

# [GraphQL] Convert monolithic query into a pattern of small queries

EditOpen with ▾

## #15974

Merged


nckslvn merged 52 commits into master from nps-individual-queries on Feb 11

💬 Conversation 21 ↶ Commits 52 📄 Checks 4 📄 File +1,689 -174


Changes from all commits ▾ File filter... ▾ Jump to... ▾ ⚙ ▾

104 src/site/stages/build/drupal/api.js

8	8	
9	9	const DRUPALS = require('../../../../constants/drupals');
10	10	const { queries, getQuery } = require('./queries');
	11	+ const {
	12	+   getIndividualizedQueries,
	13	+   CountEntityTypes,
	14	+ } = require('./individual-queries');
11	15	
12	16	const syswidecas = require('syswide-cas');
13	17	
167	171	}
168	172	},
169	173	
	174	+ async getAllPagesViaIndividualGraphQLQueries(onlyPublishedContent = t
	175	+   /* eslint-disable no-console, no-await-in-loop */

 va-vfs-bot on Feb 8 Member

ESLint disabled here

 Reply...

Resolve conversation

176	+	
177	+	say('Pulling from Drupal via GraphQL...');
178	+	
179	+	const entityCounts = await this.query({
180	+	query: CountEntityTypes,
181	+	});

```

182 +
183 +     say('Received node counts...');
184 +     console.table(entityCounts.data);
185 +
186 +     const result = {
187 +         data: {
188 +             nodeQuery: {
189 +                 entities: [],
190 +             },
191 +         },
192 +     };
193 +
194 +     const individualQueries = Object.entries(
195 +         getIndividualizedQueries(entityCounts),
196 +     );
197 +
198 +     const totalQueries = individualQueries.length;
199 +
200 +     const parallelQuery = async () => {
201 +         const [queryName, query] = individualQueries.pop();
202 +         const request = this.query({
203 +             query,
204 +             variables: {
205 +                 today: moment().format('YYYY-MM-DD'),
206 +                 onlyPublishedContent,
207 +             },
208 +         });
209 +
210 +         const startTime = moment();
211 +         const json = await request;
212 +
213 +         if (json.errors) {
214 +
215 +             console.log(json.errors);
216 +             throw new Error(`${queryName} resulted in errors`);
217 +         }
218 +
219 +         if (json.data?.nodeQuery) {
220 +             result.data.nodeQuery.entities.push(...json.data.nodeQuery.entities);
221 +         } else {
222 +             Object.assign(result.data, json.data);
223 +         }
224 +
225 +         let timeElapsed = moment().diff(startTime, 'seconds');
226 +         let pageCount = json.data.nodeQuery
227 +             ? json.data.nodeQuery.entities.length
228 +             : '[n/a]';
229 +
230 +         if (timeElapsed > 60) {
231 +             timeElapsed = chalk.red(timeElapsed);
232 +         }
233 +
234 +         if (pageCount > 100) {
235 +             pageCount = chalk.red(pageCount);

```

```

236 +
237 +     say(
238 +       `| ${chalk.blue(queryName)} | ${timeElapsed}s | ${pageCount} pages`
239 +     );
240 +
241 +     if (individualQueries.length > 0) {
242 +       return parallelQuery();
243 +     }
244 +
245 +     return true;
246 +   };
247 +
248 +   // Cap the amount of pending requests allowed out at once
249 +   // And also stagger their execution so that at no point
250 +   // are we totally overwhelming the CMS.
251 +   const maxParallelRequests = 15;
252 +   const overallStartTime = moment();
253 +   const staggeredRequests = new Array(maxParallelRequests)
254 +     .fill(null)
255 +     .map(() => {
256 +       return new Promise(resolve => {
257 +         setTimeout(() => resolve(parallelQuery()), 1000);
258 +       });
259 +     });
260 +
261 +   await Promise.all(staggeredRequests);
262 +
263 +   const overallTimeElapsed = moment().diff(overallStartTime, 'seconds');
264 +
265 +   console.log(
266 +     `Finished ${totalQueries} queries in ${overallTimeElapsed}s with ${
267 +       result.data.nodeQuery.entities.length
268 +     } pages`,
269 +   );
270 +
271 +   return result;
272 + },
273 +

```

```

170 274   getAllPages(onlyPublishedContent = true) {
171 275     return this.query({
172 276       query: getQuery(queries.GET_ALL_PAGES),

```

88 src/site/stages/build/drupal/graphql/CountEntityTypes.graphql.js

@@ -0,0 +1,88 @@

```

1 + const CountEntityTypes = `
2 + {
3 +
4 +   benefitPages: nodeQuery(
5 +     filter: {
6 +       conditions: [
7 +         {field: "status", value: ["1"]},
8 +         {field: "type", value: ["page"]}

```

```

9 +     }}
10 +     ) {
11 +         count
12 +     }
13 +
14 +     vaForm: nodeQuery(
15 +         filter: {
16 +             conditions: [
17 +                 {field: "status", value: ["1"]},
18 +                 {field: "type", value: ["va_form"]}
19 +             ]}
20 +     ) {
21 +         count
22 +     }
23 +
24 +     healthCareLocalFacility: nodeQuery(
25 +         filter: {
26 +             conditions: [
27 +                 {field: "status", value: ["1"]},
28 +                 {field: "type", value: ["health_care_local_facility"]}
29 +             ]}
30 +     ) {
31 +         count
32 +     }
33 +
34 +     healthServicesListing: nodeQuery(
35 +         filter: {
36 +             conditions: [
37 +                 {field: "status", value: ["1"]},
38 +                 {field: "type", value: ["health_services_listing"]}
39 +             ]}
40 +     ) {
41 +         count
42 +     }
43 +
44 +     event: nodeQuery(
45 +         filter: {
46 +             conditions: [
47 +                 {field: "status", value: ["1"]},
48 +                 {field: "type", value: ["event"]}
49 +             ]}
50 +     ) {
51 +         count
52 +     }
53 +
54 +     healthCareRegionDetailPage: nodeQuery(
55 +         filter: {
56 +             conditions: [
57 +                 {field: "status", value: ["1"]},
58 +                 {field: "type", value: ["health_care_region_detail_page"]}
59 +             ]}
60 +     ) {
61 +         count
62 +     }

```

```

63 +
64 +   healthCareRegionPage: nodeQuery(
65 +     filter: {
66 +       conditions: [
67 +         {field: "status", value: ["1"]},
68 +         {field: "type", value: ["health_care_region_page"]}
69 +       ]}
70 +     ) {
71 +       count
72 +     }
73 +
74 +   personProfile: nodeQuery(
75 +     filter: {
76 +       conditions: [
77 +         {field: "status", value: ["1"]},
78 +         {field: "type", value: ["person_profile"]}
79 +       ]}
80 +     ) {
81 +       count
82 +     }
83 +   }
84 + `;
85 +
86 + module.exports = {
87 +   CountEntityTypes,
88 + };

```

82 src/site/stages/build/drupal/graphql/GetAllPages.graphql.js

```

14 14   const eventPage = require('./eventPage.graphql');
15 15   const facilitySidebarQuery = require('./navigation-fragments/facilitySide
16 16   const faqMultipleQaPage = require('./faqMultipleQa.graphql');
17 17   - const fragments = require('./fragments.graphql');
17 17   + const { ALL_FRAGMENTS } = require('./fragments.graphql');
18 18   const healthCareLocalFacilityPage = require('./healthCareLocalFacilityPage
19 19   const healthCareRegionDetailPage = require('./healthCareRegionDetailPage.
20 20   const healthServicesListingPage = require('./healthServicesListingPage.gr
61 61   const nodeContentFragments = useTomeSync
62 62     ? ''
63 63     : `
64   -   ${fragments}
65   -   ${landingPage}
66   -   ${page}
67   -   ${healthCareRegionPage}
68   -   ${healthCareLocalFacilityPage}
69   -   ${healthCareRegionDetailPage}
70   -   ${pressReleasePage}
71   -   ${vamcOperatingStatusAndAlerts}
72   -   ${newsStoryPage}
73   -   ${eventPage}
74   -   ${officePage}
75   -   ${bioPage}
76   -   ${vaFormPage}
77   -   ${benefitListingPage}

```

```

78 -   ${eventListingPage}
79 -   ${storyListingPage}
80 -   ${leadershipListingPage}
81 -   ${healthServicesListingPage}
82 -   ${pressReleasesListingPage}
83 -   ${locationListingPage}
84 -   ${qaPage}
85 -   ${faqMultipleQaPage}
86 -   ${stepByStepPage}
87 -   ${mediaListImages}
88 -   ${checklistPage}
89 -   ${mediaListVideos}
90 -   ${supportResourcesDetailPage}
91 -   ${basicLandingPage}
92 -   ${nodeCampaignLandingPage}

64 +   ${ALL_FRAGMENTS}
65 +   ${landingPage.fragment}
66 +   ${page.fragment}
67 +   ${healthCareRegionPage.fragment}
68 +   ${healthCareLocalFacilityPage.fragment}
69 +   ${healthCareRegionDetailPage.fragment}
70 +   ${pressReleasePage.fragment}
71 +   ${vamcOperatingStatusAndAlerts.fragment}
72 +   ${newsStoryPage.fragment}
73 +   ${eventPage.fragment}
74 +   ${officePage.fragment}
75 +   ${bioPage.fragment}
76 +   ${vaFormPage.fragment}
77 +   ${benefitListingPage.fragment}
78 +   ${eventListingPage.fragment}
79 +   ${storyListingPage.fragment}
80 +   ${leadershipListingPage.fragment}

81 +   ${healthServicesListingPage.fragment}
82 +   ${pressReleasesListingPage.fragment}
83 +   ${locationListingPage.fragment}
84 +   ${qaPage.fragment}
85 +   ${faqMultipleQaPage.fragment}
86 +   ${stepByStepPage.fragment}
87 +   ${mediaListImages.fragment}
88 +   ${checklistPage.fragment}
89 +   ${mediaListVideos.fragment}
90 +   ${supportResourcesDetailPage.fragment}
91 +   ${basicLandingPage.fragment}
92 +   ${nodeCampaignLandingPage.fragment}

```



nckslvn on Feb 8

Author

Member

Each module exporting an addition `.fragment` is what makes these changes backwards-compatible. The same pattern is used by the preview server



Reply...

