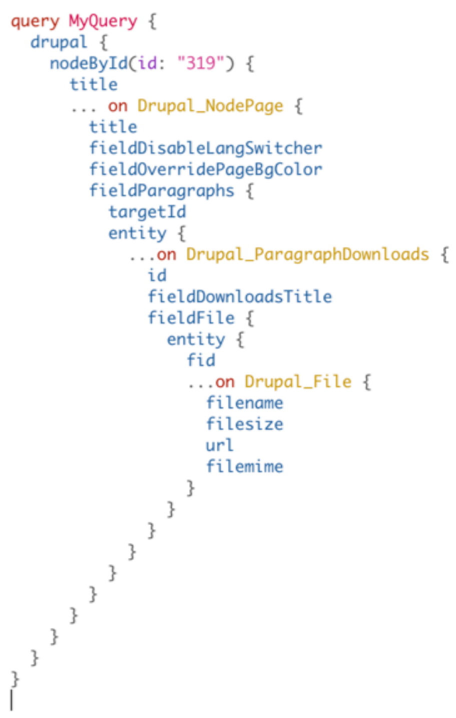
Creation of a new Paragraph

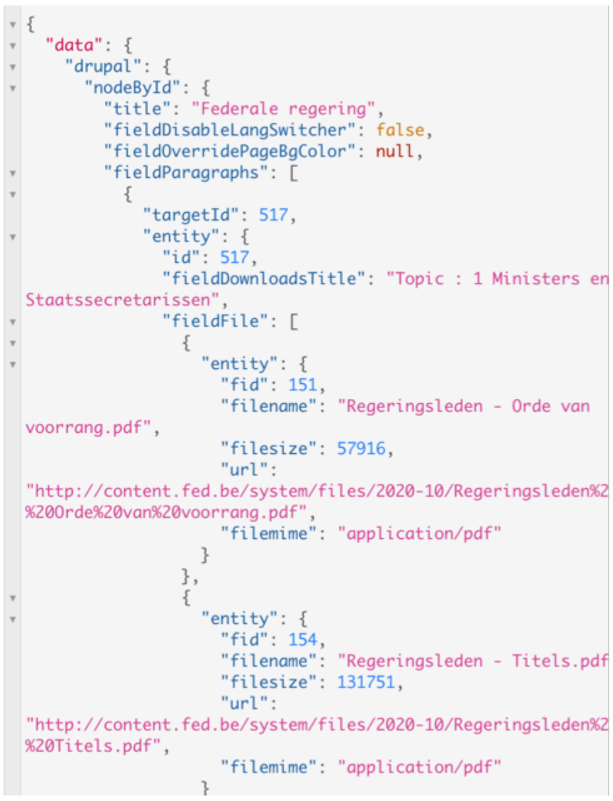
Check the Paragraph definition in Drupal and ensure that we have at least one node using that paragraph.

Start Gatsby and access the GraphQL url - http://localhost:8000/___graphql

Create a query using nodeById and the nid of the module that contains the paragraph



```
query MyQuery {
  drupal {
    nodeById(id: "319") {
      title
      ... on Drupal_NodePage {
        title
        fieldDisableLangSwitcher
        fieldOverridePageBgColor
        fieldParagraphs {
          targetId
          entity {
            ...on Drupal_ParagraphDownloads {
              id
              fieldDownloadsTitle
              fieldFile {
                entity {
                  fid
                  ...on Drupal_File {
                    filename
                    filesize
                    url
                    filemime
                  }
                }
              }
            }
          }
        }
      }
    }
  }
}
```



```
{
  "data": {
    "drupal": {
      "nodeById": {
        "title": "Federale regering",
        "fieldDisableLangSwitcher": false,
        "fieldOverridePageBgColor": null,
        "fieldParagraphs": [
          {
            "targetId": 517,
            "entity": {
              "id": 517,
              "fieldDownloadsTitle": "Topic : 1 Ministers en
Staatssecretarissen",
              "fieldFile": [
                {
                  "entity": {
                    "fid": 151,
                    "filename": "Regeringsleden - Orde van
voorrang.pdf",
                    "filesize": 57916,
                    "url":
"http://content.fed.be/system/files/2020-10/Regeringsleden%2
%20Orde%20van%20voorrang.pdf",
                    "filemime": "application/pdf"
                  }
                },
                {
                  "entity": {
                    "fid": 154,
                    "filename": "Regeringsleden - Titels.pdf
                    "filesize": 131751,
                    "url":
"http://content.fed.be/system/files/2020-10/Regeringsleden%2
%20Titels.pdf",
                    "filemime": "application/pdf"
                  }
```

On the query Check that the correct paragraph name is being used, and dig on the required fields (e.g. fieldFile)

On gatsby code, in the src/queries folder, duplicate an existing query, rename to have the new paragraph in the name, and update the query contents to match what is required.
Update the FieldParagraphs query to have the new Fragment

On src/storybook/paragraphs create a new folder with the paragraph name (using same convention as per others), we should have at least:

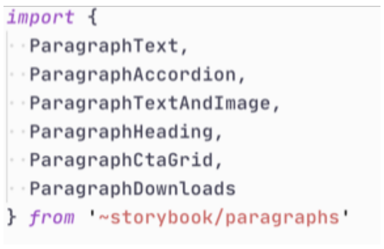
paragraph-name.tsx
styled.tsx
paragraph-name.stories.tsx
index.ts

The props we receive on the paragraph should be simple so we should avoid having direct names from drupal fields (like on the existing components)

Implement all the logic required on the paragraph code and test directly on the storybook.

