

# Use XKCD API to build a searchable Database of XKCD Comics

Projektarbeit:

**Big Data**

Des Studiengangs Informatik  
an der Dualen Hochschule Baden-Württemberg Stuttgart

von

**Simon Jess**

26. November 2021

Matrikelnummer, Kurs:

8268544, INF19C

Gutachter:

Mittelstädt, Marcel

## Inhalt

Dateien .....	1
Grundidee.....	1
Airflow.....	2
Website.....	5

## Dateien

Alle Folgenden Dateien wurden im Rahmen der Prüfungsleistung abgegeben:

### Airflow:

- Dag.py
- data\_collector.py
- hdfs\_partitioner.py

### Website:

- app.js
- Docker File
- index.html
- index.js
- package.json

### MySQL:

- CREATE TABLE xkcd\_datatable.sql

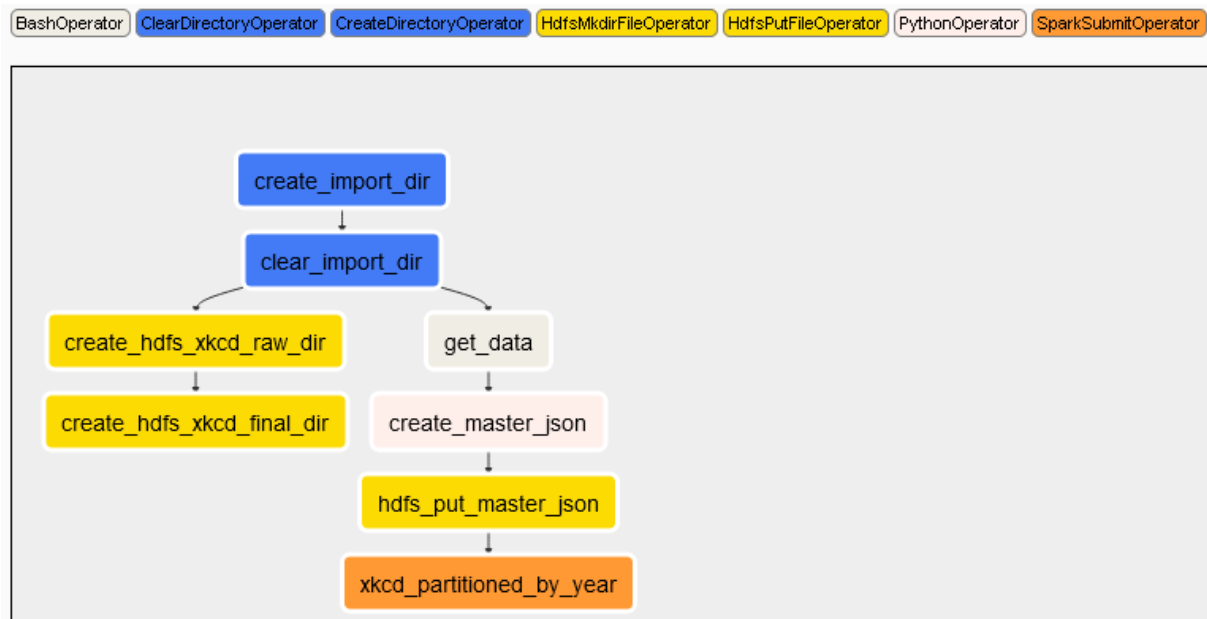
## Grundidee

Meine Grundidee ist:

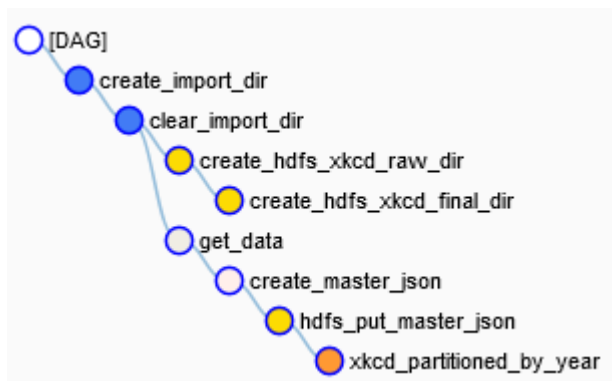
- Alle benötigten Verzeichnisse erstellen
- Die Daten auf dem Airflow Docker herunterladen und zwischenspeichern
- Eine Master JSON aus allen einzelnen Comic JSON zu erstellen
- Master JSON auf HDFS schieben
- Auf HDFS die Daten bereinigen etc. und partitionieren
- Daten in MySQL Exportieren
- Website mit Node.js und Express → Daten von MySQL Abfragen und in einem simplen Bootstrap HTML Gemisch visualisieren

## Airflow

### Workflow:



### Dag.py:



Mit der Parallelisierung wird erreicht, dass während die zwei Verzeichnisse auf HDFS erstellt werden, schonmal die JSON Dateien von der XKCD API Abgefragt werden, da dies deutlich länger benötigt.

### „create\_import\_dir“ / create\_local\_import\_dir:

Erstellen eines Verzeichnisses in dem Airflow Docker Container. Das Verzeichnis ist das Zielverzeichnis für die von der API-Abfrage empfangenen JSON Dateien.

***„clear\_import\_dir“ / clear\_local\_import\_dir:***

Alle in dem soeben erstellten Verzeichnis vorhandenen Dateien löschen, um sicher zu gehen, dass das Verzeichnis leer ist.

***create\_hdfs\_xkcd\_raw\_dir:***

Erstellen eines Verzeichnisses auf dem Hadoop File System (HDFS). Zielverzeichnis für die auf in dem Airflow erstellte Master JSON Datei.

