

Contentful to Neo4j



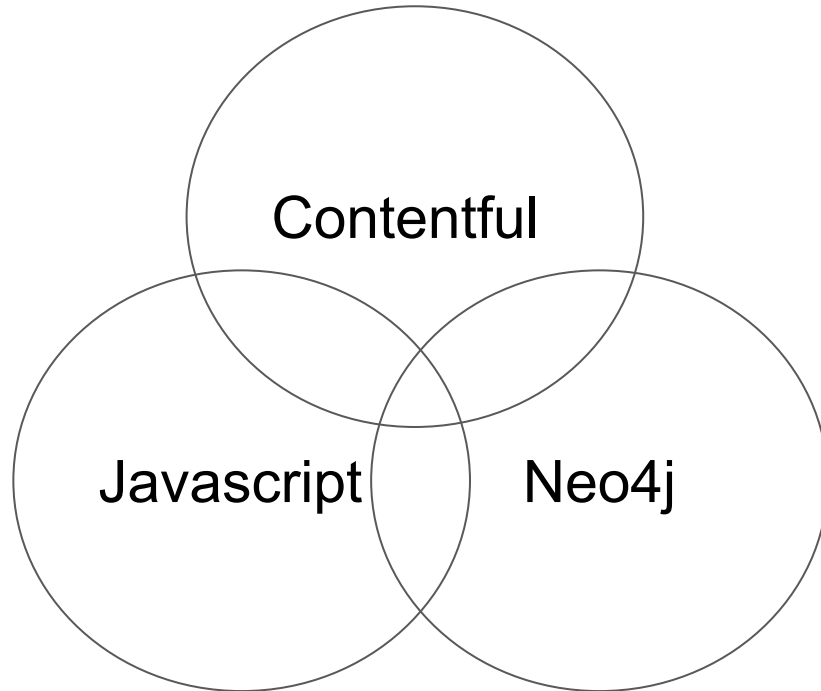
Viewing Contentful Data in Neo4j

About Me

- Chris Eyre
- Software Craftsman at Codurance
- @chriseyre2000
- Author
 - <https://leanpub.com/development2019>
- Elixir mentor on exercism.io
- <https://devrantsblog.wordpress.com/>
- <https://github.com/contentful-to-neo4j>



Three topics here



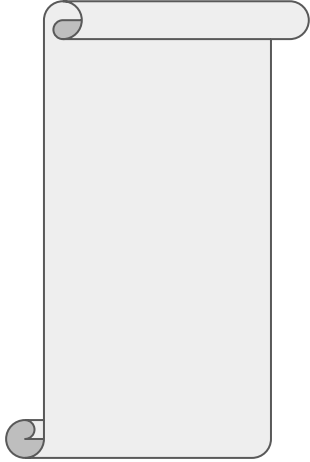
Why would I be interested in this talk?

Why would I be interested in this talk?

- I have a CMS that I want to query
- I need a queryable CMS

Why would I be interested in this talk?

- I have a CMS that I want to query
- I need a queryable CMS
- I have a dataset that I would like to put into Neo4j



Start of new project

We need a
Content Management System

Build vs Buy?

Requirements

Cloud Hosted

Controllable via an API

Traditional CMS

Has two UI's

- Entry of data
- Display of data

You have to customise the supplied UI and will find yourself having to fight to work around what is supplied.

