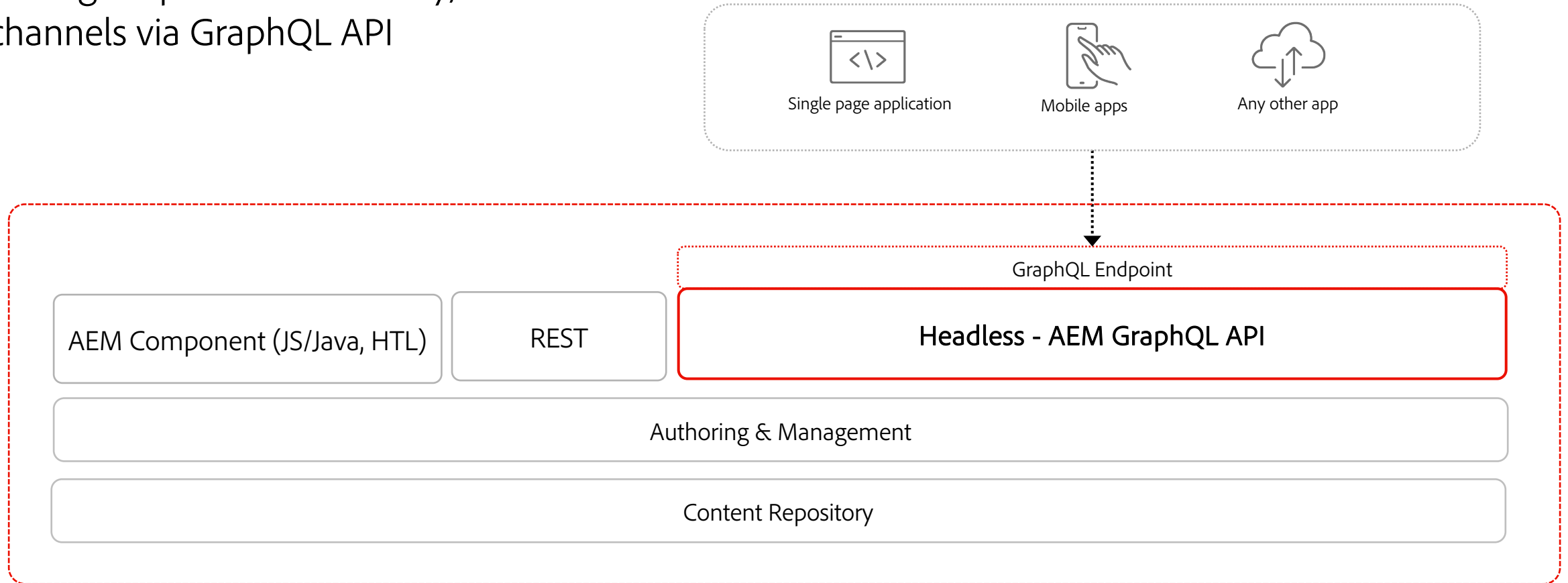


AEM content delivery & management



AEM: Content Delivery & Management

Manage experiences centrally, deliver