Resolve conversation

93	93	`;
94	94	
95	95	const todayQueryVar = useTomeSync ? '' : '\$today: String!,';
144	144	
145	145	query GetAllPages(\${todayQueryVar} \$onlyPublishedContent: Boolean!) {
146	146	\${nodeQuery}
147		- \${icsFileQuery}
148		- \${sidebarQuery}
149		- \${facilitySidebarQuery}
150		- \${outreachSidebarQuery}
151		- \${alertsQuery}
152		- \${bannerAlertsQuery}
153		- \${outreachAssetsQuery}
154		- \${homePageQuery}
	147	+ \${icsFileQuery.partialQuery}
	148	+ \${sidebarQuery.partialQuery}
	149	+ \${facilitySidebarQuery.partialQuery}
	150	+ \${outreachSidebarQuery.partialQuery}
	151	+ \${alertsQuery.partialQuery}
	152	+ \${bannerAlertsQuery.partialQuery}
	153	+ \${outreachAssetsQuery.partialQuery}
	154	+ \${homePageQuery.partialQuery}
155	155	{
156	156	cmsFeatureFlags.FEATURE_ALL_HUB_SIDE_NAVS
157		- ? `\${allSideNavMachineNamesQuery}`
	157	+ ? `\${allSideNavMachineNamesQuery.partialQuery}`
158	158	: ''
159	159	}
160		- \${menuLinksQuery}
161		- \${taxonomiesQuery}
	160	+ \${menuLinksQuery.partialQuery}
	161	+ \${taxonomiesQuery.partialQuery}
162	162	}
163	163	`;
164	164	

<div> <div> <div></div> <div>60</div> <div></div> </div> <div>src/site/stages/build/drupal/graphql/GetLatestPageById.graphql.js</div> <div></div> </div>		
...	...	@@ -1,6 +1,6 @@
1	1	const landingPage = require('./landingPage.graphql');
2	2	const page = require('./page.graphql');
3		- const fragments = require('./fragments.graphql');
	3	+ const { ALL_FRAGMENTS } = require('./fragments.graphql');
4	4	const healthCareRegionPage = require('./healthCareRegionPage.graphql');
5	5	
6	6	const alertsQuery = require('./alerts.graphql');
44	44	*/
45	45	module.exports = `
46	46	
47		- \${fragments}
48		- \${landingPage}
49		- \${page}
50		- \${healthCareRegionPage}

51		-	<code>\${healthCareLocalFacilityPage}</code>
52		-	<code>\${healthCareRegionDetailPage}</code>
53		-	<code>\${pressReleasePage}</code>
54		-	<code>\${vamcOperatingStatusAndAlerts}</code>
55		-	<code>\${newsStoryPage}</code>
56		-	<code>\${eventPage}</code>
57		-	<code>\${bioPage}</code>
58		-	<code>\${vaFormPage}</code>
59		-	<code>\${nodeQa}</code>
60		-	<code>\${faqMultipleQa}</code>
61		-	<code>\${nodeStepByStep}</code>
62		-	<code>\${nodeMediaListImages}</code>
63		-	<code>\${nodeChecklist}</code>
64		-	<code>\${nodeMediaListVideos}</code>
65		-	<code>\${nodeSupportResourcesDetailPage}</code>
66		-	<code>\${nodeBasicLandingPage}</code>
67		-	<code>\${nodeCampaignLandingPage}</code>
	47	+	<code>\${ALL_FRAGMENTS}</code>
	48	+	<code>\${landingPage.fragment}</code>
	49	+	<code>\${page.fragment}</code>
	50	+	<code>\${healthCareRegionPage.fragment}</code>
	51	+	<code>\${healthCareLocalFacilityPage.fragment}</code>
	52	+	<code>\${healthCareRegionDetailPage.fragment}</code>
	53	+	<code>\${pressReleasePage.fragment}</code>
	54	+	<code>\${vamcOperatingStatusAndAlerts.fragment}</code>
	55	+	<code>\${newsStoryPage.fragment}</code>
	56	+	<code>\${eventPage.fragment}</code>
	57	+	<code>\${bioPage.fragment}</code>
	58	+	<code>\${vaFormPage.fragment}</code>
	59	+	<code>\${nodeQa.fragment}</code>
	60	+	<code>\${faqMultipleQa.fragment}</code>
	61	+	<code>\${nodeStepByStep.fragment}</code>
	62	+	<code>\${nodeMediaListImages.fragment}</code>
	63	+	<code>\${nodeChecklist.fragment}</code>
	64	+	<code>\${nodeMediaListVideos.fragment}</code>
	65	+	<code>\${nodeSupportResourcesDetailPage.fragment}</code>
	66	+	<code>\${nodeBasicLandingPage.fragment}</code>
	67	+	<code>\${nodeCampaignLandingPage.fragment}</code>
68	68		
69	69		<code>query GetLatestPageById(\$id: String!, \$today: String!, \$onlyPublishedCo</code>
70	70		<code>nodes: nodeQuery(revisions: LATEST, filter: {</code>
95	95		<code>... nodeCampaignLandingPage</code>
96	96		<code>}</code>
97	97		<code>}</code>
98		-	<code>\${icsFileQuery}</code>
99		-	<code>\${sidebarQuery}</code>
100		-	<code>\${facilitySidebarQuery}</code>
101		-	<code>\${alertsQuery}</code>
102		-	<code>\${bannerAlertsQuery}</code>
	98	+	<code>\${icsFileQuery.partialQuery}</code>
	99	+	<code>\${sidebarQuery.partialQuery}</code>
	100	+	<code>\${facilitySidebarQuery.partialQuery}</code>
	101	+	<code>\${alertsQuery.partialQuery}</code>
	102	+	<code>\${bannerAlertsQuery.partialQuery}</code>





```

64 +   query {
65 +     ${bannerAlerts}
66 +   }
67 + `;
68 +
69 + module.exports = {
70 +   partialQuery: bannerAlerts,
71 +   GetBannnerAlerts,
72 + };

```

25 src/site/stages/build/drupal/graphql/benefitListingPage.graphql.js



```

4 4  */
5 5  const entityElementsFromPages = require('./entityElementsForPages.graphql
6 6
7 - module.exports = `
7 + const benefitListingPage = `
8 8  fragment benefitListingPage on NodePublicationListing {
9 9    ${entityElementsFromPages}
10 10    changed
16 16  }
17 17  }
18 18  `;
19 +
20 + const GetNodePublicationListingPages = `
21 +
22 +   ${benefitListingPage}
23 +
24 +   query GetNodePublicationListingPages($onlyPublishedContent: Boolean!) {
25 +     nodeQuery(limit: 500, filter: {
26 +       conditions: [
27 +         { field: "status", value: ["1"], enabled: $onlyPublishedContent }
28 +         { field: "type", value: ["publication_listing"] }
29 +       ]
30 +     }) {
31 +       entities {
32 +         ... benefitListingPage
33 +       }
34 +     }
35 +   }
36 + `;
37 +
38 + module.exports = {
39 +   fragment: benefitListingPage,
40 +   GetNodePublicationListingPages,
41 + };

```

40 src/site/stages/build/drupal/graphql/bioPage.graphql.js



```

3 3  *
4 4  */
5 5  const entityElementsFromPages = require('./entityElementsForPages.graphql
6 + const { generatePaginatedQueries } = require('./individual-queries-helpers

```

```

6   7   - const { generatePaginatedQueries } = require( './individual_queries' );
7   - module.exports = `
8   + const personProfileFragment = `
9     fragment bioPage on NodePersonProfile {
10      ${entityElementsFromPages}
11      fieldNameFirst
47    changed
48    }
49  `;
51  +
52  + function getNodePersonProfilesSlice(operationName, offset, limit) {
53  +   return `
54  +     ${personProfileFragment}
55  +
56  +     query ${operationName}($onlyPublishedContent: Boolean!) {
57  +       nodeQuery(
58  +         limit: ${limit}
59  +         offset: ${offset}
60  +         sort: { field: "changed", direction: ASC }
61  +         filter: {
62  +           conditions: [
63  +             { field: "status", value: ["1"], enabled: $onlyPublishedContent },
64  +             { field: "type", value: ["person_profile"] }
65  +           ]
66  +         }) {
67  +           entities {
68  +             ... bioPage
69  +           }
70  +         }
71  +       }
72  +     `;
73  +   }
74  +
75  + function getNodePersonProfileQueries(entityCounts) {
76  +   return generatePaginatedQueries({
77  +     operationNamePrefix: 'GetNodePersonProfile',
78  +     entitiesPerSlice: 50,
79  +     totalEntities: entityCounts.data.personProfile.count,
80  +     getSlice: getNodePersonProfilesSlice,
81  +   });
82  + }
83  +
84  + module.exports = {
85  +   fragment: personProfileFragment,
86  +   getNodePersonProfileQueries,
87  + };

```

25 src/site/stages/build/drupal/graphql/eventListingPage.graphql.js

```

4   4   */
5   5   const entityElementsFromPages = require( './entityElementsForPages.graphql' );
6   6
7   - module.exports = `
7   + const eventListingPage = `

```

```

74 74 `;
75 +
76 + const GetNodeEventListingPage = `
77 +
78 +   ${eventListingPage}
79 +
80 +   query GetNodeEventListingPage($onlyPublishedContent: Boolean!) {
81 +     nodeQuery(limit: 500, filter: {
82 +       conditions: [
83 +         { field: "status", value: ["1"], enabled: $onlyPublishedContent }
84 +         { field: "type", value: ["event_listing"] }
85 +       ]
86 +     }) {
87 +       entities {
88 +         ... eventListingPage
89 +       }
90 +     }
91 +   }
92 + `;
93 +
94 + module.exports = {
95 +   fragment: eventListingPage,
96 +   GetNodeEventListingPage,
97 + };

```

43 src/site/stages/build/drupal/graphql/eventPage.graphql.js

```

4 4 */
5 5 const entityElementsFromPages = require('./entityElementsForPages.graphql
6 6
7 - module.exports = `
8 + const { generatePaginatedQueries } = require('../individual-queries-helpe
9 + const eventPage = `
10 10 fragment eventPage on NodeEvent {
11 11   ${entityElementsFromPages}
12 12   changed
43 45   value
44 46   endValue
45 47   timezone
46 - }
48 + }
47 49   fieldAddress {
48 50     addressLine1
49 51     addressLine2
77 79   fieldAdditionalInformationAbo {processed}
78 80   }
79 81 `;

```

```

82 +
83 + function getNodeEventSlice(operationName, offset, limit = 100) {
84 +   return `
85 +     ${eventPage}
86 +
87 +     query ${operationName}($onlyPublishedContent: Boolean!) {
88 +       nodeQuery(
89 +         limit: ${limit}
90 +         offset: ${offset}
91 +         sort: { field: "field_datetime_range_timezone.end_value", direction: "DESC" }
92 +         filter: {
93 +           conditions: [
94 +             { field: "status", value: ["1"], enabled: $onlyPublishedContent }
95 +             { field: "type", value: ["event"] }
96 +           ]
97 +         }) {
98 +           entities {
99 +             ... eventPage
100 +           }
101 +         }
102 +       }
103 +     `;
104 +   }
105 +
106 + function getNodeEventQueries(entityCounts) {
107 +   return generatePaginatedQueries({
108 +     operationNamePrefix: 'GetNodeEvents',
109 +     entitiesPerSlice: 50,
110 +     totalEntities: entityCounts.data.event.count,
111 +     getSlice: getNodeEventSlice,
112 +   });
113 + }

```



nckslvn on Feb 8

Author

Member

`getNodeEventPages` was expensive, so I decided to try making it more efficient by splitting it into two. The second query gets all events that ended before the start of this month, with the hypothesis being that that event data is unlikely to change and therefore a good candidate to be cached. Then `getNodeEventPages` just gets whatever events are left over. Not certain that it will work yet but seems promising so far.



Reply...

