Mirror the Order of Randomized Options in Later Questions

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.

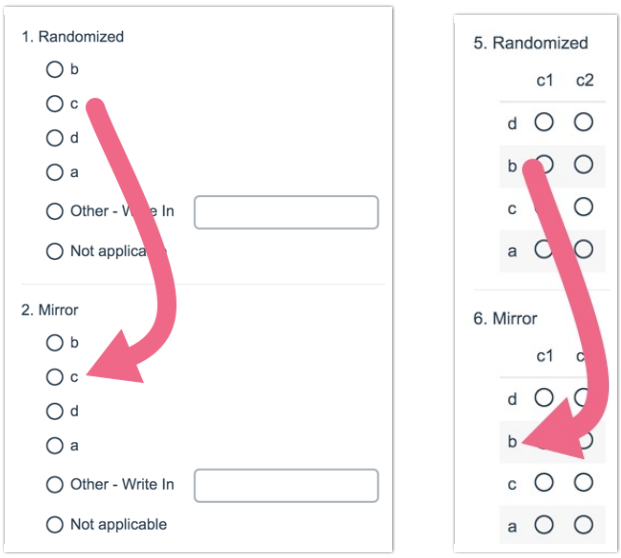
If you have customization ideas that you haven't figured out how to tackle, we're happy to be a sounding board for Alchemer features and functionality ideas that might meet your needs. Beyond this, check out our Professional Services; these folks have the scripting chops to help you to achieve what you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

Goal

Mirror the order of randomized question options in a later question of the same question type. This solution works for the following question types:

- Checkbox Question Type
- Checkbox Grid Question Type
- Radio Button Question Type
- Radio Button Grid Question Type

Examples:



Effort: ✔ ✔ ✔

Solution

The two questions may be on the same page or different pages. Because of this there are **two** scripts:

- One script for the page with the question with randomized options.
- One script for the page with the question that should mirror the order of the initial question.

> If the questions exist on the same page the second script ***must*** follow the first script.

Step 1: Choose **Style > Layout > Mobile Interaction > Standard**

Step 2: Setup question with randomized options

1. Add the initial question that will determine the order for both questions.
2. Select Layout Tab > Randomize > Randomize row order OR Layout > Randomize > Randomize columns
3. Add a **Hidden Value Action to the same page. The script will use this to** save the order of the randomized question options.
4. Add a Javascript Action below the **Hidden Value Action**, changing the **highlighted question IDs** below to the IDs for the question and Hidden Value Action added above.

```
/* Alchemer ver01

   Capture order of options of a randomized question

   Documentation and updates:  https://help.alchemer.com/help/mirror-the-order-of-randomized-option-in-later-questions
*/

/* ------------------------------------------------------
   DOM loaded
   ------------------------------------------------------ */
document.addEventListener("DOMContentLoaded", function() {

  // Checkbox or Radio button question ID with randomized option order
  const RANDOMIZED_QID = 25

  // Hidden Value or Textbox question ID to save the order
  const SAVE_ORDER_QID = 28

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

  const LOG = false

  if (LOG) console.log("saving...")

  /* ---------------------------------------
     Helper: Get a SurveyGizmo element on the page.
           Ex: In survey 1234567 on page ID 12 the call getSgId(123, "element")
              returns HTML element for "sgE-1234567-12-123-element"
     --------------------------------------- */
  function getSgElemById(qid, oid = "element") {
    let surveyInfo = SGAPI.surveyData[Object.keys(SGAPI.surveyData)[0]]
    let id = "sgE-" + surveyInfo.id + "-" + surveyInfo.currentpage + "-" + qid + "-" + oid
    let elem = document.getElementById(id)
    if (!elem)
      alert("Javascript error: can't find element with id = " + id)
    return elem
  }

  /* ---------------
     getQuestionType() - returns 'radio-button', 'checkbox', 'grid', or null
     --------------- */
  let getQuestionType = (quesElem) => {
    if (quesElem.classList.contains('sg-type-radio'))
      return 'radio-button'
    if (quesElem.classList.contains('sg-type-checkbox'))
      return 'checkbox'
    if (quesElem.classList.contains('sg-type-table'))
      return 'grid'
    alert("Javascript error: unknown question type")
    return null
  }

  /* ------------------------------------------------------
     saveOrder()

     Save a '|' delimited list of titles for each option in the randomized question
     ------------------------------------------------------ */
  let saveOrder = (randomizedQID, saveOrderQID) => {

    /* ---------------
       straightenQuotes() -- convert angled quotes to straigt
```

```
    ---------- */
  let straightenQuotes = (s) => {
    return s.replace(`“`, `"`).replace(`”`, `"`).replace(`‘`, `'`).replace(`’`, `'`)
  }

  /* ----------
     getGridOrder() -- get string with order of Grid Question
     ---------- */
  let getGridOrder = (quesElem) => {
    let rowElems = quesElem.querySelectorAll('.sg-question-options tbody tr')
    for (let i = 0; i < rowElems.length; i++) {
      order += ((order) ? '|' : '' ) + straightenQuotes(rowElems[i].querySelector('th').innerText.trim())
    }
    return order
  }

  /* ----------
     getRbCbOrder() -- get string with order of radio button or checkbox question
     ---------- */
  let getRbCbOrder = (quesElem) => {
    let inputClassname = (getQuestionType(quesElem) === 'radio-button') ? 'sg-input-radio' : 'sg-input-checkbox'

    // carefully select only the radio buttons/checkboxes, not the Other Write-in Textbox
    let inputElems = quesElem.querySelectorAll(`.sg-question-options input.${inputClassname}`)
    for (let i = 0; i < inputElems.length; i++) {
      order += ((order) ? '|' : '' ) + straightenQuotes(inputElems[i].getAttribute("aria-label"))
    }
    return order
  }

  /* ----------
     main()
     ---------- */

  let order = ''

  let quesElem = getSgElemById(randomizedQID, "box")

  // Get order from radio button, checkbox, or grid question type
  switch(getQuestionType(quesElem)) {
    case 'radio-button':
    case 'checkbox': {
      order = getRbCbOrder(quesElem)
      break
    }
    case 'grid':
      order = getGridOrder(quesElem)
      break
  }

  getSgElemById(saveOrderQID).value = order
  if (LOG) console.log("saved value = ", getSgElemById(saveOrderQID).value)
}

/* ----------
   main()
   ---------- */
saveOrder(RANDOMIZED_QID, SAVE_ORDER_QID)
})
```