across channels via GraphQL API



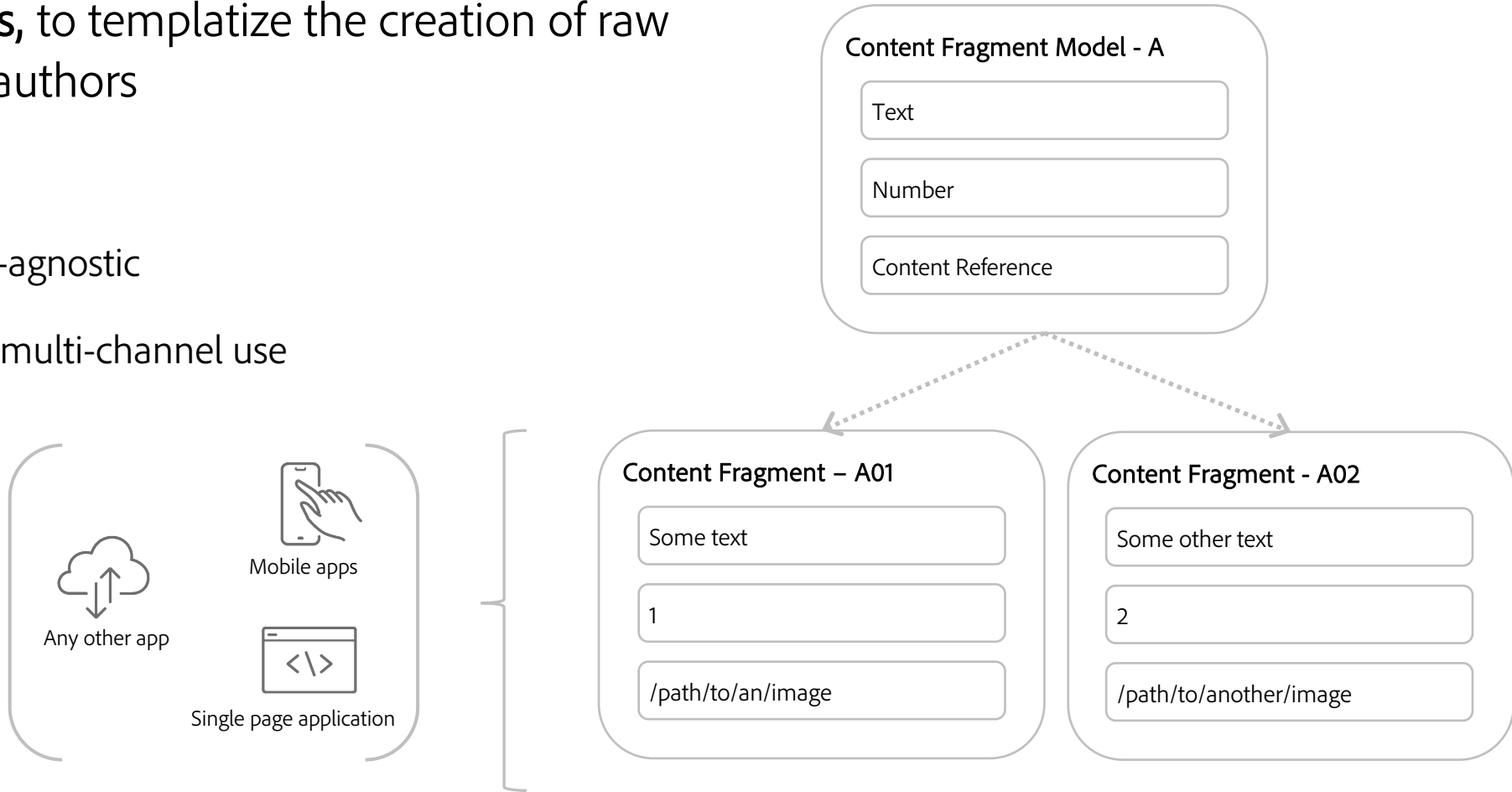
Content Fragment Models & GraphQL types



Content Fragment Models: What?

