Report - Use the 'Custom Javascript' tab (example: make clickable links from text)

Scripting Solutions

Additional scripting solutions will be added in the future. Please reach out to Alchemer with comments and suggestions on solutions you'd like to see via the link here.

Scripting and Other Custom Solutions

We're always happy to help you debug any documented script that is used as is. That said, we do not have the resources to write scripts on demand or to debug a customized script.

If you have customization ideas that you haven't figured out how to tackle, we're happy to be a sounding board for Alchemer features and functionality ideas that might meet your needs. Beyond this, check out our Professional Services; these folks have the scripting chops to help you to achieve what you are looking for!

Goal

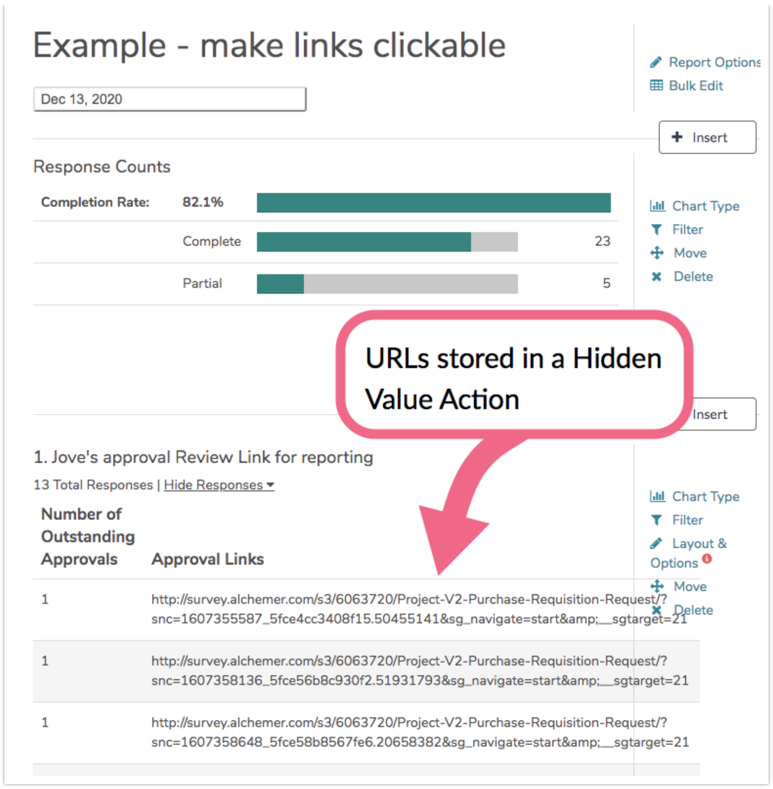
Provide an example to Javascript developers by using **Reports** > **Report Options** > **CUSTOM JAVASCRIPT.** The example provided converts the URL link text to clickable links.

Solution

The solution waits for report elements to load and uses a callback to modify the charts in the DOM.

In this specific example, the code converts the reported values of a Hidden Value Action (where the responses contain the text of a URL link) into clickable links as soon as the chart loads. The code will look for a link anywhere in the text so you can have other text before or after the link.

From this:



Example - make links clickable

Dec 13, 2020

Response Counts

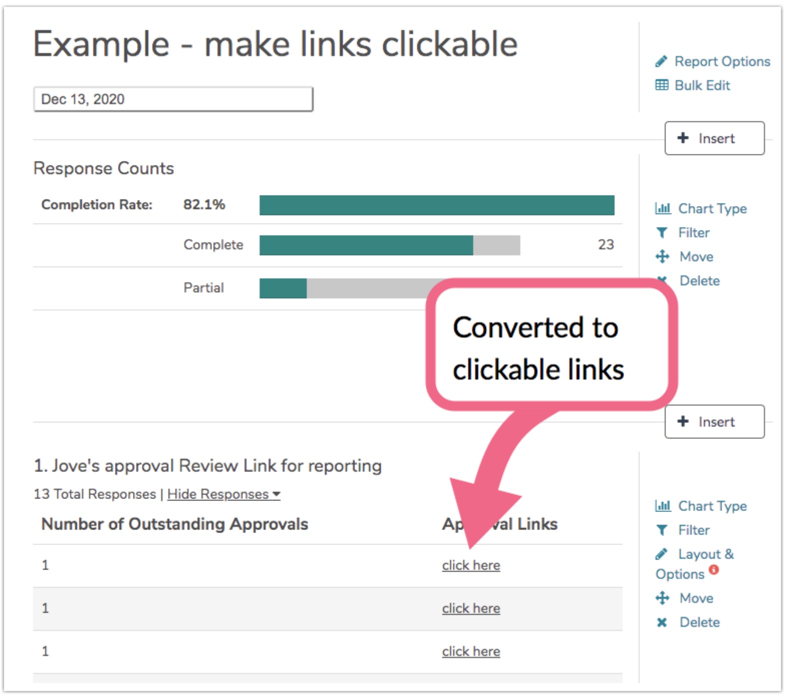
Completion Rate:	**82.1%**	
	Complete	23
	Partial	5