Headless CMS

Has one UI

- Entry of data

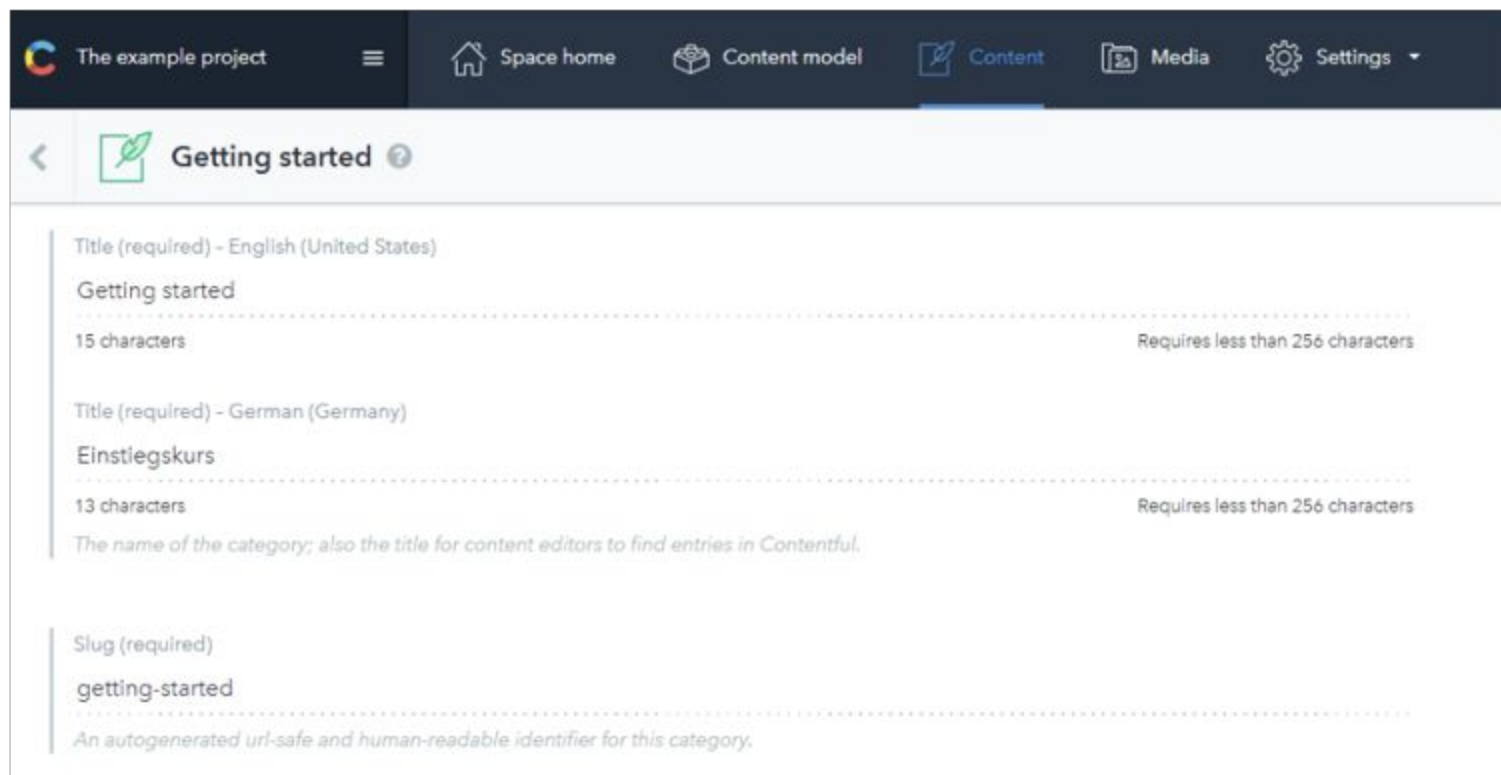
The data is fed to the application as an API (typically as a paged set of JSON data).

It does not fight the frontend, but you have to build it all yourself.

Candidates (at the time)



Contentful is a cloud-hosted headless CMS which provides a user-interface for the content editors.



The screenshot displays the Contentful admin interface. At the top, a dark navigation bar contains the project name 'The example project', a menu icon, and navigation links for 'Space home', 'Content model', 'Content' (which is highlighted), 'Media', and 'Settings'. Below this, a breadcrumb trail shows a back arrow, a leaf icon, and the text 'Getting started'. The main content area is divided into sections for different languages:

- English (United States):** The title field contains 'Getting started', with a character count of 15 and a note that it requires less than 256 characters.
- German (Germany):** The title field contains 'Einstiegskurs', with a character count of 13 and a note that it requires less than 256 characters. A descriptive note below reads: 'The name of the category; also the title for content editors to find entries in Contentful.'
- Slug (required):** The slug field contains 'getting-started', with a note below: 'An autogenerated url-safe and human-readable identifier for this category.'