Resolve conversation

```

114 +
115 + module.exports = {
116 +   fragment: eventPage,
117 +   getNodeEventQueries,
118 + };

```

...	...	@@ -1,7 +1,9 @@
	1	+ const fragments = require('./fragments.graphql');
	2	+
1	3	const entityElementsFromPages = require('./entityElementsForPages.graphql
2	4	
3	5	// faq_multiple_q_a, 6898
4		- module.exports = `
	6	+ const faqMultipleQA = `
5	7	fragment faqMultipleQA on NodeFaqMultipleQA {
6	8	\${entityElementsFromPages}
7	9	entityBundle
91	93	}
92	94	}
93	95	`;
	96	+
	97	+ const GetNodeMultipleQaPages = `
	98	+   \${fragments.richTextCharLimit1000}
	99	+   \${fragments.reactWidget}
100		+   \${fragments.alertParagraph}
101		+   \${fragments.alertParagraphSingle}
102		+   \${fragments.button}
103		+   \${fragments.contactInformation}
104		+   \${fragments.supportService}
105		+   \${fragments.linkTeaser}
106		+   \${fragments.termLcCategory}
107		+   \${fragments.audienceTopics}
108		+   \${fragments.emailContact}
109		+   \${fragments.phoneNumber}
110		+   \${fragments.audienceBeneficiaries}
111		+   \${fragments.audienceNonBeneficiaries}
112		+   \${fragments.termTopics}
113		+
114		+   \${faqMultipleQA}
115		+
116		+ query GetNodeMultipleQaPages(\$onlyPublishedContent: Boolean!) {
117		+   nodeQuery(limit: 1000, filter: {
118		+     conditions: [
119		+       { field: "status", value: ["1"], enabled: \$onlyPublishedContent }
120		+       { field: "type", value: ["faq_multiple_q_a"] }
121		+     ]
122		+   }) {
123		+     entities {
124		+       ... faqMultipleQA
125		+     }
126		+   }
127		+ }
128		+ `;
129		+
130		+ module.exports = {
131		+   fragment: faqMultipleQA,
132		+   GetNodeMultipleQaPages,
133		+ };

13 src/site/stages/build/drupal/graphql/file-fragments/ics.file.graphql.js

```

2      2      * Queries to get all .ics files
3      3      * To execute, run this query at http://staging.va.agile6.com/graphql/exp
4      4      */
5      - module.exports = `
6      + const icsFiles = `
7      icsFiles: fileQuery(filter: {
8      conditions: [
9      { field: "filemime", value: "text/calendar"}
10     ]
11   }
12   }
13   `;
14
15   +
16   + const GetIcsFiles = `
17   +   query {
18   +     ${icsFiles}
19   +   }
20   + `;
21   +
22   + module.exports = {
23   +   partialQuery: icsFiles,
24   +   GetIcsFiles,
25   + };

```

13 src/site/stages/build/drupal/graphql/file-fragments/outreachAssets.graphql.js

```

...    ...    @@ -1,4 +1,4 @@
1      - module.exports = `
2      + const outreachAssets = `
3      outreachAssets: nodeQuery(filter: {conditions: [{field: "type", value:
4      entities {
5      ... on NodeOutreachAsset {
6      }
7      }
8      `;
9
10     +
11     + const GetOutreachAssets = `
12     +   query($onlyPublishedContent: Boolean!) {
13     +     ${outreachAssets}
14     +   }
15     + `;
16     +
17     + module.exports = {
18     +   partialQuery: outreachAssets,
19     +   GetOutreachAssets,
20     + };

```

37 src/site/stages/build/drupal/graphql/fragments.graphql.js

```

32      } = require('./paragraph-fragments/listOfLinkTeasers.paragraph.graphql');
33      const { promo } = require('./block-fragments/promo-block.graphql');

```

33	33	<code>const { promo } = require('../block-fragments/promo.block.graphql');</code>
34	34	
35		<code>- module.exports = `</code>
	35	<code>+ const ALL_FRAGMENTS = `</code>
36	36	<code>  \${administration}</code>
37	37	<code>  \${alertParagraphSingle}</code>
38	38	<code>  \${alertParagraph}</code>
65	65	<code>  \${termTopics}</code>
66	66	<code>  \${wysiwyg}</code>
67	67	<code>`;</code>
	68	<code>+ </code>
	69	<code>+ module.exports = {</code>
	70	<code>+   ALL_FRAGMENTS,</code>
	71	<code>+   administration,</code>
	72	<code>+   alertParagraphSingle,</code>
	73	<code>+   alertParagraph,</code>
	74	<code>+   alert,</code>
	75	<code>+   audienceBeneficiaries,</code>
	76	<code>+   audienceNonBeneficiaries,</code>
	77	<code>+   audienceTopics,</code>
	78	<code>+   button,</code>
	79	<code>+   collapsiblePanel,</code>
	80	<code>+   contactInformation,</code>
	81	<code>+   downloadableFile,</code>
	82	<code>+   emailContact,</code>
	83	<code>+   embeddedImage,</code>
	84	<code>+   linkTeaser,</code>
	85	<code>+   listOfLinkTeasers,</code>
	86	<code>+   listsOfLinks,</code>
	87	<code>+   numberCallout,</code>
	88	<code>+   phoneNumber,</code>
	89	<code>+   process,</code>
	90	<code>+   promo,</code>
	91	<code>+   qaSection,</code>
	92	<code>+   qa,</code>
	93	<code>+   reactWidget,</code>
	94	<code>+   richTextCharLimit1000,</code>
	95	<code>+   spanishSummary,</code>
	96	<code>+   staffProfile,</code>
	97	<code>+   supportService,</code>
	98	<code>+   table,</code>
	99	<code>+   termLcCategory,</code>
	100	<code>+   termTopics,</code>
	101	<code>+   wysiwyg,</code>
	102	<code>+ };</code>

<div> <div>▼ ⓘ 48</div> <div>src/site/stages/build/drupal/graphql/healthCareLocalFacilityPage.graphql.js</div> </div>		
...	...	<code>@@ -1,9 +1,12 @@</code>
	1	<code>+ const fragments = require('../fragments.graphql');</code>
1	2	<code>const entityElementsFromPages = require('../entityElementsForPages.graphql');</code>
2	3	<code>const socialMediaFields = require('../facilities-fragments/healthCareSocialMediaFields.graphql');</code>



3	4	<code>const serviceLocation = require('./paragraph-fragments/serviceLocation.pa</code>
4	5	<code>const appointmentItems = require('./file-fragments/appointmentItems.graph</code>
5	6	
6		<code>- module.exports = `</code>
	7	<code>+ const { generatePaginatedQueries } = require('./individual-queries-helpe</code>
	8	<code>+ </code>
	9	<code>+ const healthCareLocalFacilityPageFragment = `</code>
7	10	<code>fragment healthCareLocalFacilityPage on NodeHealthCareLocalFacility {</code>
8	11	<code>  \${entityElementsFromPages}</code>
9	12	<code>  changed</code>
119	122	<code>  }</code>
120	123	<code>}</code>
121	124	<code>`;</code>
	125	<code>+ </code>
	126	<code>+ function getNodeHealthCareLocalFacilityPagesSlice(</code>
	127	<code>+   operationName,</code>
	128	<code>+   offset,</code>
	129	<code>+   limit,</code>
	130	<code>+ ) {</code>
	131	<code>+   return `</code>
	132	<code>+     \${fragments.listOfLinkTeasers}</code>
	133	<code>+     \${fragments.linkTeaser}</code>
	134	<code>+     \${healthCareLocalFacilityPageFragment}</code>
	135	<code>+ </code>
	136	<code>+     query \${operationName}(\$onlyPublishedContent: Boolean!) {</code>
	137	<code>+       nodeQuery(</code>
	138	<code>+         limit: \${limit}</code>
	139	<code>+         offset: \${offset}</code>
	140	<code>+         sort: { field: "changed", direction: ASC }</code>
	141	<code>+         filter: {</code>
	142	<code>+           conditions: [</code>
	143	<code>+             { field: "status", value: ["1"], enabled: \$onlyPublishedConten</code>
	144	<code>+             { field: "type", value: ["health_care_local_facility"]} ]</code>
	145	<code>+           }</code>
	146	<code>+         }) {</code>
	147	<code>+           entities {</code>
	148	<code>+             ... healthCareLocalFacilityPage</code>
	149	<code>+           }</code>
	150	<code>+         }</code>
	151	<code>+       }</code>
	152	<code>+     `;</code>
	153	<code>+ }</code>
	154	<code>+ </code>
	155	<code>+ function getNodeHealthCareLocalFacilityPageQueries(entityCounts) {</code>
	156	<code>+   return generatePaginatedQueries({</code>
	157	<code>+     operationNamePrefix: 'GetNodeHealthCareLocalFacilityPages',</code>
	158	<code>+     entitiesPerSlice: 50,</code>
	159	<code>+     totalEntities: entityCounts.data.healthCareLocalFacility.count,</code>
	160	<code>+     getSlice: getNodeHealthCareLocalFacilityPagesSlice,</code>
	161	<code>+   });</code>
	162	<code>+ }</code>
	163	<code>+ </code>
	164	<code>+ module.exports = {</code>
	165	<code>+   fragment: healthCareLocalFacilityPageFragment,</code>

```
166 +   getNodeHealthCareLocalFacilityPageQueries,  
167 + };
```

59

src/site/stages/build/drupal/graphql/healthCareRegionDetailPage.graphql.js

```
3      3      * Example: /pittsburgh_health_care_system  
4      4      */  
5      5  
6      6      + const fragments = require('./fragments.graphql');  
7      7      +  
8      8      const {  
9      9      FIELD_RELATED_LINKS,  
10     10     } = require('./paragraph-fragments/listOfLinkTeasers.paragraph.graphql');  
23    25     const MEDIA_PARAGRAPH = '... embeddedImage';  
24    26     const entityElementsFromPages = require('./entityElementsForPages.graphql');  
25    27  
26    26     - module.exports = `  
28    28     + const { generatePaginatedQueries } = require('../individual-queries-helper');  
29    29     +  
30    30     + const healthCareRegionDetailPage = `  
31    31     fragment healthCareRegionDetailPage on NodeHealthCareRegionDetailPage {  
32    32     title  
33    33     ${entityElementsFromPages}  
70    74     }  
71    75     }  
72    76     `;  
77    77     +  
78    78     + function getNodeHealthCareRegionDetailPageSlice(operationName, offset, limit) {  
79    79     +   return `  
80    80     +     ${fragments.wysiwyg}  
81    81     +     ${fragments.staffProfile}  
82    82     +     ${fragments.collapsiblePanel}  
83    83     +     ${fragments.process}  
84    84     +     ${fragments.qaSection}  
85    85     +     ${fragments.qa}  
86    86     +     ${fragments.listOfLinkTeasers}  
87    87     +     ${fragments.reactWidget}  
88    88     +     ${fragments.numberCallout}  
89    89     +     ${fragments.table}  
90    90     +     ${fragments.alertParagraph}  
91    91     +     ${fragments.downloadableFile}  
92    92     +     ${fragments.embeddedImage}  
93    93     +     ${fragments.linkTeaser}  
94    94     +     ${fragments.alert}  
95    95     +  
96    96     +     ${healthCareRegionDetailPage}  
97    97     +  
98    98     +     query ${operationName}($onlyPublishedContent: Boolean!) {  
99    99     +       nodeQuery(  
100   100     +         limit: ${limit}  
101   101     +         offset: ${offset}  
102   102     +         sort: { field: "changed", direction: ASC }  
103   103     +         filter: {
```

```

104 +         conditions: [
105 +             { field: "status", value: ["1"], enabled: $onlyPublishedContent },
106 +             { field: "type", value: ["health_care_region_detail_page"] }
107 +         ]
108 +     }) {
109 +         entities {
110 +             ... healthCareRegionDetailPage
111 +         }
112 +     }
113 + }
114 + `;
115 + }
116 +
117 + function getNodeHealthCareRegionDetailPageQueries(entityCounts) {
118 +     return generatePaginatedQueries({
119 +         operationNamePrefix: 'GetNodeHealthCareRegionDetailPage',
120 +         entitiesPerSlice: 50,
121 +         totalEntities: entityCounts.data.healthCareRegionDetailPage.count,
122 +         getSlice: getNodeHealthCareRegionDetailPageSlice,
123 +     });
124 + }
125 +
126 + module.exports = {
127 +     fragment: healthCareRegionDetailPage,
128 +     getNodeHealthCareRegionDetailPageQueries,
129 + };