1. Jove's approval Review Link for reporting

13 Total Responses | Hide Responses ▾

Number of Outstanding Approvals	Approval Links
1	http://survey.alchemer.com/s3/6063720/Project-V2-Purchase-Requisition-Request/?snc=1607355587_5fce4cc3408f15.50455141&sg_navigate=start&amp;__sgtarget=21
1	http://survey.alchemer.com/s3/6063720/Project-V2-Purchase-Requisition-Request/?snc=1607358136_5fce56b8c930f2.51931793&sg_navigate=start&amp;__sgtarget=21
1	http://survey.alchemer.com/s3/6063720/Project-V2-Purchase-Requisition-Request/?snc=1607358648_5fce58b8567fe6.20658382&sg_navigate=start&amp;__sgtarget=21

To this:



Example - make links clickable

Dec 13, 2020

Response Counts

Completion Rate:	**82.1%**	
	Complete	23
	Partial	

1. Jove's approval Review Link for reporting

13 Total Responses | Hide Responses ▾

Number of Outstanding Approvals	Approval Links
1	click here
1	click here
1	click here

Javascript code to add to **Report > Report Options > CUSTOM JAVASCRIPT:**

```
/* Alchemer ver 01


   Example using CUSTOM JAVASCRIPT in a report.
   For this example question ID 359 is a Hidden Value Action populated with the text
   for a single URL.  The report shows this Hidden Value Action with the settings:


     -- Summary Table, CHECKED
     -- Collapse Open Text Responses, UNCHECKED


   The result is an on screen list of the Hidden Value URLs for each response.


   The example script below will convert these to clickable  links.


   Reports load asynchronously so we use a MutationObserver to respond as the
   report charts are loaded.
*/
document.addEventListener("DOMContentLoaded", function() {


  const VERSION = '05'


/*
  // Question IDs for report elements we want to change after they are loaded
  //   (Array of string)
  const UPDATE_QIDS = [
    "359",
  ]
*/


  // * * * * * * * * * * * * * *
  // * no changes needed below *
  // * * * * * * * * * * * * * *


  const LOG = true


  /***
   * Tests if url is a valid url structure
   *
   * url (string) url starting with http/s, ex: https://www.google.com
   * return (t/f) true if string is a valid URL
   */
  const isValidUrl = (url) => {
    // see https://stackoverflow.com/questions/3809401/what-is-a-good-regular-expression-to-match-a-url
    const regex = /https?:\/\/(www\.)?[-a-zA-Z0-9@:%._\+~#=]{1,256}\.[a-zA-Z0-9()]{1,6}\b([-a-zA-Z0-9()@:%_
\+.~#?&//=]*)/gi
    return url.trim().match(regex)
  }


  /***
   * Make the URL https
   *
   * url (string)
   * return (string) if url starts 'http://' change to 'https://' otherwise return original string
   */
   const makeHttps = (url) => {
     if (url.slice(0, 7).toLowerCase() === 'http://')
       return 'https://' + url.slice(7)
     return url
    return url
  }


/***
 * Convert to a clickable link
 *
 * url (string) A url, ex:  https://www.google.com
 * return (string)
 *         -- if url is a valid URL, return the HTML for an  tag
 *         -- else return the original string
 */
const makeClickable = (url) => {
  if (isValidUrl(url)) {
    return `CLICK HERE`
  }
  return url
}


/***
 * Parse string looking for a URL
 *
 * s (string) string that may contain a URL with optional text before or after the url
 * return (obj) return an oject in the form:
 *                   { before: 'The searh engine: ',
 *                     url: 'https://www.google.com',
 *                     after: '.' }
 *              Returns empty strings for any part that's missing
 */
const parseForUrl = (s) => {

   if (LOG) console.log("\n\nparseForUrl(), s = \n", s)


   // Parse for a URL that can be in the middle of s
   const regexUrl = /(.*)(https?:\/\/[^\s]+)(.*)/
   const aParsed = s.match(regexUrl)

   if (LOG) console.log("aParsed = ", aParsed)

   if (!aParsed) {
     if (LOG) console.log("-- no url found")
     return {
       before: '',
       url: '',
       after: ''
     }
   }

   if (LOG) console.log("PART 1 = ", aParsed[1])
   if (LOG) console.log("PART 2 = ", aParsed[2])
   if (LOG) console.log("PART 3 = ", aParsed[3])


   return {
     before: aParsed[1],
     url:    aParsed[2],
     after:  aParsed[3]
   }
  }

/***
 * Look in the 2nd column of this question's table for 's that contain a URL
 *   and convert to a clickable link, else do nothing.
 *
 * qid (string) The qid for the report element that just loaded
 */
const convertTextToLinks = (qid) => {


  const tdElems = document.querySelectorAll(`#report-${qid} table tr td:nth-child(2)`)
  if (LOG) console.log(">> tdElems = ", tdElems)
  tdElems.forEach(tdElem => {
    if (LOG) console.log(">> examining tdElem = ", tdElem)
    const parsedObj = parseForUrl(tdElem.innerText)
    if (LOG) console.log(">>>> parsedObj = ", parsedObj)


    if (isValidUrl(parsedObj.url))
      tdElem.innerHTML = parsedObj.before + makeClickable(parsedObj.url) + parsedObj.after
  })
}


/***
 * Callback for when charts are loaded into the report
 *
 * mutationRecords (array of MutationRecord)
 * observer (MutationObserver)
 */
function updateLoadedCharts(mutationRecords, observer) {


  if (LOG) console.log("\n\nXXXXXXXX callback: updateLoadedCharts() XXXXXXXXXXX")
  if (LOG) console.log("mutationRecords = ", mutationRecords)


  mutationRecords.forEach( mutationRecord => {


    mutationRecord.addedNodes.forEach(addedNode => {


      if (LOG) console.log("\n-----------------------\n-- addedNode = ", addedNode)
      if (LOG) console.log("-- addedNode.dataset = ", addedNode.dataset)


      if (addedNode.tagName === 'LI' && addedNode.dataset) {
        if (LOG) console.log("--- LI for QID: ", addedNode.dataset.qid)


        // try converting any
```