Contentful does not provide a UI for the end user

Instead it provides API's

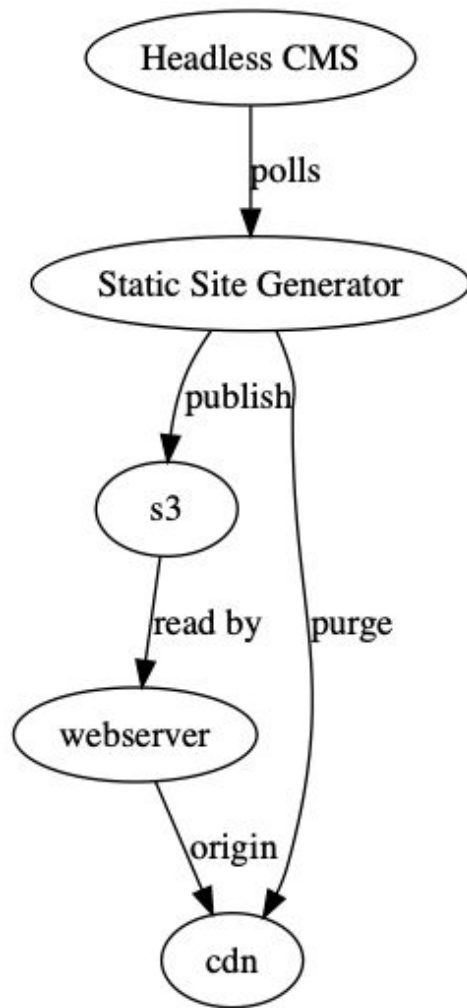
- Content Delivery
- Content Management
- REST
- GraphQL (did not exist when I wrote this utility)

Contentful is partitioned into Spaces

Spaces Contain the following:

- Content Model (Content Type, Schema)
- Content (Instances of Content Type)
- Media (Images, Videos, Zip files)

Demo Contentful





This worked well for a year

Complex Data Types

Related content

Needed 6 different associated categories for the recommendation system to work.


It was hard to find the items that had less than the six related items.

Content editors were claiming that the system had bugs, when it was due to missing content.



Problem

**I can't easily query data in
contentful.**





Insight

The data in a contentful space forms a graph.





Solution

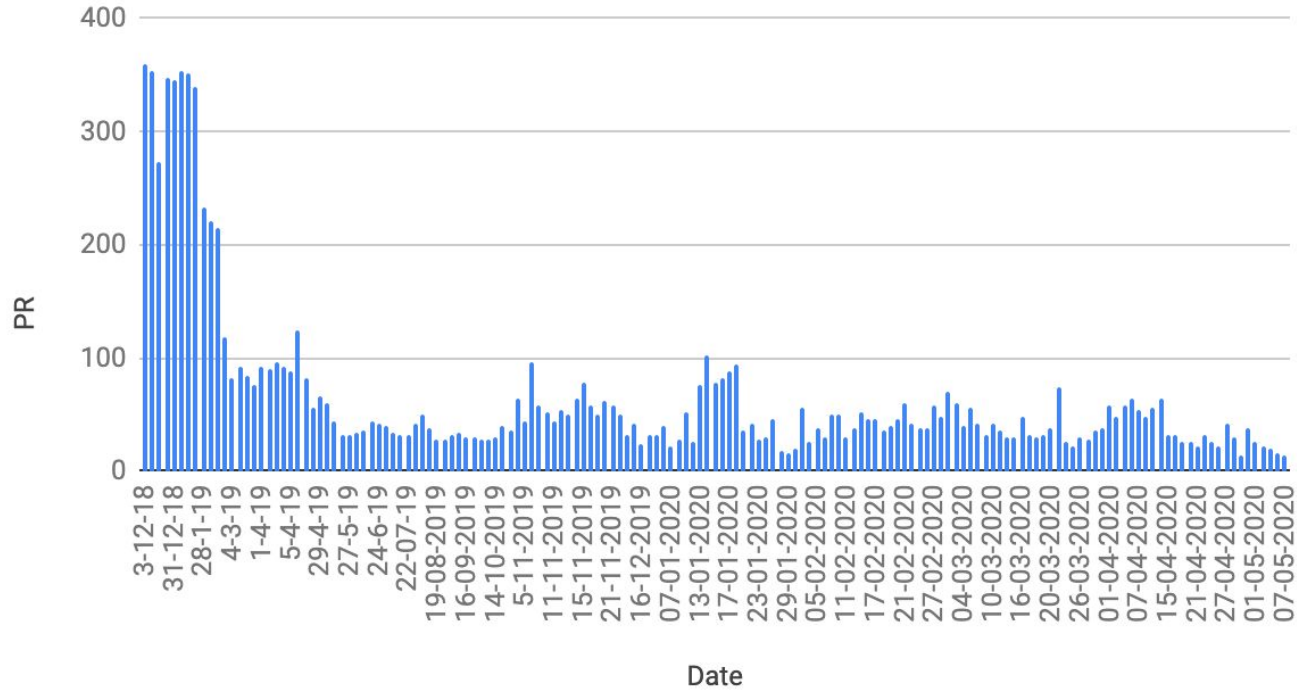
Feed a Neo4j database from Contentful.