```

✓ ⓘ 44 src/site/stages/build/drupal/graphql/healthCareRegionPage.graphql.js



2	2	* The top-level page for a health care region.
3	3	* Example: /pittsburgh_health_care_system
4	4	*/
	5	+ const fragments = require('./fragments.graphql');
5	6	const entityElementsFromPages = require('./entityElementsForPages.graphql');
6	7	const healthCareLocalFacilities = require('./facilities-fragments/healthCareLocalFacilities.graphql');
7	8	const healthCareRegionHealthServices = require('./facilities-fragments/healthCareRegionHealthServices.graphql');
8	9	const healthCareRegionNewsStories = require('./facilities-fragments/healthCareRegionNewsStories.graphql');
9	10	const healthCareRegionEvents = require('./facilities-fragments/healthCareRegionEvents.graphql');
10	11	
11		- module.exports = `
	12	+ const { generatePaginatedQueries } = require('./individual-queries-helper');
	13	+
	14	+ const healthCareRegionPageFragment = `
12	15	fragment healthCareRegionPage on NodeHealthCareRegionPage {
13	16	\${entityElementsFromPages}
14	17	\${healthCareRegionNewsStories}
190	193	}
191	194	}
192	195	`;
	196	+
	197	+ function getNodeHealthCareRegionPageSlice(operationName, offset, limit) {
	198	+     return `

```

199 +     ${fragments.linkTeaser}
200 +     ${fragments.listOfLinkTeasers}
201 +     ${healthCareRegionPageFragment}
202 +
203 +     query ${operationName}($today: String!, $onlyPublishedContent: Boolean!){
204 +         nodeQuery(
205 +             limit: ${limit}
206 +             offset: ${offset}
207 +             sort: { field: "title", direction: ASC }
208 +             filter: {
209 +                 conditions: [
210 +                     { field: "status", value: ["1"], enabled: $onlyPublishedContent }
211 +                     { field: "type", value: ["health_care_region_page"] }
212 +                 ]
213 +             }) {
214 +                 entities {
215 +                     ... healthCareRegionPage
216 +                 }
217 +             }
218 +         }
219 +     `;
220 + }
221 +
222 + function getNodeHealthCareRegionPageQueries(entityCounts) {
223 +     return generatePaginatedQueries({
224 +         operationNamePrefix: 'GetNodeHealthCareRegionPage',
225 +         entitiesPerSlice: 5,
226 +         totalEntities: entityCounts.data.healthCareRegionPage.count,
227 +         getSlice: getNodeHealthCareRegionPageSlice,
228 +     });
229 + }
230 +
231 + module.exports = {
232 +     fragment: healthCareRegionPageFragment,
233 +     getNodeHealthCareRegionPageQueries,
234 + };

```

▼ ⓘ 40

src/site/stages/build/drupal/graphql/healthServicesListingPage.graphql.js

4	4	*/
5	5	const entityElementsFromPages = require('./entityElementsForPages.graphql')
6	6	
7	7	- module.exports = `
	7	+ const { generatePaginatedQueries } = require('../individual-queries-helper')
	8	+
	9	+ const healthServicesListingPage = `
8	10	fragment healthServicesListingPage on NodeHealthServicesListing {
9	11	\${entityElementsFromPages}
10	12	title
110	112	}
111	113	}
112	114	`;
	115	+

```

116 + function getNodeHealthServicesListingPages(operationName, offset, limit)
117 +   return `
118 +     ${healthServicesListingPage}
119 +     query ${operationName}($onlyPublishedContent: Boolean!) {
120 +       nodeQuery(
121 +         limit: ${limit}
122 +         offset: ${offset}
123 +         sort: { field: "title", direction: ASC }
124 +         filter: {
125 +           conditions: [
126 +             { field: "status", value: ["1"], enabled: $onlyPublishedContent }
127 +             { field: "type", value: ["health_services_listing"] }
128 +           ]
129 +         }) {
130 +           entities {
131 +             ... healthServicesListingPage
132 +           }
133 +         }
134 +       }
135 +     `;
136 +   }
137 +
138 + function getNodeHealthServicesListingPageQueries(entityCounts) {
139 +   return generatePaginatedQueries({
140 +     operationNamePrefix: 'GetNodeHealthServicesListingPage',
141 +     entitiesPerSlice: 5,
142 +     totalEntities: entityCounts.data.healthServicesListing.count,
143 +     getSlice: getNodeHealthServicesListingPages,
144 +   });
145 + }
146 +
147 + module.exports = {
148 +   fragment: healthServicesListingPage,
149 +   getNodeHealthServicesListingPageQueries,
150 + };

```

11 src/site/stages/build/drupal/graphql/homePage.graphql.js

```

83   }
84   `;
85
86 - module.exports = query;
86 + const GetHomepage = `
87 +   query {
88 +     ${query}
89 +   }
90 + `;
91 +
92 + module.exports = {
93 +   partialQuery: query,
94 +   GetHomepage,
95 + };

```

... @@ -1,3 +1,11 @@

```

1 + const fragments = require('./fragments.graphql');
2 +
3 + // String Helpers
4 + const {
5 +   updateQueryString,
6 +   queryParamsToBeChanged,
7 + } = require('./../../../../utilities/stringHelpers');
8 +
9   const entityElementsFromPages = require('./entityElementsForPages.graphql');
10  const { FIELD_PROMO } = require('./block-fragments/promo.block.graphql');
11  const {
12    */
13    const ADMIN = '...administration';
14  - module.exports = `
15  + const landingPageFragment = `
16
17    fragment landingPage on NodeLandingPage {
18      ${entityElementsFromPages}
19      ${ADMIN}
20    }
21  `;
22
23  +
24  + let regString = '';
25  + queryParamsToBeChanged.forEach(param => {
26  +   regString += `${param}|`;
27  + });
28  +
29  + const regex = new RegExp(`${regString}`, 'g');
30  + const landingPageFragmentModified = landingPageFragment.replace(
31  +   regex,
32  +   updateQueryString,
33  + );
34  +
35  +
36  +
37  +
38  +
39  +
40  +
41  +
42  +
43  +
44  +
45  +
46  +
47  +
48  +
49  +
50  +
51  +
52  +
53  +
54  +
55  +
56  +
57  +
58  +
59  +
60  +
61  +
62  +
63  +
64  +
65  +
66  +
67  +
68  +
69  +
70  +
71  +
72  +
73  +
74  +
75  +
76  +
77  +
78  +
79  +
80  +
81  +
82  +
83  +
84  +
85  +
86  +
87  +
88  +
89  +
90  +
91  +
92  +
93  +
94  +
95  +
96  +
97  +
98  +
99  +
100 +

```



nckslvn on Feb 8 • edited

Author

Member

I do not understand why we have these string manipulations versus just using the correct value in the actual query, but without it I encountered a GraphQL error so from that I discovered how it is used. I assume there is some CMS-internal context/knowledge about this.

Same with this query, <https://github.com/department-of-veterans-affairs/vets-website/pull/15974/files#diff-e7722a353cb5ab1ec4123b9e2fa33ec57e253d1f8be2942882b1303e0b6702daR54>.



Reply...

Resolve conversation

81 + const GetNodeLandingPages = `

```

82 +   ${fragments.administration}
83 +   ${fragments.alert}
84 +   ${fragments.promo}
85 +   ${fragments.listOfLinkTeasers}
86 +   ${fragments.linkTeaser}
87 +
88 +   ${landingPageFragmentModified}
89 +
90 +   query GetNodeLandingPages($onlyPublishedContent: Boolean!) {
91 +     nodeQuery(limit: 1000, filter: {
92 +       conditions: [
93 +         { field: "status", value: ["1"], enabled: $onlyPublishedContent }
94 +         { field: "type", value: ["landing_page"] }
95 +       ]
96 +     }) {
97 +       entities {
98 +         ... landingPage
99 +       }
100 +     }
101 +   }
102 + `;
103 +
104 + module.exports = {
105 +   fragment: landingPageFragment,
106 +   modifiedFragment: landingPageFragmentModified,
107 +   GetNodeLandingPages,
108 + };

```

✓ ⓘ 25 src/site/stages/build/drupal/graphql/leadershipListingPage.graphql.js



4	4	*/
5	5	const entityElementsFromPages = require('./entityElementsForPages.graphql
6	6	
7		- module.exports = `
	7	+ const leadershipListingPage = `
8	8	fragment leadershipListingPage on NodeLeadershipListing {
9	9	\${entityElementsFromPages}
10	10	title
115	115	}
116	116	}
117	117	`;
	118	+
	119	+ const GetNodeLeadershipListingPages = `
	120	+
	121	+   \${leadershipListingPage}
	122	+
	123	+   query GetNodeLeadershipListingPages(\$onlyPublishedContent: Boolean!) {
	124	+     nodeQuery(limit: 500, filter: {
	125	+       conditions: [
	126	+         { field: "status", value: ["1"], enabled: \$onlyPublishedContent }
	127	+         { field: "type", value: ["leadership_listing"] }
	128	+       ]
	129	+     }) {

```

130 +     entities {
131 +         ... leadershipListingPage
132 +     }
133 + }
134 + }
135 + `;
136 +
137 + module.exports = {
138 +   fragment: leadershipListingPage,
139 +   GetNodeLeadershipListingPages,
140 + };

```

✓ ⓘ 25 src/site/stages/build/drupal/graphql/locationsListingPage.graphql.js



```

5 5 const entityElementsFromPages = require('./entityElementsForPages.graphql
6 6 const healthCareLocalFacilities = require('./facilities-fragments/healthC
7 7
8 8 - module.exports = `
8 8 + const locationListingPage = `
9 9   fragment locationListingPage on NodeLocationsListing {
10 10     ${entityElementsFromPages}
11 11     title
26 26   }
27 27 }
28 28 `;
29 +
30 + const GetNodeLocationsListingPages = `
31 +
32 +   ${locationListingPage}
33 +
34 +   query GetNodeLocationsListingPages($onlyPublishedContent: Boolean!) {
35 +     nodeQuery(limit: 500, filter: {
36 +       conditions: [
37 +         { field: "status", value: ["1"], enabled: $onlyPublishedContent }
38 +         { field: "type", value: ["locations_listing"] }
39 +       ]
40 +     }) {
41 +       entities {
42 +         ... locationListingPage
43 +       }
44 +     }
45 +   }
46 + `;
47 +
48 + module.exports = {
49 +   fragment: locationListingPage,
50 +   GetNodeLocationsListingPages,
51 + };

```

✓ ⓘ 13 ...te/stages/build/drupal/graphql/navigation-

fragments/allSideNavMachineNames.nav.graphql.js



2	2	* A query to get all benefits hub sidebar machine names
3	3	*
4	4	*/
5		- module.exports = `
	5	+ const partialQuery = `
6	6	allSideNavMachineNamesQuery: siteMenus
7	7	`;
	8	+
	9	+ const GetAllSideNavMachineNames = `
10		+ query {
11		+ \${partialQuery}
12		+ }
13		+ `;
14		+
15		+ module.exports = {
16		+ partialQuery,
17		+ GetAllSideNavMachineNames,
18		+ };

28 src/site/stages/build/drupal/graphql/navigation-fragments/facilitySidebar.nav.graphql.js

160	160	}
161	161	`;
162	162	
	163	+ const VaFacilitySidebars = {};
163	164	let compiledQuery = `;
164	165	
165	166	FACILITY_MENU_NAMES.forEach(facilityMenuName => {
166		- compiledQuery += `
167		- \${camelize(
168		- facilityMenuName,
169		- )}FacilitySidebarQuery: menuByName(name: "\${facilityMenuName}")
170		- \${FACILITY_SIDEBAR_QUERY}
171		- }
172		- `;
	167	+ const facilityMenuNameCamel = camelize(facilityMenuName);
	168	+ const operationName = `\${facilityMenuNameCamel}FacilitySidebarQuery`;
	169	+ const nextSidebar = `
170		+ \${operationName}: menuByName(name: "\${facilityMenuName}") {
171		+ \${FACILITY_SIDEBAR_QUERY}
172		+ }
173		+ `;
174		+
175		+ compiledQuery += nextSidebar;
176		+
177		+ VaFacilitySidebars[`\${GetFacilitySidebar__\${operationName}`}] = `
178		+ query {
179		+ \${nextSidebar}
180		+ }
181		+ `;
173	182	});
174	183	
175		- module.exports = compiledQuery;

```

184 + module.exports = {
185 +   partialQuery: compiledQuery,
186 +   VaFacilitySidebars,
187 + };