Content schemas, to templatize the creation of raw content by AEM authors

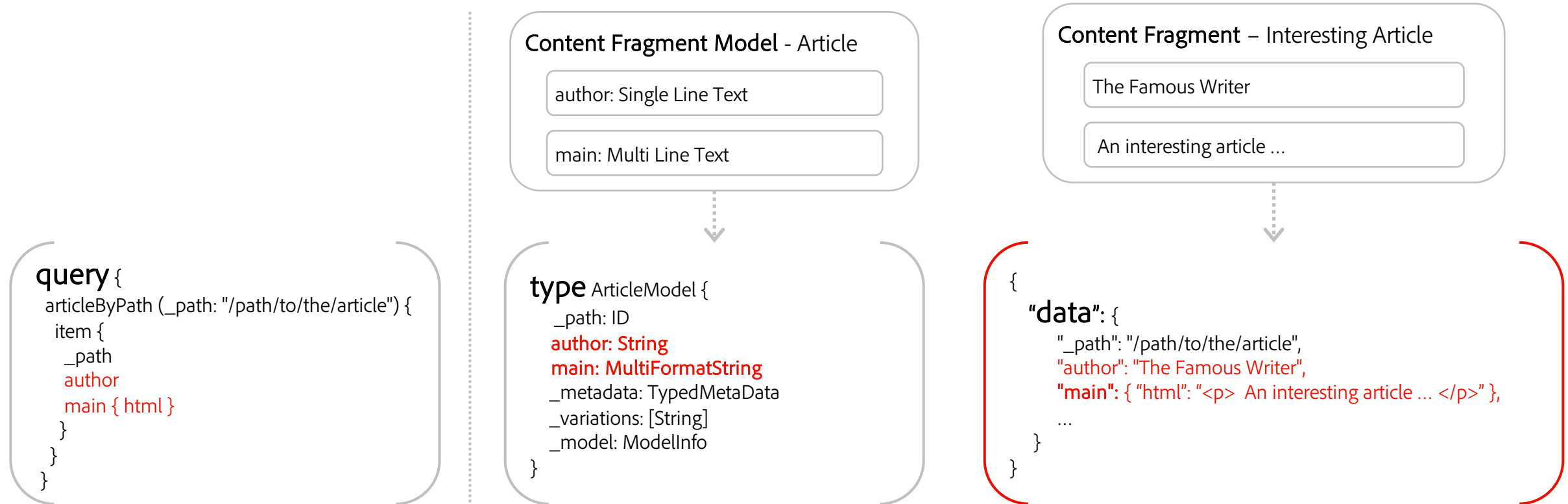
- Structured
- Presentation-agnostic
- Intended for multi-channel use



Content Fragment Models: GraphQL Types

AEM APIs for headless capabilities converts Content Fragment Models to GraphQL types

- Content Fragments are retrieved via GraphQL queries, in JSON format



AEM headless: GraphQL query features

Supported GraphQL features

- Aliases
- Arguments
- Fragments
- Input types
- Introspection
- Meta fields
- Nested fragments
- Union types
- Variables

Not supported (yet)

- Mutations
- Subscriptions
- Paging/Sorting (for GraphQL JSON results)

