How to implement real-time interaction between participants in online surveys: A practical guide to SMARTRIQS

Andras Molnar [a]

[a]Department of Social and Decision Sciences, Carnegie Mellon University

Abstract ■ SMARTRIQS is an open-source add-on to the popular survey platform Qualtrics, offering researchers the ability to design online surveys that feature real-time interaction between participants—including live text chat—without requiring researchers to learn any programming language or install any software. Using SMARTRIQS does not incur any additional costs to researchers who have an institutional Qualtrics account. This paper not only provides an overview of SMARTRIQS and its potential applications but also walks readers through the step-by-step instructions for setting up a particular study (Dictator Game with chat). These instructions start from the very basics, assuming no prior expertise in online experimentation, and are accessible to everyone, even those who are less—or not at all—familiar with Qualtrics.

Keywords ■ chat, real-time interaction, survey design, tutorial. **Tools** ■ Qualtrics.

andrasm@andrew.cmu.edu

10.20982/tqmp.16.4.p334

Acting Editor ■ Denis Cousineau (Université d'Ottawa)

Reviewers
■ One anonymous reviewer

Introduction

Online experimentation is becoming increasingly popular in social sciences (Arechar, Gächter, & Molleman, 2018; Gosling & Mason, 2015)—largely due to the massive technological improvements in survey software (e.g., "Qualtrics," 2020; "SurveyMonkey," 2020), and the advance of large-scale online participant panels (Chandler, Rosenzweig, Moss, Robinson, & Litman, 2019) and crowdsourcing platforms such as Amazon Mechanical Turk (Bohannon, 2016; Mason & Suri, 2012; Paolacci & Chandler, 2014), Prolific, or CrowdFlower (Peer, Brandimarte, Samat, & Acquisti, 2017). These tools allow researchers to collect and analyze data at an unprecedented scale, speed, and efficiency, which makes them superior to conventional lab studies (i.e., pen-and-paper experiments or computerized interactions implemented on local networks, e.g., via z-Tree, Fischbacher, 2007) in many ways.

While lab studies typically provide researchers more experimental control over the procedures and have lower attrition rates than online experiments (Arechar et al., 2018; Zhou & Fishbach, 2016), they have several major limitations (see "Offline RTI" in Table 1). Offline studies are usually more expensive—both in terms of participant compensation and administrative costs—require a physical lab and equipment, and have considerably smaller and less heterogeneous subject pools (Berinsky, Huber, & Lenz, 2012; Henrich, Heine, & Norenzayan, 2010). Also, since the number of participants who can concurrently participate in lab studies is limited, data collection takes significantly longer.

Online studies can overcome most of the limitations of conventional lab studies, but they also pose unique methodological challenges. Most importantly, implementing real-time interaction and communication in an online experiment is notoriously difficult, unlike allowing participants to seamlessly interact with each other in a conventional lab environment. Setting up such a study takes a considerable amount of time, and requires researchers either to have advanced programming skills, or to pay for third-party services and products. These challenges prevent many social scientists from utilizing online methods in their research. Researchers who wish to study social interaction, group dynamics, communication, or any other form of interpersonal decision-making, often have to resort to simpler but also inferior alternatives, which only al-

Table 1 ■ A comparison of methods for studying social interaction between participants

Type	Method	Main advantages	Limitations
(1) Offline RTI*	Lab (e.g., z-Tree, pen-and-paper studies)	High experimental control Low attrition rates	Requires lab & equipment; Expensive; time consuming; Limited subject pool
(2) Online non-RTI*	"Wave" recruitment	Does not require programming	Potential selection confound; Slow; manual matching
	Strategy method	Does not require programming; More observations per subject	Might affect behavior; Manual matching
	Deception	Does not require programming; Easy setup	Subject pool contamination; Potential IRB issues
(3) Online RTI*	Experimental platforms (e.g., oTree)	Highly customizable; Repeated, complex interaction	Requires programming
	Third-party services (e.g. iDecisionGames)	Highly customizable; Advanced features (e.g., video chat)	Expensive; closed source; Limited experimental control
	Specialized applications (e.g., Chatplat)	Does not require programming; Easy setup; platform-independent	Limited to one type of interaction
	SMARTRIQS	Does not require programming; Easy setup; highly customizable	None of the above (see Table 2)

Note. "RTI" indicates true (non-deceptive) **real-time interaction** between participants.

low for non-real-time interaction or simulated interaction between participants (see "Online non-RTI" in Table 1).