```

✓ ⓘ 13 src/site/stages/build/drupal/graphql/navigation-  
fragments/menuLinks.nav.graphql.js

```

6 6
7 7   const headerQuery = require('./header.nav.graphql');
8 8
9 9   - module.exports = `
10 10 + const menuLinkContentQuery = `
11 11     menuLinkContentQuery(limit: 5000, filter: {conditions: [{field: "enabled
12 12         entities {
13 13             entityId
14 14         }
15 15     }
16 16 }
17 17 `;
18 +
19 + const GetMenuLinks = `
20 +   query {
21 +     ${menuLinkContentQuery}
22 +   }
23 + `;
24 +
25 + module.exports = {
26 +   partialQuery: menuLinkContentQuery,
27 +   GetMenuLinks,
28 + };

```

✓ ⓘ 13 src/site/stages/build/drupal/graphql/navigation-  
fragments/outreachSidebar.nav.graphql.js

```

5 5
6 6   const MENU_NAME = 'outreach-and-events';
7 7
8 8   - module.exports = `
9 9 + const outreachSidebarQuery = `
10 10     outreachSidebarQuery: menuByName(name: "${MENU_NAME}") {
11 11         name
12 12         description
13 13     }
14 14 }
15 15 `;
16 +
17 + const GetOutreachSidebar = `
18 +   query {
19 +     ${outreachSidebarQuery}
20 +   }
21 + `;
22 +

```

```

69 + module.exports = {
70 +   partialQuery: outreachSidebarQuery,
71 +   GetOutreachSidebar,
72 + };

```

17 src/site/stages/build/drupal/graphql/navigation-fragments/sidebar.nav.graphql.js

```

69 69 `;
70 70 }
71 71
72 72 + let partialQuery = null;
73 73 +
74 74 if (cmsFeatureFlags.FEATURE_ALL_HUB_SIDE_NAVS && hubNavNames !== null) {
75 75   let compiledQuery = '';
76 76   hubNavNames.forEach(navName => {
78 80     }
79 81     `;
80 82   });
81 81 - module.exports = compiledQuery;
83 83 + partialQuery = compiledQuery;
84 84 } else {
83 83 - module.exports = `
85 85 + partialQuery = `
86 86   burialsAndMemorialsBenefQuery: ${queryFilter(
87 87     'burials-and-memorials-benef',
88 88   )} {
126 128   }
127 129 `;
128 130 }
131 131 +
132 132 + const GetSidebars = `
133 133 +   query {
134 134 +     ${partialQuery}
135 135 +   }
136 136 + `;
137 137 +
138 138 + module.exports = {
139 139 +   partialQuery,
140 140 +   GetSidebars,
141 141 + };

```

24 src/site/stages/build/drupal/graphql/newStoryPage.graphql.js

```

4 4 */
5 5 const entityElementsFromPages = require('./entityElementsForPages.graphql
6 6
7 7 - module.exports = `
7 7 + const newStoryPage = `
8 8   fragment newStoryPage on NodeNewsStory {
9 9     ${entityElementsFromPages}
10 10     promote
46 46   }

```

```

4 / 4 /      }
48 48      `;

49 +
50 + const GetNodeNewsStoryPages = `
51 +   ${newsStoryPage}
52 +
53 +   query GetNodeNewsStoryPages($onlyPublishedContent: Boolean!) {
54 +     nodeQuery(limit: 1000, filter: {
55 +       conditions: [
56 +         { field: "status", value: ["1"], enabled: $onlyPublishedContent }
57 +         { field: "type", value: ["news_story"] }
58 +       ]
59 +     }) {
60 +       entities {
61 +         ... newsStoryPage
62 +       }
63 +     }
64 +   }
65 + `;
66 +
67 + module.exports = {
68 +   fragment: newsStoryPage,
69 +   GetNodeNewsStoryPages,
70 + };

```

41 src/site/stages/build/drupal/graphql/nodeBasicLandingPage.graphql.js



... @@ -1,6 +1,7 @@

```

1 + const fragments = require('./fragments.graphql');
2   const entityElementsFromPages = require('./entityElementsForPages.graphql');
3 - module.exports = `
4 + const nodeBasicLandingPage = `
5   fragment nodeBasicLandingPage on NodeBasicLandingPage {
6     ${entityElementsFromPages}
7     title
43 44     fieldTableOfContentsBoolean
44 45   }
45 46 `;

47 +
48 + const GetNodeBasicLandingPage = `
49 +
50 +   ${fragments.linkTeaser}
51 +   ${fragments.listOfLinkTeasers}
52 +   ${fragments.listsOfLinks}
53 +   ${fragments.wysiwyg}
54 +   ${fragments.collapsiblePanel}
55 +   ${fragments.process}
56 +   ${fragments.qaSection}
57 +   ${fragments.qa}
58 +   ${fragments.reactWidget}
59 +   ${fragments.spanishSummary}

```

```

60 +   ${fragments.alertParagraph}
61 +   ${fragments.table}
62 +   ${fragments.downloadableFile}
63 +   ${fragments.embeddedImage}
64 +   ${fragments.numberCallout}
65 +
66 +   ${nodeBasicLandingPage}
67 +
68 +   query GetNodeBasicLandingPage($onlyPublishedContent: Boolean!) {
69 +     nodeQuery(limit: 100, filter: {
70 +       conditions: [
71 +         { field: "status", value: ["1"], enabled: $onlyPublishedContent }
72 +         { field: "type", value: ["basic_landing_page"] }
73 +       ]
74 +     }) {
75 +       entities {
76 +         ... nodeBasicLandingPage
77 +       }
78 +     }
79 +   }
80 + `;
81 + module.exports = {
82 +   fragment: nodeBasicLandingPage,
83 +   GetNodeBasicLandingPage,
84 + };

```

✓ 34

src/site/stages/build/drupal/graphql/nodeCampaignLandingPage.graphql.js

```

...   ...   @@ -1,6 +1,10 @@
1   + const fragments = require('./fragments.graphql');
2   + const {
3   +   modifiedFragment: landingPageFragment,
4   + } = require('./landingPage.graphql');
1   5   const entityElementsFromPages = require('./entityElementsForPages.graphql
2   6
3   - module.exports = `
7   + const nodeCampaignLandingPage = `
4   8   fragment nodeCampaignLandingPage on NodeCampaignLandingPage {
5   9     ${entityElementsFromPages}
6   10    changed
528 532    }
529 533  }
530 534  `;
535  +
536  + const GetCampaignLandingPages = `
537  +   ${fragments.button}
538  +   ${fragments.promo}
539  +   ${fragments.listOfLinkTeasers}
540  +   ${fragments.linkTeaser}
541  +   ${fragments.alert}
542  +   ${fragments.administration}
543  +   ${landingPageFragment}

```

```

543 +   ${landingPageFragment}
544 +   ${nodeCampaignLandingPage}
545 +
546 +   query GetCampaignLandingPages($onlyPublishedContent: Boolean!) {
547 +     nodeQuery(limit: 100, filter: {
548 +       conditions: [
549 +         { field: "status", value: ["1"], enabled: $onlyPublishedContent }
550 +         { field: "type", value: ["campaign_landing_page"] }
551 +       ]
552 +     }) {
553 +       entities {
554 +         ... nodeCampaignLandingPage
555 +       }
556 +     }
557 +   }
558 + `;
559 + module.exports = {
560 +   fragment: nodeCampaignLandingPage,
561 +   GetCampaignLandingPages,
562 + };

```

39 src/site/stages/build/drupal/graphql/nodeChecklist.graphql.js

```

...   ...   @@ -1,6 +1,7 @@
1     + const fragments = require('./fragments.graphql');
2     + const entityElementsFromPages = require('./entityElementsForPages.graphql');
3     - const fragment = `
4     + const nodeChecklist = `
5     + fragment nodeChecklist on NodeChecklist {
6     +   ${entityElementsFromPages}
7     +   entityBundle
82    83   }
83    84   `;
84    85
85    - module.exports = fragment;
86    + const GetNodeChecklist = `
87    +   ${fragments.alertParagraph}
88    +   ${fragments.alertParagraphSingle}
89    +   ${fragments.button}
90    +   ${fragments.contactInformation}
91    +   ${fragments.supportService}
92    +   ${fragments.linkTeaser}
93    +   ${fragments.termLcCategory}
94    +   ${fragments.audienceTopics}
95    +   ${fragments.emailContact}
96    +   ${fragments.phoneNumber}
97    +   ${fragments.audienceBeneficiaries}
98    +   ${fragments.audienceNonBeneficiaries}
99    +   ${fragments.termTopics}
100   +
101   +   ${nodeChecklist}
102   +
103   +   query GetNodeChecklist($onlyPublishedContent: Boolean!) {
104   +     nodeQuery(limit: 100, filter: {

```

```

104 +     nodeQuery(limit: 1000, filter: {
105 +         conditions: [
106 +             { field: "status", value: ["1"], enabled: $onlyPublishedContent }
107 +             { field: "type", value: ["checklist"] }
108 +         ]
109 +     }) {
110 +         entities {
111 +             ... nodeChecklist
112 +         }
113 +     }
114 + }
115 + `;
116 +
117 + module.exports = {
118 +     fragment: nodeChecklist,
119 +     GetNodeChecklist,
120 + };

```

39 src/site/stages/build/drupal/graphql/nodeMediaListImages.graphql.js



... @@ -1,6 +1,7 @@

```

1 + const fragments = require('./fragments.graphql');
2   const entityElementsFromPages = require('./entityElementsForPages.graphql');
3 - const fragment = `
4 + const nodeMediaListImages = `
5   fragment nodeMediaListImages on NodeMediaListImages {
6     ${entityElementsFromPages}
7     entityBundle
88   }
89   `;
90
91 - module.exports = fragment;
92 + const GetNodeMediaListImages = `
93 +   ${fragments.alertParagraph}
94 +   ${fragments.alertParagraphSingle}
95 +   ${fragments.button}
96 +   ${fragments.contactInformation}
97 +   ${fragments.supportService}
98 +   ${fragments.linkTeaser}
99 +   ${fragments.termLcCategory}
100 +   ${fragments.audienceTopics}
101 +   ${fragments.emailContact}
102 +   ${fragments.phoneNumber}
103 +   ${fragments.audienceBeneficiaries}
104 +   ${fragments.audienceNonBeneficiaries}
105 +   ${fragments.termTopics}
106 +
107 +   ${nodeMediaListImages}
108 +
109 +   query GetNodeMediaListImages($onlyPublishedContent: Boolean!) {
110 +     nodeQuery(limit: 1000, filter: {

```

```

111 +     conditions: [
112 +       { field: "status", value: ["1"], enabled: $onlyPublishedContent }
113 +       { field: "type", value: ["media_list_images"] }
114 +     ]
115 +   }) {
116 +     entities {
117 +       ... nodeMediaListImages
118 +     }
119 +   }
120 + }
121 + `;
122 +
123 + module.exports = {
124 +   fragment: nodeMediaListImages,
125 +   GetNodeMediaListImages,
126 + };