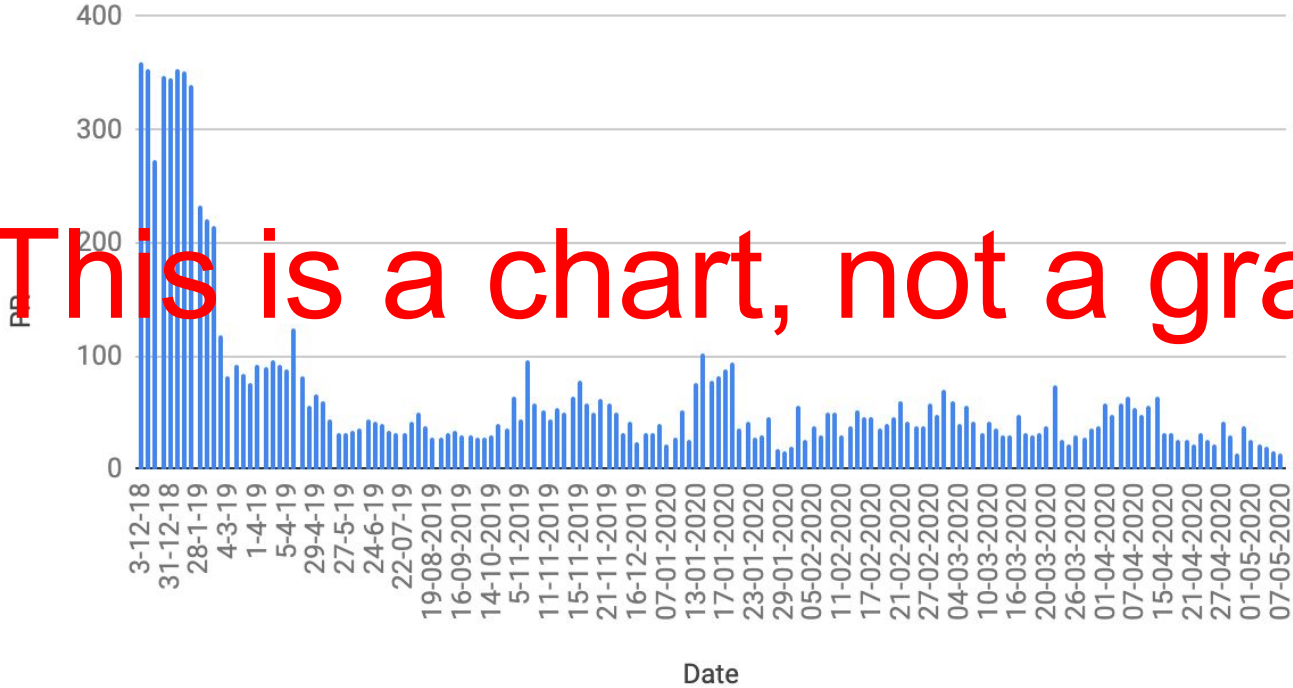
Graphs

Is this a graph?