Simple but flawed alternatives to real-time interaction in online studies

There are three simple methods that allow researchers to study social interactions in online experiments, without requiring participants to interact in real-time: "wave" recruitment, the strategy method, and deception. While all of these alternatives help researchers to overcome the limitations of conventional lab studies, they also introduce new limitations, which, depending on the research objectives and the standards of the researcher's discipline, can render these methods unsuitable for conducting online experiments.

Wave recruitment

Wave recruitment refers to the practice of recruiting a group of participants, recording their decisions, and displaying these decisions to a second group of participants, who are recruited in subsequent sessions. Although this alternative does not require any programming skills and is relatively easy to set up, researchers have to match participants and transmit their responses manually, which is not only very labor-intensive and error-prone, but also makes complex interactions (e.g. multiple rounds, large groups) extremely challenging—if not impossible—to implement.

Strategy method

When using the strategy method, participants make conditional decisions for each possible action of other participants. Similarly to the wave recruitment method, participants are "matched" post hoc, and decisions are implemented only after the experiment. While the strategy method allows researchers to collect more data per participant, it is also very labor-intensive and error-prone, since it requires the experimenter to manually match participants and determine outcomes. More importantly, participants might behave differently when their choices and preferences are elicited using the strategy method, as opposed to a direct-response method, which might affect the validity of the experiment. For instance, Casari and Cason (2009) found that people showed significantly lower levels of trustworthiness when using the strategy method, compared to the direct-response method, which suggests that the strategy method should be used with caution when studying trust-related behaviors.

Deception

Deception is arguably the most problematic alternative to implementing real-time interaction in online studies. When using deception, participants are explicitly told or led to believe that they are interacting with other people, while they are actually "interacting" with the computer. This not only violates the research standards of several disciplines (e.g., experimental economics and experimental finance), but also conflicts with most Institutional Review Board (IRB) policies, which explicitly specify that deception can only be used when there are no reasonably effective alternative methods available to achieve the goals of the research. The practice of deceiving subjects out of convenience, because it is *difficult* (but not impossible) to implement real-time interaction online, is thus violating IRB guidelines. Furthermore, the excessive use of participant deception can contaminate subject pools—i.e., erode the credibility of experimental instructions and alter participants' behavior over time—especially in large-scale crowdsourcing platforms such as MTurk or Prolific (for a detailed discussion and a review of empirical evidence on the effects of deception, see Hertwig & Ortmann, 2008).

Existing methods that allow for real-time interaction in online studies

The existing solutions that allow for conducting interactive studies online, can be classified into three broad categories: standalone experimental platforms, closed-source third-party services, and specialized applications (see "Online RTI" in Table 1).

Standalone experimental platforms

The first category, standalone experimental platforms such as ConG (Pettit, Friedman, Kephart, & Oprea, 2014), LIONESS Lab (Giamattei, Molleman, Seyed Yahosseini, & Gachter, 2019), MWERT (Hawkins, 2015), nodeGame (Balietti, 2017), oTree (Chen, Schonger, & Wickens, 2016), Psynteract (Henninger, Kieslich, & Hilbig, 2017), or SoPHIE (Hendriks, 2012) offer researchers great flexibility in study design and are typically freely accessible and open-source. However, all of these platforms require users to have at least some expertise in a programming language (e.g., Python, JavaScript, PHP). This is a limitation that prevents many social scientists from studying behavior in online settings, unless they are willing to invest a substantial amount of time in learning a programming language.