```

39 src/site/stages/build/drupal/graphql/nodeMediaListVideos.graphql.js



```

...   ...   @@ -1,6 +1,7 @@
1     + const fragments = require('./fragments.graphql');
2     + const entityElementsFromPages = require('./entityElementsForPages.graphql');
3     - const fragment = `
4     + const nodeMediaListVideos = `
5     + fragment nodeMediaListVideos on NodeMediaListVideos {
6     +   ${entityElementsFromPages}
7     +   entityBundle
83    84   }
84    85   `;
85    86
86    - module.exports = fragment;
87    + const GetNodeMediaListVideos = `
88    +   ${fragments.alertParagraph}
89    +   ${fragments.alertParagraphSingle}
90    +   ${fragments.button}
91    +   ${fragments.contactInformation}
92    +   ${fragments.supportService}
93    +   ${fragments.linkTeaser}
94    +   ${fragments.termLcCategory}
95    +   ${fragments.audienceTopics}
96    +   ${fragments.emailContact}
97    +   ${fragments.phoneNumber}
98    +   ${fragments.audienceBeneficiaries}
99    +   ${fragments.audienceNonBeneficiaries}
100   +   ${fragments.termTopics}
101   +
102   +   ${nodeMediaListVideos}
103   +
104   +   query GetNodeMediaListVideos($onlyPublishedContent: Boolean!) {
105   +     nodeQuery(limit: 1000, filter: {
106   +       conditions: [

```



```

107 +         { field: "status", value: ["1"], enabled: $onlyPublishedContent }
108 +         { field: "type", value: ["media_list_videos"] }
109 +     ]
110 + }) {
111 +     entities {
112 +         ... nodeMediaListVideos
113 +     }
114 + }
115 + }
116 + `;
117 +
118 + module.exports = {
119 +     fragment: nodeMediaListVideos,
120 +     GetNodeMediaListVideos,
121 + };

```

41 src/site/stages/build/drupal/graphql/nodeQa.graphql.js

@@ -1,6 +1,7 @@

```

1 + const fragments = require('./fragments.graphql');
2   const entityElementsFromPages = require('./entityElementsForPages.graphql');
3 - const fragment = `
4 + const nodeQa = `
5   fragment nodeQa on NodeQA {
6     ${entityElementsFromPages}
7     entityBundle
70  }
71  `;
72
73 - module.exports = fragment;
74 + const GetNodeQa = `
75 +   ${fragments.richTextCharLimit1000}
76 +   ${fragments.reactWidget}
77 +   ${fragments.alertParagraph}
78 +   ${fragments.alertParagraphSingle}
79 +   ${fragments.button}
80 +   ${fragments.contactInformation}
81 +   ${fragments.supportService}
82 +   ${fragments.linkTeaser}
83 +   ${fragments.termLcCategory}
84 +   ${fragments.audienceTopics}
85 +   ${fragments.emailContact}
86 +   ${fragments.phoneNumber}
87 +   ${fragments.audienceBeneficiaries}
88 +   ${fragments.audienceNonBeneficiaries}
89 +   ${fragments.termTopics}
90 +
91 +   ${nodeQa}
92 +
93 +   query GetNodeQa($onlyPublishedContent: Boolean!) {
94 +     nodeQuery(limit: 1000, filter: {
95 +       conditions: [
96 +         { field: "status", value: ["1"], enabled: $onlyPublishedContent }

```

```

96 +         { field: "status", value: ["1"], enabled: $onlyPublishedContent }
97 +         { field: "type", value: ["q_a"] }
98 +     ]
99 + }) {
100 +     entities {
101 +         ... nodeQa
102 +     }
103 + }
104 + }
105 + `;
106 +
107 + module.exports = {
108 +     fragment: nodeQa,
109 +     GetNodeQa,
110 + };

```

39 src/site/stages/build/drupal/graphql/nodeStepByStep.graphql.js

@@ -1,6 +1,7 @@

```

1 + const fragments = require('./fragments.graphql');
2   const entityElementsFromPages = require('./entityElementsForPages.graphql');
3 - const fragment = `
4 + const nodeStepByStep = `
5   fragment nodeStepByStep on NodeStepByStep {
6     ${entityElementsFromPages}
7     entityBundle
97 98   }
98 99   `;
99 100
100 - module.exports = fragment;
101 + const GetNodeStepByStep = `
102 +   ${fragments.alertParagraph}
103 +   ${fragments.alertParagraphSingle}
104 +   ${fragments.button}
105 +   ${fragments.contactInformation}
106 +   ${fragments.supportService}
107 +   ${fragments.linkTeaser}
108 +   ${fragments.termLcCategory}
109 +   ${fragments.audienceTopics}
110 +   ${fragments.emailContact}
111 +   ${fragments.phoneNumber}
112 +   ${fragments.audienceBeneficiaries}
113 +   ${fragments.audienceNonBeneficiaries}
114 +   ${fragments.termTopics}
115 +
116 +   ${nodeStepByStep}
117 +
118 +   query GetNodeStepByStep($onlyPublishedContent: Boolean!) {
119 +     nodeQuery(limit: 1000, filter: {
120 +       conditions: [
121 +         { field: "status", value: ["1"], enabled: $onlyPublishedContent }
122 +         { field: "type", value: ["step_by_step"] }
123 +       ]
124 +     }) {

```

```

124 +     }) {
125 +         entities {
126 +             ... nodeStepByStep
127 +         }
128 +     }
129 + }
130 + `;
131 +
132 + module.exports = {
133 +     fragment: nodeStepByStep,
134 +     GetNodeStepByStep,
135 + };

```

▼ 51

src/site/stages/build/drupal/graphql/nodeSupportResourcesDetailPage.graphql.js

...	...	@@ -1,6 +1,7 @@
	1	+ const fragments = require('./fragments.graphql');
1	2	const entityElementsFromPages = require('./entityElementsForPages.graphql
2	3	
3		- const fragment = `
	4	+ const nodeSupportResourcesDetailPage = `
4	5	fragment nodeSupportResourcesDetailPage on NodeSupportResourcesDetailPage
5	6	{entityElementsFromPages}
6	7	entityBundle
86	87	}
87	88	`;
88	89	
89		- module.exports = fragment;
	90	+ const GetNodeSupportResourcesDetailPage = `
	91	+
	92	+ \${fragments.linkTeaser}
	93	+ \${fragments.alertParagraphSingle}
	94	+ \${fragments.button}
	95	+ \${fragments.contactInformation}
	96	+ \${fragments.supportService}
	97	+ \${fragments.termTopics}
	98	+ \${fragments.termLcCategory}
	99	+ \${fragments.audienceTopics}
	100	+ \${fragments.emailContact}
	101	+ \${fragments.phoneNumber}
	102	+ \${fragments.audienceBeneficiaries}
	103	+ \${fragments.audienceNonBeneficiaries}
	104	+ \${fragments.wysiwyg}
	105	+ \${fragments.collapsiblePanel}
	106	+ \${fragments.process}
	107	+ \${fragments.qaSection}
	108	+ \${fragments.qa}
	109	+ \${fragments.listOfLinkTeasers}
	110	+ \${fragments.reactWidget}
	111	+ \${fragments.spanishSummary}
	112	+ \${fragments.alertParagraph}
	113	+ \${fragments.table}

```

114 +   ${fragments.downloadableFile}
115 +   ${fragments.embeddedImage}
116 +   ${fragments.numberCallout}
117 +
118 +   ${nodeSupportResourcesDetailPage}
119 +
120 +   query GetNodeSupportResourcesDetailPage($onlyPublishedContent: Boolean!
121 +     nodeQuery(limit: 100, filter: {
122 +       conditions: [
123 +         { field: "status", value: ["1"], enabled: $onlyPublishedContent }
124 +         { field: "type", value: ["support_resources_detail_page"] }
125 +       ]
126 +     }) {
127 +       entities {
128 +         ... nodeSupportResourcesDetailPage
129 +       }
130 +     }
131 +   }
132 + `;
133 + module.exports = {
134 +   fragment: nodeSupportResourcesDetailPage,
135 +   GetNodeSupportResourcesDetailPage,
136 + };

```

25 src/site/stages/build/drupal/graphql/officePage.graphql.js

```

4 4  */
5 5  const entityElementsFromPages = require('./entityElementsForPages.graphql
6 6
7 7  - module.exports = `
7 7  + const officeFragment = `
8 8  fragment officePage on NodeOffice {
9 9    ${entityElementsFromPages}
10 10  changed
39 39  }
40 40  }
41 41  `;
42 +
43 + const GetNodeOffices = `
44 +
45 +   ${officeFragment}
46 +
47 +   query GetNodeOffices($onlyPublishedContent: Boolean!) {
48 +     nodeQuery(limit: 1000, filter: {
49 +       conditions: [
50 +         { field: "status", value: ["1"], enabled: $onlyPublishedContent }
51 +         { field: "type", value: ["office"] }
52 +       ]
53 +     }) {
54 +       entities {
55 +         ... officePage
56 +       }
57 +     }
58 +   }

```

```

59 + `;
60 +
61 + module.exports = {
62 +   fragment: officeFragment,
63 +   GetNodeOffices,
64 + };

```

107 src/site/stages/build/drupal/graphql/page.graphql.js

@@ -1,30 +1,13 @@

```

1 + const fragments = require('./fragments.graphql');
2   const entityElementsFromPages = require('./entityElementsForPages.graphql');
3   const { FIELD_ALERT } = require('./block-fragments/alert.block.graphql');
4   const {
5     FIELD_RELATED_LINKS,
6   } = require('./paragraph-fragments/listOfLinkTeasers.paragraph.graphql');
7
8   /**
9    * A standard content page, that is ordinarily two-levels deep (a child page)
10   * For example, /health-care/apply.
11   */

```

```

11 - const WYSIWYG = '... wysiwyg';
12 - const COLLAPSIBLE_PANEL = '... collapsiblePanel';
13 - const PROCESS = '... process';
14 - const QA_SECTION = '... qaSection';
15 - const QA = '... qa';
16 - const LIST_OF_LINK_TEASERS = '... listOfLinkTeasers';
17 - const REACT_WIDGET = '... reactWidget';
18 - const SPANISH_SUMMARY = '... spanishSummary';
19 - const ALERT_PARAGRAPH = '... alertParagraph';
20 - const TABLE = '... table';
21 - const DOWNLOADABLE_FILE_PARAGRAPH = '... downloadableFile';
22 - const MEDIA_PARAGRAPH = '... embeddedImage';
23 - const NUMBER_CALLOUT = '... numberCallout';

```

```

8 + const { generatePaginatedQueries } = require('../individual-queries-helper');

```

```

25 - const fieldAdministrationKey = 'FieldNodePageFieldAdministration';
26 -
27 - module.exports = `

```

```

10 + const pageFragment = `

```

```

11   fragment page on NodePage {
12     ${entityElementsFromPages}
13
14     entity {
15       entityType
16       entityBundle
17
18       ${WYSIWYG}
19       ${QA}
20
21       ... wysiwyg
22       ... qa