- report charts

```
        convertTextToLinks(addedNode.dataset.qid)


        /*if (UPDATE_QIDS.includes(addedNode.dataset.qid)) {
          if (LOG) console.log("Act on this one!!!")
          convertTextToLinks(addedNode.dataset.qid)
        }*/
      }
      else if (addedNode.tagName === 'DIV' && addedNode.dataset) {
        if (LOG) console.log("--- DIV for QID: ", addedNode.dataset.question)


        // try converting any report charts
        convertTextToLinks(addedNode.dataset.question)
        /*if (UPDATE_QIDS.includes(addedNode.dataset.question)) {
          if (LOG) console.log("Act on this one!!!")
          convertTextToLinks(addedNode.dataset.question)
        }*/

      }
      else {
        if (LOG) console.log("--- skipping")
      }
    })
  })
}


/**
 * main()
 */


if (LOG) console.log("\n\nStarting version: ", VERSION, "\n")


// Get the parent UL for the all of the question charts
//   so we can watch when a question is loaded.
// When a question loads it's replaced in the unorded list (
        ),
          //   so we can't watch the questions themselves, we have to
          //   watch the parent of the questions for changes to it's structure.
          // For efficiency longer reports will have a series of
                  's, so we need
                    //   to use quearySelectorAll().


                    const chartsParentElems = document.querySelectorAll('ul.js-spy-load')
                    if (LOG) console.log("chartsParentElems = ", chartsParentElems)


                    chartsParentElems.forEach(chartsParentElem => {
                      if (LOG) console.log("setup an observer")
                      // Setup an observer for DOM changes
                      const observer = new MutationObserver(updateLoadedCharts);
                      const config = {
                        childList: true,  // observe adding/removing child nodes
                        subtree: true     // and again since report loads can be many layered when slow
                      };
                      observer.observe(chartsParentElem, config);
                    })


                  })
```

Related Articles