Step 3: Setup question that mirrors the option order of the initial question

1. Copy the question added in Step 1 to ensure the options are exactly the same
2. Select **Layout > Randomize > Uncheck Randomize Columns and Randomize row orde** r OR
3. Move the question to a later page if desired.
4. Add the Javascript Action below setting the highlighted question IDs for the **Hidden Value Action** added in Step 1 and the question to mirror the initial question option order.

```
/* Alchemer ver 01

   Mirror order of previously randomized question options

   Documentation and updates:  https://help.alchemer.com/help/mirror-the-order-of-randomized-option-in-later-questions
*/

/* ------------------------------------------------------
   DOM loaded
   ------------------------------------------------------ */
document.addEventListener("DOMContentLoaded", function() {

  // the value from the the Hidden Value Action that stores the order of the randomized options
  let ORDER_STRING = getSgElemById_value(28) || '[question("value"), id="28"]'

  // the question id of the question to mirror the randomized question option order
  const MIRROR_QID = 26

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

  const LOG = false

  if (LOG) console.log("mirroring... = ", ORDER_STRING)

  /* ---------------------------------------
     Helper: Get a SurveyGizmo element on the page.
           Ex: In survey 1234567 on page ID 12 the call getSgId(123, "element")
              returns HTML element for "sgE-1234567-12-123-element"
     --------------------------------------- */
  function getSgElemById(qid, oid = "element", suppressError = false) {
    let surveyInfo = SGAPI.surveyData[Object.keys(SGAPI.surveyData)[0]]
    let id = "sgE-" + surveyInfo.id + "-" + surveyInfo.currentpage + "-" + qid + "-" + oid
    let elem = document.getElementById(id)
    if (!elem && !suppressError)
      alert("Javascript error: can't find element with id = " + id)
    return elem
  }

  /* ---------------------------------------
     Helper: get value of a textbox/hidden value QID or null if it doesn't exist
     --------------------------------------- */
  function getSgElemById_value(qid) {
    let elem = getSgElemById(qid,'element',true)
    return elem ? elem.value : null
  }

  /* ---------------
```

```
  getQuestionType() - returns 'radio-button', 'checkbox', 'grid', or null
  ---------- */
let getQuestionType = (quesElem) => {
 if (quesElem.classList.contains('sg-type-radio'))
  return 'radio-button'
 if (quesElem.classList.contains('sg-type-checkbox'))
  return 'checkbox'
 if (quesElem.classList.contains('sg-type-table'))
  return 'grid'
 alert("Javascript error: unknown question type")
 return null
}

/* ----------------------------------------------------
  pullElemsFromDOM()
  Return array of elems pulled out of the DOM for the mirrored question
  ---------------------------------------------------- */
let pullElemsFromDOM = (mirrorElem) => {
 let unorderedElems = []
 switch(getQuestionType(mirrorElem)) {
  case 'radio-button':
  case 'checkbox':
   let liElems = mirrorElem.querySelectorAll('li')
   for (let i = 0; i < liElems.length; i++)
    unorderedElems.push(liElems[i].parentElement.removeChild(liElems[i]))
   break
  case 'grid':
   let rowElems = mirrorElem.querySelectorAll('.sg-question-options tbody tr')
   for (let i = 0; i < rowElems.length; i++) {
    unorderedElems.push(rowElems[i].parentElement.removeChild(rowElems[i]))
   }
   break
 }
 return unorderedElems
}

/* ---------------
  getContainer() - get the container to add the ordered elements into
  ---------- */
let getContainer = (mirrorElem) => {
 switch(getQuestionType(mirrorElem)) {