***create\_hdfs\_xkcd\_final\_dir***

Erstellen eines Verzeichnisses auf dem Hadoop File System (HDFS). Zielverzeichnis für die mit Spark partitionierten Parquet Dateien und die external Table. Zielort für die Aufbereiteten Daten.

***get\_data***

Bash Operator, um die data\_collector.py Datei mit dem enthaltenen Python Code aufzurufen und auszuführen.

***convert\_Data():***

Öffnen des Verzeichnisses in dem Airflow Docker, welches alle heruntergeladenen JSON Dateien enthält. Daraufhin wird eine Liste mit allen in diesem Verzeichnis vorhandenen Dateien angelegt. Diese Liste wird zum Schluss durch iteriert und jede JSON Datei gelesen. Dabei wird der Inhalt erst einer Liste angefügt, welche dann mittels Pandas zu einem Dataframe konvertiert wird und in eine einzelne „große“ Master JSON Datei geschrieben wird.

***create\_master\_json***

Python Operator, um die in dem Dag enthaltene Funktion convert\_Data() aufzurufen.

***hdfs\_put\_master\_json***

Put File Operator, um die in dem Airflow Docker erstellte Master JSON Datei auf das HDFS zu verschieben.

### **„*xkcd\_partitioned\_by\_year*“ / *partitionate\_hdfs\_final***

Spark Submit Operator, um die `hdfs_partitioner.py` Datei mit dem enthaltenen PySpark Code auszuführen.

#### **`data_collector.py`:**

Idee von dem Datacollector ist es, mit einer initiale API-Anfrage den Statuscode 200 zurück zu bekommen, um damit in eine „While“ Schleife zu gelangen. Diese bricht ab, sobald keine Daten, sondern ein Fehler mit bspw. dem Statuscode 404 zurückgeliefert wird. Jede Request wird in dem Airflow Docker in einem lokalen Verzeichnis gespeichert, um später weiterverarbeitet zu werden.

Der Index 404 wird übersprungen, da es vermutlich ein Spaß seitens xkcd ist, dass es das File mit dem Index des Fehlercodes: „404 File not found“ nicht gibt.

#### **`hdfs_partitioner.py`:**

Hier ist all der PySpark Code, um die Rohdaten zu bereinigen, bearbeiten, selektieren und partitionieren. Um die Daten einzulesen wird der gesamte Ordner „raw“ auf HDFS eingelesen. Dann werden in dem Dataframe lediglich die Daten behalten, welche am Ende für das Frontend benötigt werden, dazu gehören „num“, „year“, „title“ und „img“. Diese werden dann noch von Einträgen mit Nullwerten bereinigt und nach Jahren partitioniert in das „final“ Verzeichnis geschrieben. Des Weiteren werden die Daten in dem Dataframe auch direkt in die angegebene MySQL Datenbank geschrieben.

P.S. erstellen der Tabelle in der SQL-Datei.

## Website

### **app.js**

Hier wird express initialisiert, die index.js und index.html, welche den Aufbau der Website beschrieben importiert. Des Weiteren wird die Verbindung mit der Datenbank hier aufgebaut und angegeben, dass die Website über den internen Port 3000 läuft. Des Weiteren wird hier die Post Anfrage für die MySQL Datenbank erstellt, welche dann mit dem Wortbestandteil eine DB Abfrage macht.

### **index.js**

Die Index.js Datei ist das Herzstück der Website. Hier wird der http Request, welche nach außen über Port 5000 erreichbar ist ausgeführt. Dazu wird die in der app.js implementierte Methode xkcd\_search aufgerufen.

Die von der DB erhaltenen Daten werden dann zu einer Liste gemacht, welche dann iterativ über DOM Elemente der Tabelle der index.html hinzugefügt werden können. Dafür wird für jeden Listeneintrag eine neue Reihe hinzugefügt und die Spalten der Tabelle definiert.

Clear\_table() wird bei jeder abgesendeten Abfrage ausgeführt, dass die Tabelle zu Beginn der Tabellenerstellung auch keine DOM Elemente enthält und somit leer ist.

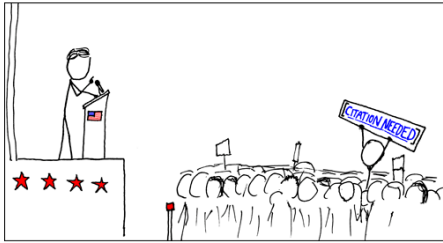

### **index.html**

Die HTML Seite liefert den Eingabebereich, für die Texteingabe des Wortteils oder zu suchenden Worts. Im zweiten Bereich der Website wird die Struktur, sowie der Header der Tabelle bereitgestellt, welche durch das JavaScript dann mit den DOM Elementen und damit den Comic Inhalten aufgebaut wird, Des Weiteren wird Bootstrap importiert, um sich lästiges und aufwendiges CSS-Designen zu ersparen.

Input area:

Test

Comics suchen

Number:	Title:	Image:	Source:
285	Wikipedian Protester		<a href="https://xkcd.com/285/">https://xkcd.com/285/</a>
329	Turing Test	<p>TURING TEST EXTRA CREDIT: CONVINCE THE EXAMINER THAT <u>HE'S</u> A COMPUTER.</p> <p>YOU KNOW, YOU MAKE SOME REALLY GOOD POINTS. I'M ... NOT EVEN SURE WHO I AM ANYMORE.</p> 	<a href="https://xkcd.com/329/">https://xkcd.com/329/</a>