PR vs Date

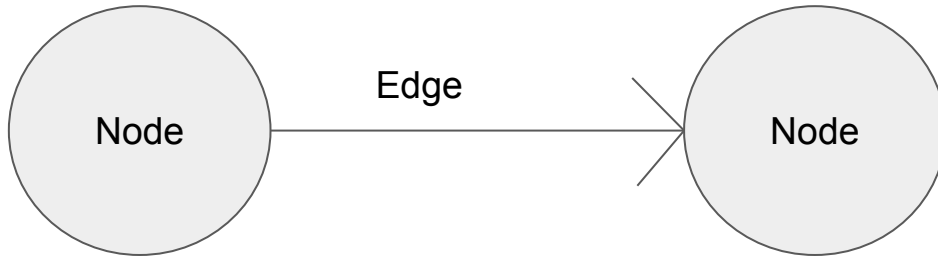


PR vs Date



This is a chart, not a graph

This is a graph:



Neo4j is a graph database.

- Nodes and Relationships instead of tables.
- May be queried in a language called Cypher.
- Also has an api that allows you to manage the database
- Nodes and Relationships can both have attributes.

Simplest Query

The screenshot shows a query interface with the following components:

- Query:** `$ MATCH(a:category) RETURN a`
- Query Editor:** A sidebar on the left with icons for Graph, Table, Text, and Code.
- Graph Visualization:** A central graph showing a node labeled "Getting started" connected to a node labeled "Applicat...".
- Result Preview:** A table at the bottom showing the query result.

category
<code><id>: 248 cmsid: 6ucY5w3oswEUBEYSCEIIC8 cmstype: Entry slug: getting-started title: Getting started</code>

```
$ MATCH(a:category)-[]-(b) RETURN a,b
```



*(4) category(2) course(2)

- Graph
- Table
- Text
- Code



Displaying 4 nodes, 2 relationships.

Demo Neo4j

Javascript

```
const limiter = new Bottleneck({
  minTime: config.contentful.minTimeBetweenTransactions,
  maxConcurrent: 1
});

const getEntries = (limit, skip) => {
  return limiter.schedule( () => contentfulClient.getEntries({
    skip: skip,
    limit: limit,
    include: 0,
    order: 'sys.createdAt'
  }));
};
```

How to go from Contentful to Neo4j



Warnings !

- **This script starts by deleting the graph database.**
 - This means I never need to back it up since it is quickly recreated.
- If it can't migrate a field it will skip it, log and carry on
 - Let me know what went wrong and I'll try and fix it
- The entire graph is saved in one transaction which can be slow for large Contentful spaces (over 10K Entries + Assets).

Warnings

- **The script starts by deleting the graph database.**
 - This means that I can avoid backups by having the ability to quickly recreate the graph db
- If I can't create a field it ignores it, logs an error and continues.
- The entire database is loaded in one transaction
- Currently only works on Neo4j 3.5 (not had time to make it work with 4 yet)

Mappings

To start with I performed the following inserts:

- Assets
- Entities
- Relationships

This worked for 95% of the data.

Scalar properties on Contentful become properties on the node
(includes arrays of scalar data).

Entity relationship become relationships to other entities
(array order is noted on the relationship).

Mappings part 2

To handle the special characters that could not be directly inserted I then started to use parameterised queries. This allowed me to store all of the content that I wanted and maintain the unicode characters.

At one point we had an article translated into 27 languages. That required a lot of special unicode character. The parameterised queries handled them all

This is where I changed the strategy to:

- Assets
- Entities (just the name and primary id)
- Relationships + Properties

This is slower, but far more reliable. Note that the size limits on a field in Contentful were (and may still be) smaller than the field size limits in Neo4j

How to use

- github.com/chriseyre2000/contentful-to-neo4j
- Npm install
- Node index.js (+environment variables)

Data Extraction

- Contentful entries have primitive fields, references to other entries, assets and lists of the above.
- The data extraction process copies every published asset in a space, every published entry in the space (including its fields) and creates relationships between them.
- Technically it's a visualization tool – but it is the complete dataset.