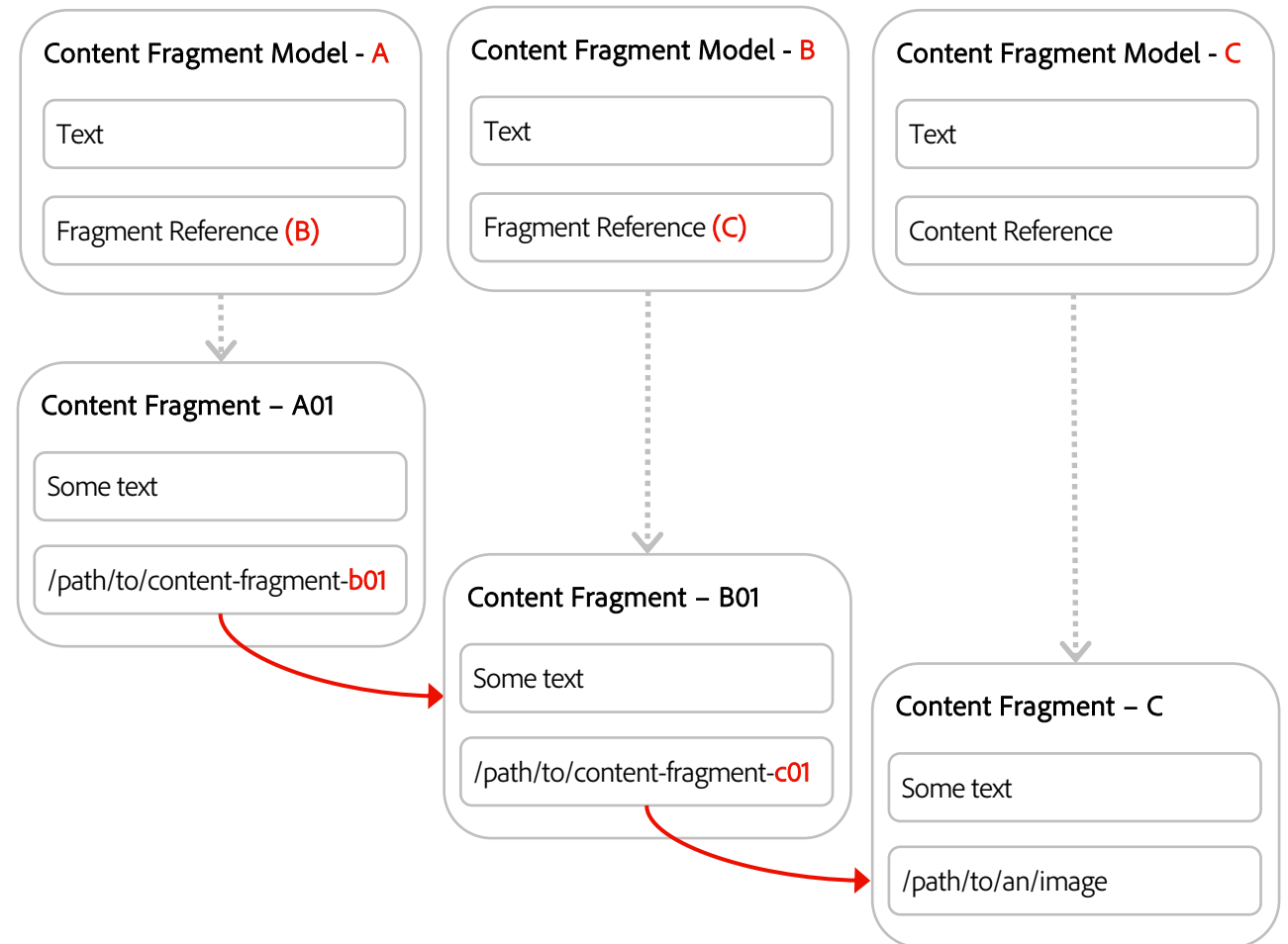
Content Fragment Modeling enhancements



Content Fragment Model editor: New data types

Fragment Reference type

- References another fragment, dependent on a specific Content Fragment Model
- Used to define additional layers of structure when defining a model
- Can be nested over multiple levels
- Enables to retrieve nested Content Fragments for a prime Content Fragment



Content Fragment Model editor: New data types

JSON data type

- Enables you to enter a JSON value in a field
- The JSON value is returned as a JSON scalar in the GraphQL result

Content Fragment Model - Book

author: Single Line Text

info: JSON

Content Fragment - Nice Book

The Famous Writer

{"importantProperty": " Important value"}

```
query {  
  bookByPath (_path: "/path/to/the/book") {  
    item {  
      author  
      info  
    }  
  }  
}
```

```
type BookModel {  
  _path: ID  
  author: String  
  info: Json  
  _metadata: TypedMetaData  
  _variations: [String]  
  _model: ModelInfo  
}
```

```
{  
  "data": {  
    "author": "The Famous Writer",  
    "info": {  
      "importantProperty": " Important value"  
    },  
    ...  
  }  
}
```