```

```

23   }
24   }
25   fieldContentBlock {
26     entity {

```

45	28		entity {
46	29		entityType
47	30		entityBundle
48		-	\${WYSIWYG}
49		-	\${COLLAPSIBLE_PANEL}
50		-	\${PROCESS}
51		-	\${QA_SECTION}
52		-	\${LIST_OF_LINK_TEASERS}
53		-	\${REACT_WIDGET}
54		-	\${SPANISH_SUMMARY}
55		-	\${TABLE}
56		-	\${ALERT_PARAGRAPH}
57		-	\${DOWNLOADABLE_FILE_PARAGRAPH}
58		-	\${MEDIA_PARAGRAPH}
59		-	\${NUMBER_CALLOUT}
	31	+	... wysiwyg
	32	+	... collapsiblePanel
	33	+	... process
	34	+	... qaSection
	35	+	... qa
	36	+	... listOfLinkTeasers
	37	+	... reactWidget
	38	+	... spanishSummary
	39	+	... alertParagraph
	40	+	... table
	41	+	... downloadableFile
	42	+	... embeddedImage
	43	+	... numberCallout
60	44		}
61	45		}
62	46		\${FIELD_ALERT}
63	47		\${FIELD_RELATED_LINKS}
64	48		fieldAdministration {
65		-	... on \${fieldAministrationKey} {
	49	+	... on FieldNodePageFieldAdministration {
66	50		entity {
67	51		... on TaxonomyTermAdministration {
68	52		name
76	60		changed
77	61		}
78	62		};
	63	+	+
	64	+	function getPageNodeSlice(operationName, offset, limit) {
	65	+	return `
	66	+	\${fragments.linkTeaser}
	67	+	\${fragments.alert}
	68	+	\${fragments.wysiwyg}
	69	+	\${fragments.collapsiblePanel}
	70	+	\${fragments.process}
	71	+	\${fragments.qaSection}
	72	+	\${fragments.qa}
	73	+	\${fragments.listOfLinkTeasers}
	74	+	\${fragments.reactWidget}
	75	+	\${fragments.spanishSummary}
	76	+	\${fragments.numberCallout}

```

76 +     ${fragments.alertParagraph}
77 +     ${fragments.table}
78 +     ${fragments.downloadableFile}
79 +     ${fragments.embeddedImage}
80 +     ${fragments.numberCallout}
81 +
82 +     ${pageFragment}
83 +
84 +     query ${operationName}($onlyPublishedContent: Boolean!) {
85 +       nodeQuery(
86 +         limit: ${limit}
87 +         offset: ${offset}
88 +         sort: { field: "changed", direction: ASC }
89 +         filter: {
90 +           conditions: [
91 +             { field: "status", value: ["1"], enabled: $onlyPublishedContent }
92 +             { field: "type", value: ["page"] }
93 +           ]
94 +         }) {
95 +           entities {
96 +             ... page
97 +           }
98 +         }
99 +       }
100 + `;
101 + }
102 +
103 + function getNodePageQueries(entityCounts) {
104 +   return generatePaginatedQueries({
105 +     operationNamePrefix: 'GetNodePage',
106 +     entitiesPerSlice: 25,
107 +     totalEntities: entityCounts.data.benefitPages.count,
108 +     getSlice: getPageNodeSlice,
109 +   });
110 + }
111 +
112 + module.exports = {
113 +   fragment: pageFragment,
114 +   getNodePageQueries,
115 + };

```

6 src/site/stages/build/drupal/graphql/paragraph-

fragments/staffProfile.paragraph.graphql.js

16	16	fieldDescription
17	17	fieldEmailAddress
18	18	fieldPhoneNumber
19	-	
19	+	
20	20	fieldBody {
21	21	processed
22	22	}
34	34	image {
35	35	alt

--	--	---
36	36	title
37	-	derivative(style: <u>1_1_SQUARE_MEDIUM_THUMBNAIL</u> ) {
37	+	derivative(style: <u>11SQUAREMEDIUMTHUMBNAIL</u> ) {



**mpelzsherman** on Feb 9 Member

Curious why this change was needed?



**nckslvn** on Feb 9 Author Member

Using `_1_1_SQUARE_MEDIUM_THUMBNAIL` was reported as a GraphQL error and was relying on `src/site/utilities/stringHelpers.js` to turn it into `_11SQUAREMEDIUMTHUMBNAIL`. Instead of using the `stringHelper`, I just changed the value, because I can't find any context for why it was like that. Asking the CMS folks if they have any idea, <https://dsva.slack.com/archives/CDHBKAL9W/p1612907370100500>



**nckslvn** on Feb 9 Author Member

Found it...it's safe to just make this change. The string-helpers that did the replacement before is just leftover complexity from a CMS feature flag, [7b54b49 #diff-78d2ba7081dd26ac2a8a81b3bc6c0c80461f62412580c370dff8564362e33ee4R71](#).



**nckslvn** on Feb 11 Author Member

I wrote a ticket to remove this old code, [department-of-veterans-affairs/va.gov-team#19722](#)



Reply...

Resolve conversation

38	38	url
39	39	width
40	40	height
49	49	}
50	50	}
51	51	`;
	52	+
	53	+ // Had to change "_11SQUAREMEDIUMTHUMBNAIL" to "_11SQUAREMEDIUMTHUMBNAIL"

✓ 25 src/site/stages/build/drupal/graphql/pressReleasePage.graphql.js

4	4	*/
5	5	const entityElementsFromPages = require('./entityElementsForPages.graphql
6	6	
7	-	module.exports = `
7	+	const pressReleasePage = `
8	8	fragment pressReleasePage on NodePressRelease {
9	9	\${entityElementsFromPages}
10	10	fieldReleaseDate {
74	74	}
75	75	}
76	76	`;
77	+	



```

78 + const GetNodePressReleasePages = `
79 +
80 +   ${pressReleasePage}
81 +
82 +   query GetNodeVaForms($onlyPublishedContent: Boolean!) {
83 +     nodeQuery(limit: 1000, filter: {
84 +       conditions: [
85 +         { field: "status", value: ["1"], enabled: $onlyPublishedContent }
86 +         { field: "type", value: ["press_release"] }
87 +       ]
88 +     }) {
89 +       entities {
90 +         ... pressReleasePage
91 +       }
92 +     }
93 +   }
94 + `;
95 +
96 + module.exports = {
97 +   fragment: pressReleasePage,
98 +   GetNodePressReleasePages,
99 + };

```

✓ 25

src/site/stages/build/drupal/graphql/pressReleasesListingPage.graphql.js

4	4	*/
5	5	const entityElementsFromPages = require('./entityElementsForPages.graphql
6	6	
7		- module.exports = `
	7	+ const pressReleasesListingPage = `
8	8	fragment pressReleasesListingPage on NodePressReleasesListing {
9	9	\${entityElementsFromPages}
10	10	fieldIntroText
45	45	}
46	46	}
47	47	`;
	48	+
	49	+ const GetNodePressReleaseListingPages = `
	50	+
	51	+   \${pressReleasesListingPage}
	52	+
	53	+   query GetNodePressReleaseListingPages(\$onlyPublishedContent: Boolean!) {
	54	+     nodeQuery(limit: 500, filter: {
	55	+       conditions: [
	56	+         { field: "status", value: ["1"], enabled: \$onlyPublishedContent }
	57	+         { field: "type", value: ["press_releases_listing"] }
	58	+       ]
	59	+     }) {
	60	+       entities {
	61	+         ... pressReleasesListingPage
	62	+       }
	63	+     }
	64	+   }

```

64 +   }
65 + `;
66 +
67 + module.exports = {
68 +   fragment: pressReleasesListingPage,
69 +   GetNodePressReleaseListingPages,
70 + };

```

✓ ⓘ 24 src/site/stages/build/drupal/graphql/storyListingPage.graphql.js

```

4 4  */
5 5  const entityElementsFromPages = require('./entityElementsForPages.graphql
6 6
7 7  - module.exports = `
7 7  + const storyListingPage = `
8 8    fragment storyListingPage on NodeStoryListing {
9 9      ${entityElementsFromPages}
10 10     fieldIntroText
71 71   }
72 72   }
73 73 `;
74 +
75 + const GetNodeStoryListingPages = `
76 +   ${storyListingPage}
77 +
78 +   query GetNodeStoryListingPages($onlyPublishedContent: Boolean!) {
79 +     nodeQuery(limit: 1000, filter: {
80 +       conditions: [
81 +         { field: "status", value: ["1"], enabled: $onlyPublishedContent }
82 +         { field: "type", value: ["story_listing"] }
83 +       ]
84 +     }) {
85 +       entities {
86 +         ... storyListingPage
87 +       }
88 +     }
89 +   }
90 + `;
91 +
92 + module.exports = {
93 +   fragment: storyListingPage,
94 +   GetNodeStoryListingPages,
95 + };

```

✓ ⓘ 13 src/site/stages/build/drupal/graphql/taxonomy-

fragments/GetTaxonomies.graphql.js

```

...  ...  @@ -1,4 +1,4 @@
1 1  - module.exports = `
1 1  + const partialQuery = `
2 2    allTaxonomies: taxonomyTermQuery(limit: 1000) {
3 3      entities {
4 4        entityBundle
...  ...

```

```

27     }
28   }
29   `;
30 +
31 + const GetTaxonomies = `
32 +   query {
33 +     ${partialQuery}
34 +   }
35 + `;
36 +
37 + module.exports = {
38 +   partialQuery,
39 +   GetTaxonomies,
40 + };

```

✓ 44 src/site/stages/build/drupal/graphql/vaFormPage.graphql.js

```

...   ...   @@ -1,7 +1,10 @@
1     1     const entityElementsFromPages = require('./entityElementsForPages.graphql
2     2     const { FIELD_ALERT } = require('./block-fragments/alert.block.graphql');
3     3     + const fragments = require('./fragments.graphql');
4     4
5     5     + const { generatePaginatedQueries } = require('./individual-queries-helpe
6     6     +
7     7     + const vaFormFragment = `
8     8     fragment vaFormPage on NodeVaForm {
9     9         ${entityElementsFromPages}
10    10        changed
11    11    }
12    12    `;
13    13
14    14    - module.exports = fragment;
15    15
16    16    + function getNodeVaFormSlice(operationName, offset, limit) {
17    17    +     return `
18    18    +       ${fragments.alert}
19    19    +       ${fragments.linkTeaser}
20    20    +       ${vaFormFragment}
21    21    +
22    22    +       query ${operationName}($onlyPublishedContent: Boolean!) {
23    23    +         nodeQuery(
24    24    +           limit: ${limit}
25    25    +           offset: ${offset}
26    26    +           sort: { field: "changed", direction: ASC }
27    27    +           filter: {
28    28    +             conditions: [
29    29    +               { field: "status", value: ["1"], enabled: $onlyPublishedContent
30    30    +               { field: "type", value: ["va_form"] }
31    31    +             ]
32    32    +           }) {
33    33    +             entities {
34    34    +               ... vaFormPage
35    35    +             }
36    36    +           }
37    37    +         }
38    38    +       }

```

```

104 +   }
105 + `;
106 + }
107 +
108 + function getNodeVaFormQueries(entityCounts) {
109 +   return generatePaginatedQueries({
110 +     operationNamePrefix: 'GetNodeVaForm',
111 +     entitiesPerSlice: 25,
112 +     totalEntities: entityCounts.data.vaForm.count,
113 +     getSlice: getNodeVaFormSlice,
114 +   });
115 + }
116 +
117 + module.exports = {
118 +   fragment: vaFormFragment,
119 +   getNodeVaFormQueries,
120 + };

```

▼ ⓘ 27

src/site/stages/build/drupal/graphql/vamcOperatingStatusAndAlerts.graphql.js

...	...	@@ -1,6 +1,6 @@
1	1	const entityElementsFromPages = require('./entityElementsForPages.graphql
2	2	
3		- module.exports = `
	3	+ const vamcOperatingStatusAndAlerts = `
4	4	fragment vamcOperatingStatusAndAlerts on NodeVamcOperatingStatusAndAlerts {
5	5	\${entityElementsFromPages}
6	6	title
54	54	endValue
55	55	endTime
56	56	timezone
57		- }
	57	+ }
58	58	fieldWysiwyg {
59	59	processed
60	60	}
69	69	}
70	70	}
71	71	`;
	72	+
	73	+ const GetNodeVamcOperatingStatusAndAlerts = `
	74	+
	75	+   \${vamcOperatingStatusAndAlerts}
	76	+
	77	+   query GetNodeVamcOperatingStatusAndAlerts(\$onlyPublishedContent: Boolean!
	78	+     nodeQuery(limit: 500, filter: {
	79	+       conditions: [
	80	+         { field: "status", value: ["1"], enabled: \$onlyPublishedContent }
	81	+         { field: "type", value: ["vamc_operating_status_and_alerts"] }
	82	+       ]
	83	+     }) {
	84	+       entities {

```

85 +         ... vamcOperatingStatusAndAlerts
86 +     }
87 + }
88 + }
89 + `;
90 +
91 + module.exports = {
92 +   fragment: vamcOperatingStatusAndAlerts,
93 +   GetNodeVamcOperatingStatusAndAlerts,
94 + };