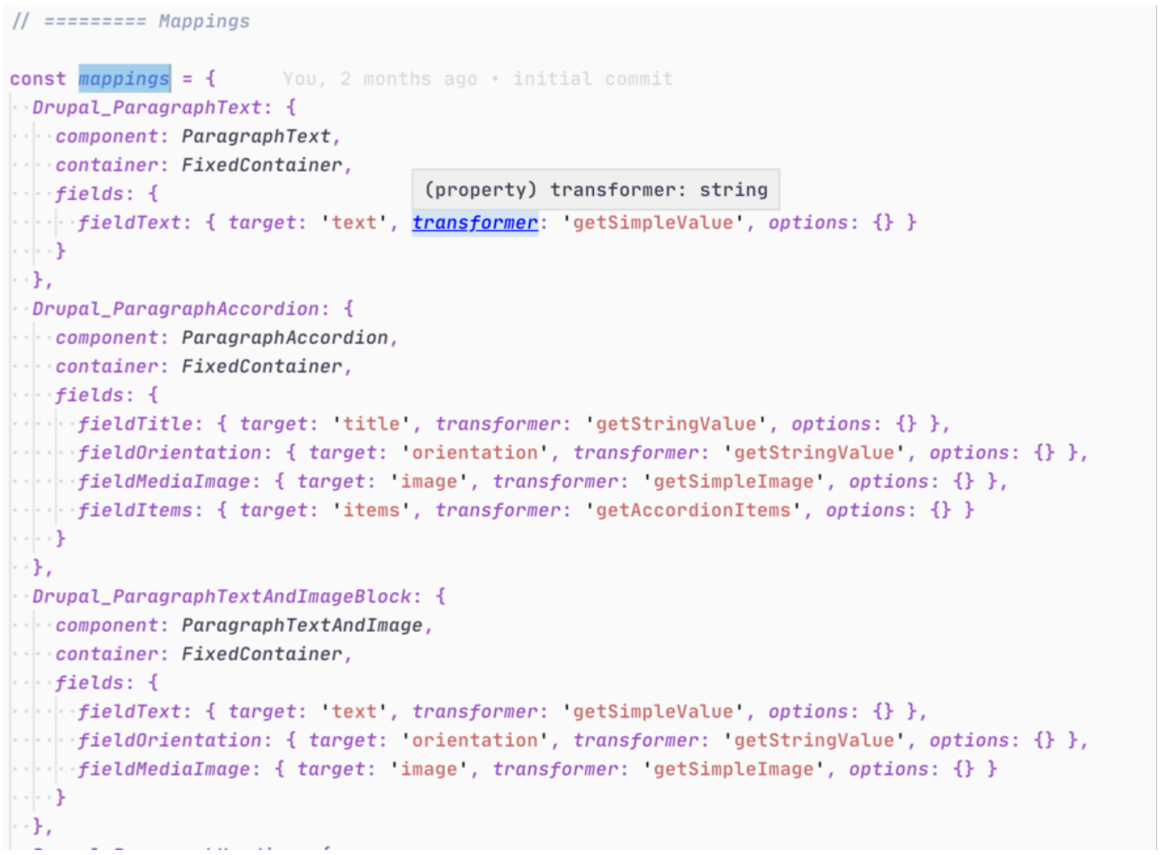
When the component is stable on the storybook open the src/utils/components.tsx

Import the new component:



```
import {
  ParagraphText,
  ParagraphAccordion,
  ParagraphTextAndImage,
  ParagraphHeading,
  ParagraphCtaGrid,
  ParagraphDownloads
} from '~storybook/paragraphs'
```

Update the mappings object, e.g:


(property) transformer: string
You, 2 months ago • initial commit

```
// ========= Mappings

const mappings = {
  Drupal_ParagraphText: {
    component: ParagraphText,
    container: FixedContainer,
    fields: {
      fieldText: { target: 'text', transformer: 'getSimpleValue', options: {} }
    }
  },
  Drupal_ParagraphAccordion: {
    component: ParagraphAccordion,
    container: FixedContainer,
    fields: {
      fieldTitle: { target: 'title', transformer: 'getStringValue', options: {} },
      fieldOrientation: { target: 'orientation', transformer: 'getStringValue', options: {} },
      fieldMediaImage: { target: 'image', transformer: 'getSimpleImage', options: {} },
      fieldItems: { target: 'items', transformer: 'getAccordionItems', options: {} }
    }
  },
  Drupal_ParagraphTextAndImageBlock: {
    component: ParagraphTextAndImage,
    container: FixedContainer,
    fields: {
      fieldText: { target: 'text', transformer: 'getSimpleValue', options: {} },
      fieldOrientation: { target: 'orientation', transformer: 'getStringValue', options: {} },
      fieldMediaImage: { target: 'image', transformer: 'getSimpleImage', options: {} }
    }
  },
```

The mappings is composed by:

component: Name of the Storybook Component
container: FluidContainer if the component is edge to edge otherwise FixedContainer
fields: object keyed by each Drupal field returned by GraphQL query, where target is the corresponding prop name in the storybook component, and transformer is the function that will handle any manipulation of data that is required.