Content Fragment Editor enhancements



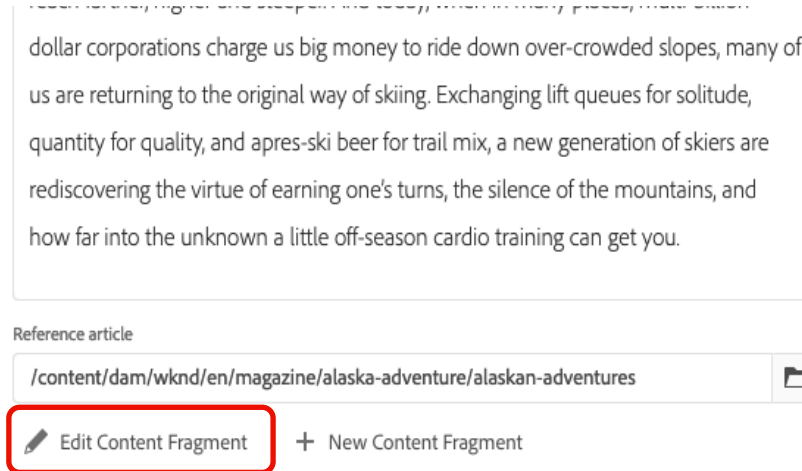
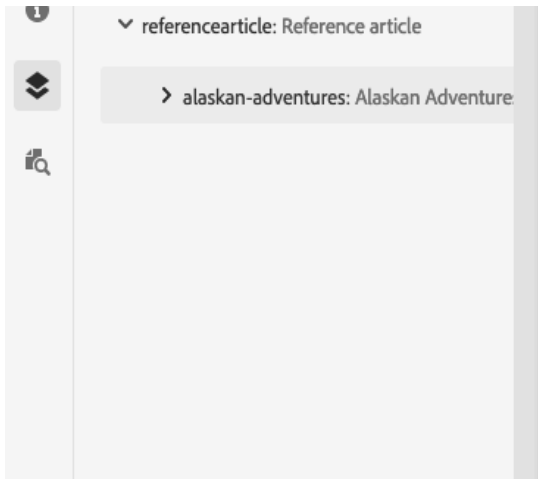
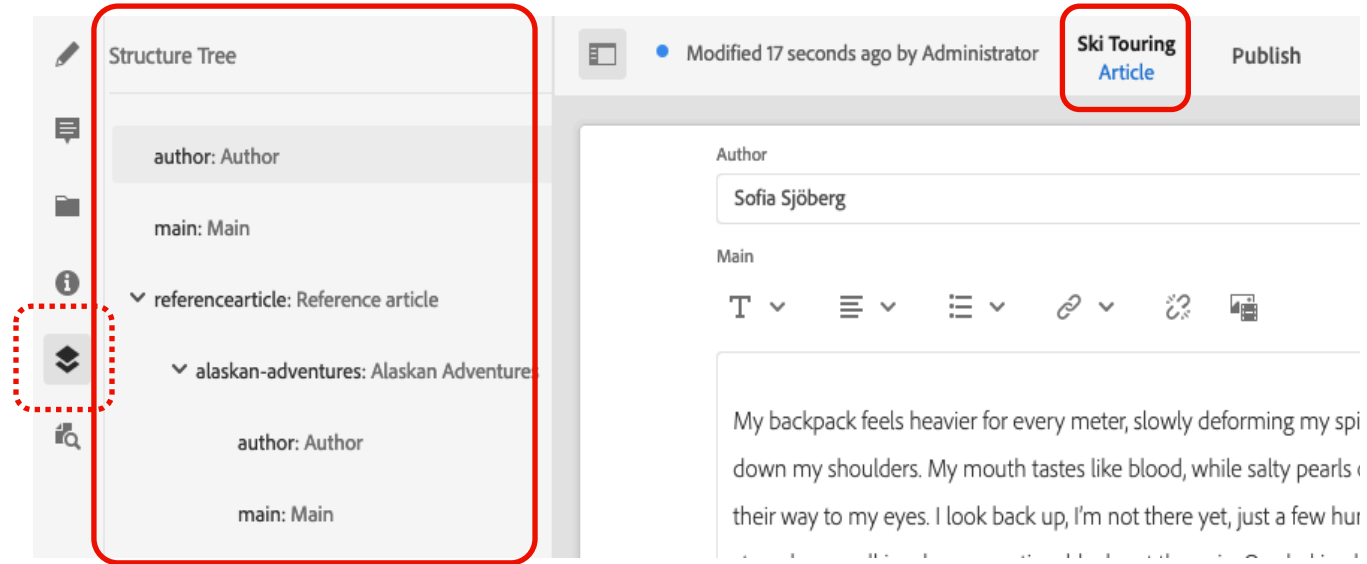
Content Fragment editor enhancements