```

33 src/site/stages/build/drupal/individual-queries-helpers.js

@@ -0,0 +1,33 @@

```

1 + function generatePaginatedQueries({
2 +   operationNamePrefix,
3 +   entitiesPerSlice,
4 +   totalEntities,
5 +   getSlice,
6 + }) {
7 +   const numberOfSlices = Math.ceil(totalEntities / entitiesPerSlice);
8 +
9 +   return new Array(numberOfSlices)
10 +     .fill(null)
11 +     .map((_, index) => {
12 +       const operationName = `${operationNamePrefix}__slice${index + 1}`;
13 +       const offset = entitiesPerSlice * index;
14 +       const isLastPage = index + 1 === numberOfSlices;
15 +
16 +       let limit = entitiesPerSlice;
17 +       if (isLastPage) {
18 +         limit = totalEntities - offset;
19 +       }
20 +
21 +       return [operationName, getSlice(operationName, offset, limit)];
22 +     })
23 +     .reduce((queriesByOperationName, [operationName, query]) => {
24 +       return {
25 +         ...queriesByOperationName,
26 +         [operationName]: query,
27 +       };
28 +     }, {});
29 + }
30 +
31 + module.exports = {
32 +   generatePaginatedQueries,
33 + };

```

184 src/site/stages/build/drupal/individual-queries.js

@@ -0,0 +1,184 @@

```

1 + const { CountEntityTypes } = require('./graphql/CountEntityTypes.graphql')
2 +

```

```
3 + const { getNodePageQueries } = require('./graphql/page.graphql');
4 + const { GetNodeLandingPages } = require('./graphql/landingPage.graphql');
5 + const { getNodeVaFormQueries } = require('./graphql/vaFormPage.graphql');
6 +
7 + const { getNodePersonProfileQueries } = require('./graphql/bioPage.graphql');
8 +
9 + const {
10 +   getNodeHealthCareRegionPageQueries,
11 + } = require('./graphql/healthCareRegionPage.graphql');
12 +
13 + const { GetNodeOffices } = require('./graphql/officePage.graphql');
14 +
15 + const {
16 +   getNodeHealthCareLocalFacilityPageQueries,
17 + } = require('./graphql/healthCareLocalFacilityPage.graphql');
18 +
19 + const {
20 +   getNodeHealthServicesListingPageQueries,
21 + } = require('./graphql/healthServicesListingPage.graphql');
22 +
23 + const { GetNodeNewsStoryPages } = require('./graphql/newStoryPage.graphql');
24 +
25 + const {
26 +   GetNodePressReleasePages,
27 + } = require('./graphql/pressReleasePage.graphql');
28 +
29 + const {
30 +   GetNodePressReleaseListingPages,
31 + } = require('./graphql/pressReleasesListingPage.graphql');
32 +
33 + const {
34 +   GetNodeEventListingPage,
35 + } = require('./graphql/eventListingPage.graphql');
36 +
37 + const { getNodeEventQueries } = require('./graphql/eventPage.graphql');
38 +
39 + const {
40 +   GetNodeStoryListingPages,
41 + } = require('./graphql/storyListingPage.graphql');
42 +
43 + const {
44 +   GetNodeLocationsListingPages,
45 + } = require('./graphql/locationsListingPage.graphql');
46 +
47 + const {
48 +   GetNodeLeadershipListingPages,
49 + } = require('./graphql/leadershipListingPage.graphql');
50 +
51 + const {
52 +   GetNodeVamcOperatingStatusAndAlerts,
53 + } = require('./graphql/vamcOperatingStatusAndAlerts.graphql');
54 +
55 + const {
56 +   GetNodePublicationListingPages,
```

```
57 + } = require('./graphql/benefitListingPage.graphql');
58 +
59 + const {
60 +   getNodeHealthCareRegionDetailPageQueries,
61 + } = require('./graphql/healthCareRegionDetailPage.graphql');
62 +
63 + const { GetNodeQa } = require('./graphql/nodeQa.graphql');
64 + const { GetNodeMultipleQaPages } = require('./graphql/faqMultipleQa.graphql');
65 + const { GetNodeStepByStep } = require('./graphql/nodeStepByStep.graphql');
66 + const {
67 +   GetNodeMediaListImages,
68 + } = require('./graphql/nodeMediaListImages.graphql');
69 + const { GetNodeChecklist } = require('./graphql/nodeChecklist.graphql');
70 + const {
71 +   GetNodeMediaListVideos,
72 + } = require('./graphql/nodeMediaListVideos.graphql');
73 +
74 + const {
75 +   GetNodeSupportResourcesDetailPage,
76 + } = require('./graphql/nodeSupportResourcesDetailPage.graphql');
77 +
78 + const {
79 +   GetNodeBasicLandingPage,
80 + } = require('./graphql/nodeBasicLandingPage.graphql');
81 +
82 + const {
83 +   GetCampaignLandingPages,
84 + } = require('./graphql/nodeCampaignLandingPage.graphql');
85 +
86 + function getNodeQueries(entityCounts) {
87 +   return {
88 +     ...getNodePageQueries(entityCounts),
89 +     GetNodeLandingPages,
90 +     ...getNodeVaFormQueries(entityCounts),
91 +     ...getNodeHealthCareRegionPageQueries(entityCounts),
92 +     ...getNodePersonProfileQueries(entityCounts),
93 +     GetNodeOffices,
94 +     ...getNodeHealthCareLocalFacilityPageQueries(entityCounts),
95 +     ...getNodeHealthServicesListingPageQueries(entityCounts),
96 +     GetNodeNewsStoryPages,
97 +     GetNodePressReleasePages,
98 +     GetNodePressReleaseListingPages,
99 +     GetNodeEventListingPage,
100 +     ...getNodeEventQueries(entityCounts),
101 +     GetNodeStoryListingPages,
102 +     GetNodeLocationsListingPages,
103 +     GetNodeLeadershipListingPages,
104 +     GetNodeVamcOperatingStatusAndAlerts,
105 +     GetNodePublicationListingPages,
106 +     ...getNodeHealthCareRegionDetailPageQueries(entityCounts),
107 +     GetNodeQa,
108 +     GetNodeMultipleQaPages,
109 +     GetNodeStepByStep,
110 +     GetNodeMediaListImages,
```

```

111 +     GetNodeChecklist,
112 +     GetNodeMediaListVideos,
113 +     GetNodeSupportResourcesDetailPage,
114 +     GetNodeBasicLandingPage,
115 +     GetCampaignLandingPages,
116 + };
117 + }
118 +
119 + function nonNodeQueries() {
120 +     // Get current feature flags
121 +     const { cmsFeatureFlags } = global;
122 +
123 +     const { GetIcsFiles } = require('./graphql/file-fragments/ics.file.graphql');
124 +
125 +     const {
126 +         GetSidebars,
127 +     } = require('./graphql/navigation-fragments/sidebar.nav.graphql');
128 +
129 +     const {
130 +         VaFacilitySidebars,
131 +     } = require('./graphql/navigation-fragments/facilitySidebar.nav.graphql');
132 +
133 +     const {
134 +         GetOutreachSidebar,
135 +     } = require('./graphql/navigation-fragments/outreachSidebar.nav.graphql');
136 +
137 +     const { GetAlerts } = require('./graphql/alerts.graphql');
138 +     const { GetBannerAlerts } = require('./graphql/bannerAlerts.graphql');
139 +     const {
140 +         GetOutreachAssets,
141 +     } = require('./graphql/file-fragments/outreachAssets.graphql');
142 +     const { GetHomepage } = require('./graphql/homePage.graphql');
143 +     const {
144 +         GetMenuLinks,
145 +     } = require('./graphql/navigation-fragments/menuLinks.nav.graphql');
146 +     const {
147 +         GetTaxonomies,
148 +     } = require('./graphql/taxonomy-fragments/GetTaxonomies.graphql');
149 +
150 +     const componentQueries = {
151 +         GetIcsFiles,
152 +         GetSidebars,
153 +         ...VaFacilitySidebars,
154 +         GetOutreachSidebar,
155 +         GetAlerts,
156 +         GetBannerAlerts,
157 +         GetOutreachAssets,
158 +         GetHomepage,
159 +         GetMenuLinks,
160 +         GetTaxonomies,
161 +     };
162 +
163 +     if (cmsFeatureFlags.FEATURE_ALL_HUB_SIDE_NAVS) {

```



```

164 +     const {
165 +         GetAllSideNavMachineNames,
166 +     } = require('./graphql/navigation-fragments/allSideNavMachineNames.nav
167 +
168 +     componentQueries.GetAllSideNavMachineNames = GetAllSideNavMachineName
169 + }
170 +
171 + return componentQueries;
172 + }
173 +
174 + function getIndividualizedQueries(entityCounts) {
175 +     return {
176 +         ...getNodeQueries(entityCounts),
177 +         ...nonNodeQueries(),
178 +     };
179 + }
180 +
181 + module.exports = {
182 +     getIndividualizedQueries,
183 +     CountEntityTypes,
184 + };

```

✓ 17 src/site/stages/build/drupal/menus.js

231	231	}
232	232	
233	233	/**
234		- * Make a 'section' in the first tab of the megaMenu.
235		- *
236		- * The first tab of the megaMenu is broken down by 'section', each of
237		- * which corresponds to a benefit hub.
	234	+ * Make a 'section' in the first tab of the megaMenu.
	235	+ *
	236	+ * The first tab of the megaMenu is broken down by 'section', each of
	237	+ * which corresponds to a benefit hub.
238	238	
239		- * The title of the section (e.g., 'Health care') lives in a list
	239	+ * The title of the section (e.g., 'Health care') lives in a list
240	240	* in the left side of the menu block. The hub's links live in
241	241	* columns to the right.
242		- *
	242	+ *
243	243	* @param {string} hostUrl - Absolute url for the site.
244	244	* @param {Object} hub - Collection of title and links for this section.
245	245	* @param {number} arrayDepth - Total depth of this tab.
265	265	* @return {Array} headerData - Menu information formatted for the megaMenu
266	266	*/
267	267	function formatHeaderData(buildOptions, contentData) {
268		- if (!contentData?.data) {
	268	+ if (!contentData?.data?.menuLinkContentQuery?.entities) {
269	269	// eslint-disable-next-line no-console
270		- console.warn('formatHeaderData has no data');
271		- return null;

270 | + throw new Error('formatHeaderData has no data');



nckslvn on Feb 11

Author

Member

I made this check stronger because I just can't imagine a case where it is acceptable for a b success when the MegaMenu is actually failing to build.



Reply...

Resolve conversation

```
272 | 271 | }
273 | 272 |
274 | 273 | let menuLinks = contentData.data.menuLinkContentQuery.entities;
```

✓ ⓘ 2 ■■■■ src/site/stages/build/drupal/metalsmith-drupal.js

```
159 | 159 |
160 | 160 | console.time(drupalTimer);
161 | 161 |
162 | 162 | - drupalPages = await contentApi.getAllPages();
    | 162 | + drupalPages = await contentApi.getAllPagesViaIndividualGraphQLQueries
163 | 163 |
164 | 164 | console.timeEnd(drupalTimer);
165 | 165 |
```

💡 **ProTip!** Use `n` and `p` to navigate between commits in a pull request.