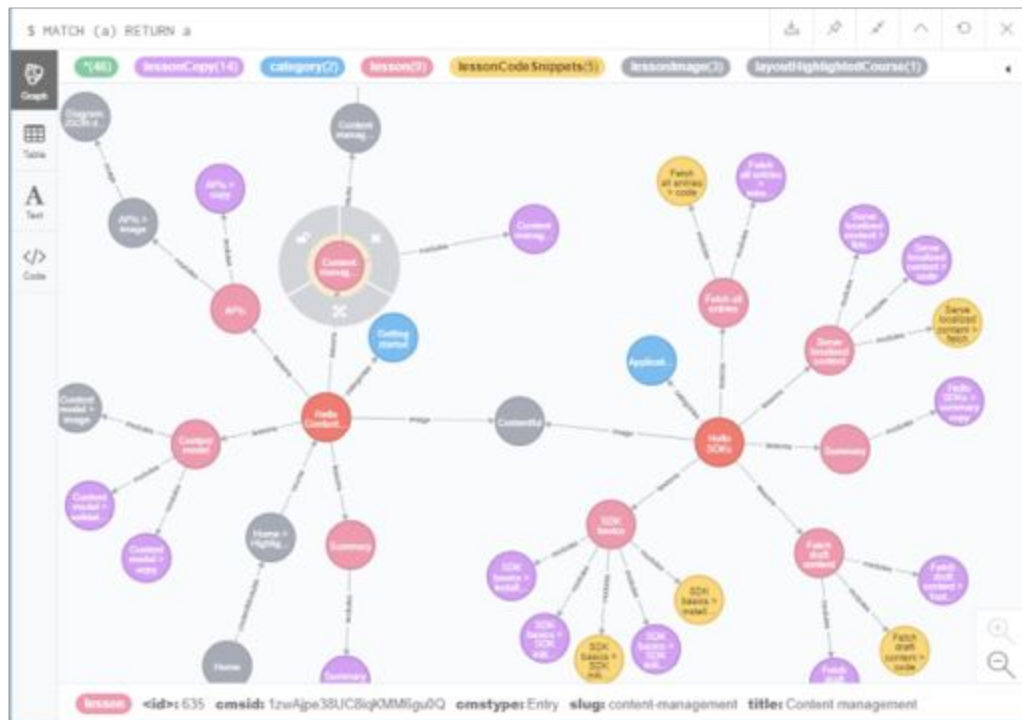
Fun issues whilst developing

- Contentful is highly rate limited (espically on the free tier). So throttle your reads. Also read as much as you can in each batch.
- There is a size limit on a page of data that can be fetched. Exclude includes (the items that the data returned references – you will already have these as we page the whole db).
- Promises are fun to coordinate and test. Promise.resolve/setTimeout 1 is your friend.
- You need to fetch all assets and entries to guarantee that the relationships will work.
- Neo4j is very good at telling you that something is wrong in a transaction – not always what.

Other environment variables

- NEO4J_USER
- GRAPHENEDB_BOLT_USER
- NEO4J_SERVER
- GRAPHENEDB_BOLT_URL
- CONTENTFUL_BATCH_SIZE
- CONTENTFUL_DELAY

Sample Query



Heroku and Graphene

- If you don't want to host Neo4j yourself there are options on Heroku.
- Heroku is a hosting platform with per second billing (and a free tier).
- Graphene is a hosted version of Neo4j
- The app has the defaults set up so that it's trivial to get it working on Heroku.

Demo

- Demonstrating moving data from contentful to Neo4j

Environment Variables : Minimal

If you are running Neo4j locally this is all you need (assuming that you are using the Neo4j user).

SPACE_ID=xxx

CONTENTFUL_ACCESS_TOKEN=yyy

NEO4J_PASSWORD=badpassword

Questions