Structure tree

- Easily navigate through Content Fragment structure

Direct access to corresponding model

Direct editing of nested Content Fragments



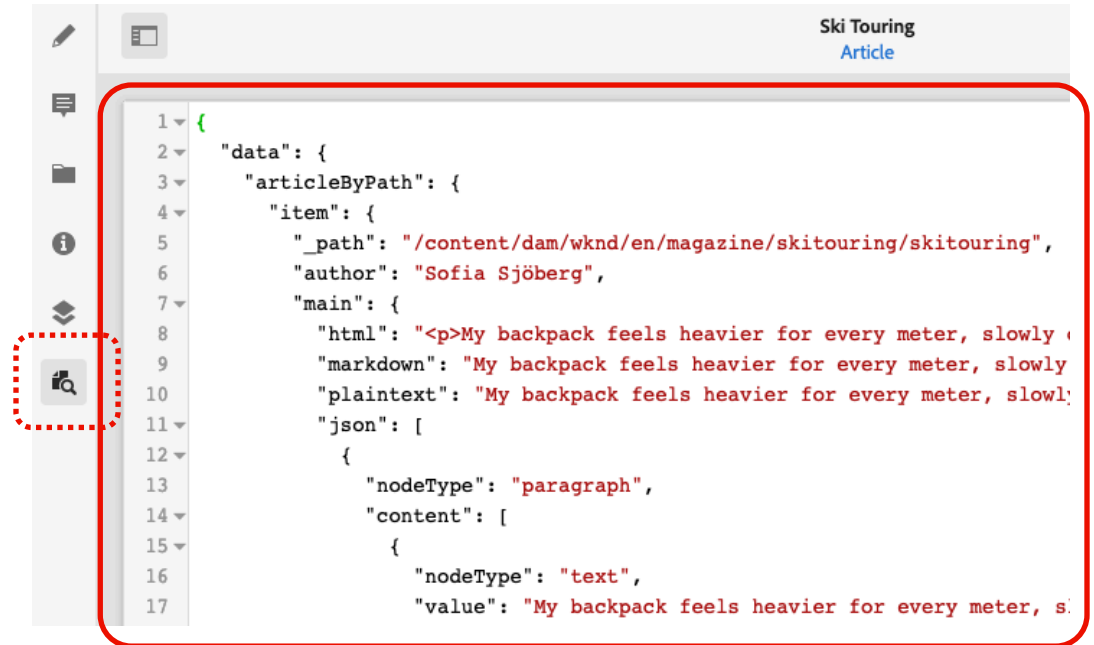
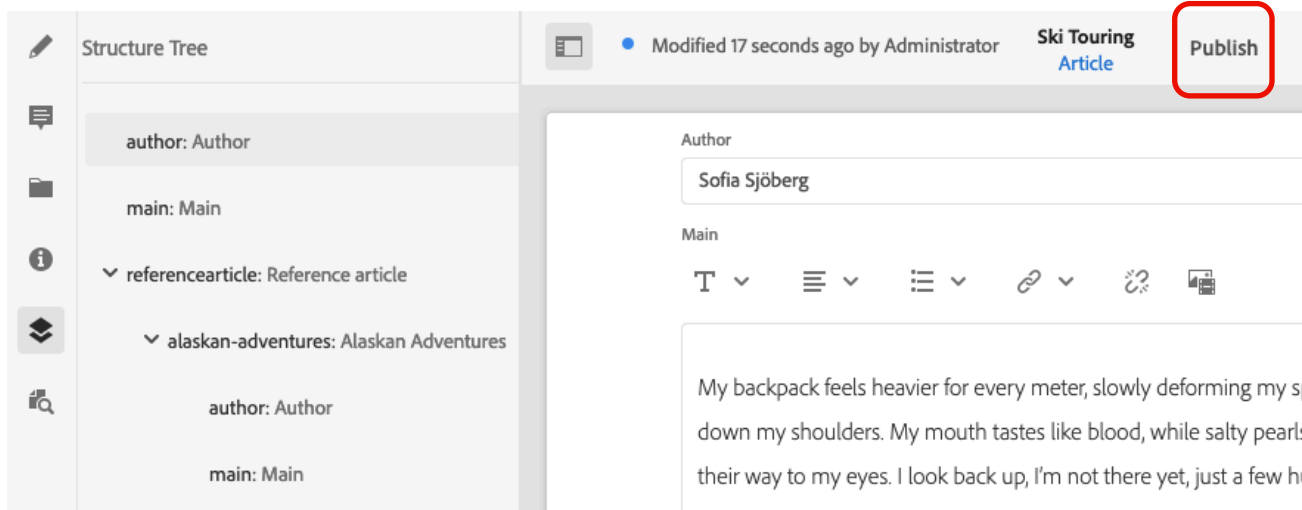
Content Fragment editor enhancements