  case 'radio-button':
  case 'checkbox':
   return mirrorElem.querySelector("ul")
   break
  case 'grid':
   return mirrorElem.querySelector("tbody")
   break
 }
 return null
}

/* ---------------
  getOptionTitle() - get the option title from an element
  ---------- */
let getOptionTitle = (questionType, elem) => {

 /* ---------------
   straightenQuotes() -- convert angled quotes to straigt
   ---------- */
 let straightenQuotes = (s) => {
  return s.replace(`“`, `"`).replace(`”`, `"`).replace(`‘`, `'`).replace(`’`, `'`)
 }

 /* ---------------
```

```
    main()
    ---------- */
  switch(questionType) {
    case 'radio-button':
    case 'checkbox':
      return straightenQuotes(elem.querySelector('input').getAttribute('aria-label'))
      break
    case 'grid':
      return straightenQuotes(elem.querySelector('th').innerText.trim())
      break
  }
  return null
}

/* ----------
   getUnorderedElem() - get unoirdered elem matching optionTitle
   ---------- */
let getUnorderedElem = (questionType, unorderedElems, optionTitle) => {
  if (LOG) console.log("Search for: ", optionTitle)
  let elem = unorderedElems.find((testElem) => {
    if (LOG) console.log("-- testing: ", getOptionTitle(questionType, testElem))
    return optionTitle === getOptionTitle(questionType, testElem)
  })
  if (!elem)
    alert('Javascript error - missing option in mirrored list: ' + optionTitle)
  return elem
}

/* ----------
   fixupDisplayDetails() - fix classes that affect appearance
   ---------- */
let fixupDisplayDetails = (mirrorElem) => {

  /* ----------
     fixup radio button / checkbox to have the first/last classes in the correct spots
     ---------- */
  let fixupFirstLastClass = () => {
    let liElems = mirrorElem.querySelectorAll("li")
    for (let i = 0; i < liElems.length; i++) {
      liElems[i].classList.remove('sg-first-li')
      liElems[i].classList.remove('sg-last-li')
    }
    liElems[0].classList.add('sg-first-li')
    liElems[liElems.length - 1].classList.add('sg-last-li')
  }

  /* ----------
     fixup grid to alternate the gray shading for each row
     ---------- */
  let fixupAlternateRowGray = () => {
    let oddRow = true
    mirrorElem.querySelectorAll("tbody tr").forEach( row => {
      row.classList.remove("sg-odd-row")
      row.classList.remove("sg-even-row")
      // apply correct sg-odd-row or sg-even-row if not hidden
      if (!row.classList.contains("sg-hide")) {
        row.classList.add(oddRow ? "sg-odd-row" : "sg-even-row")
        oddRow = !oddRow
      }
    })
  }

  /* ----------
     main
     ---------- */
```

```
    switch(getQuestionType(mirrorElem)) {
      case 'radio-button':
      case 'checkbox':
        fixupFirstLastClass()
        break
      case 'grid':
        fixupAlternateRowGray()
        break
    }
  }

  /* ------------------------------------------------------
     mirrorOrder()
     Mirror order of previously randomized question
     ------------------------------------------------------ */
  let mirrorOrder = (mirrorQID, orderString, ) => {
    let mirrorElem = getSgElemById(mirrorQID, 'box')
    let unorderedElems = pullElemsFromDOM(mirrorElem)
    let orderArray = ORDER_STRING.split('|')
    let container = getContainer(mirrorElem)
    if (LOG) console.log("ordering...")
    orderArray.forEach((optionTitle) => {
      let unorderedElem = getUnorderedElem(getQuestionType(mirrorElem), unorderedElems, optionTitle)
      container.appendChild(unorderedElem)
    })
    fixupDisplayDetails(mirrorElem)
  }

  /* ------------------------------------------------------
     main()
     ------------------------------------------------------ */
  mirrorOrder(MIRROR_QID, ORDER_STRING)
})
```

Related Articles