JSON Preview

- Generates a JSON output for of the current content fragment

Direct publishing

- To save & publish content fragment instantly*



* Upcoming feature, post GA.

Develop and test GraphQL queries

- Syntax highlighting
- Auto completion
- Schema documentation

The screenshot displays the GraphiQL interface with three main sections:

- Query Editor:** Contains a GraphQL query for fetching all adventures with detailed fields like path, title, price, trip length, and primary image.
- Response Viewer:** Shows the JSON response for the query, including an array of adventure objects with their respective details.
- Schema Explorer:** Lists the available fields and their types, such as `adventureTitle: StringFilter` and `adventurePrice: StringFilter`.

Note: GraphiQL content-package is downloadable at: <https://experience.adobe.com/.../aem-graphql/graphiql-0.0.4.zip>

AEM headless: Persisted queries (sneak-peek)*

“GET” your “PUT” queries

- PUT your GraphQL query to save as a persisted query
- GET (execute) the persisted query for result
- Critical for high performance and scalability
- Support HTTP or CDN caching for queries and results

Persisted queries can be enabled via AEM [configuration console](#)*

* Upcoming feature, post GA.

1. Create persisted query through PUT

```
curl -X PUT \  
-H 'authorization: Basic YWRtaW46YWRtaW4=' \  
-H "Content-Type: application/json" \  
"https://<aem-cloud-service-host>/graphql/persist.json/wknd/your-query-name" \  
-d '{your-graphql-query}'
```

2. Check success result

```
{  
  "action": "create",  
  "configurationName": "wknd",  
  "name": "your-query-name",  
  "shortPath": "/wknd/your-query-name",  
  "path": "/conf/wknd/settings/graphql/persistentQueries/your-query-name"  
}
```

3. Execute the persisted query using GET

```
curl -X GET \  
https://<aem-cloud-service-host>/graphql/execute.json/wknd/your-query-name
```


Headless – AEM GraphQL API Demo



AEM & headless: Wrap up

AEM headless capabilities **modernize and simplify** building new experiences

- **Enhances** existing content fragment **modeling capabilities**
- **Streamlines AEM UX** for authoring content fragments
- **Industry standard** for querying contents headless-ly

AEM

UX

Headless

Content Fragments

GraphQL

JSON

HTTP

Experiences

AEM & headless: What's next?

Jump start with headless features of AEM!

- Read more in the docs
 - [Headless and AEM](#)
 - [Explore AEM GraphQL APIs](#)

& start playing with AEM, headless-ly!