Third-party services

Another alternative is to outsource programming tasks to a third-party or to hire a professional programmer, however, doing so can be prohibitively expensive, especially for smaller labs or junior researchers who have limited funds. Furthermore, individual programmers usually lack the experience with behavioral experimental research and need a lot of guidance and consultation when designing and editing studies. By contrast, there are private companies that specialize in designing and conducting online experiments (e.g., iDecisionGames, https://idecisiongames.com). While these services offer great flexibility in experimental design and allow researchers to use advanced features such as live video chat between participants, researchers have no direct control over the design process and have to communicate every minor edit to the third-party company, which can slow down the design and testing phase substantially. Furthermore, since these are for-profit companies, their products are closed-source, which forces researchers to keep paying for services, even if they just want to run an exact replication of an experiment conducted by another lab. Since replicability and transparent research practices are increasingly important in the social sciences (see e.g., Collaboration*, 2015), this is a rather serious limitation.

Specialized applications

Finally, there are specialized applications such as Chatplat (https://www.chatplat.com), that address the limitations of both generic experimental platforms and third-party services. These applications are free, do not involve any programming, and can be set up relatively easily. However, these advantages come at the expense of flexibility: Specialized applications are designed for a specific type of interaction, and their applicability is limited to a narrow range of experiments. For example, Chatplat allows participants to chat with each other in online surveys, but does not allow for setting up different types of studies (e.g., which require transmitting decisions).

A new tool for studying real-time interaction in online studies: SMARTRIQS

The previous sections highlight not only the demand for, but also the limitations of, methods that allow researchers to implement real-time interaction between participants in online studies. These methods are either too difficult to use (i.e., require programming), too expensive, or have limited applications. SMARTRIQS is a first-of-its-kind solution that addresses all of these challenges, and allows researchers to conduct interactive experiments without the hassle of programming or paying for extra services. Table 2 sum-

Table 2 ■ Features and limitations of SMARTRIQS

Feature	Description	Limitation
Cost and access	Free* and open-source	Requires Qualtrics account*
Implementation	Default or custom (private) server	–
Ease of use	Requires no programming; Requires no installation	–
Integration with Qualtrics	All data saved in Qualtrics	–
Participant matching	Fixed groups	No re-matching
Group size	2–8 participants per group; Unlimited number of groups	Max. 8 participants per group
Randomization	Random or custom assignment to roles and conditions	–
Type of interaction	One-shot or repeated; synchronous or sequential	–
Types of data that can be transmitted	Any data type that is supported in Qualtrics (numeric, text, scale)	Transmission of personal data may be prohibited (consult IRB)
Supported communication	Highly customizeable text chat (e.g., group or private chat; multiple stages; custom format)	No audio or video chat
Advanced features	Waiting rooms; Built-in math operations (e.g., sum, rank, min, max); Dropout management; Bots and default responses; Free survey templates and demos; Data collection monitor	

Note. SMARTRIQS requires the researcher to have an institutional subscription to Qualtrics, but there is no additional fee associated with using SMARTRIQS.

marizes the main features and limitations of SMARTRIQS.

The conceptual framework and the architecture of SMARTRIQS were introduced in Molnar (2019). In essence, SMARTRIQS is a collection of generic scripts that can be added to any Qualtrics survey, allowing the researcher to match survey respondents in real-time and transmit responses between them. However, researchers do not have to edit any of these scripts when creating their own studies, instead, the design and customization process is done entirely within the Qualtrics Survey Editor, which has a very intuitive and user-friendly graphical user interface.

While Molnar (2019) provided a comprehensive conceptual overview, it addressed a specialist audience (researchers in experimental finance) and did not provide any instructions to *how* to design interactive studies. The current paper, by contrast, focuses on the practical applications of SMARTRIQS and provides researchers step-by-step instructions and hands-on guidance. The next section lists and discusses various applications of SMARTRIQS across several disciplines of the social sciences, while the rest of the paper walks the reader through a detailed tutorial. This guide is designed to help researchers to become familiar with SMARTRIQS and to set up and customize interactive studies effortlessly, without requiring them to have *any* prior experience with online experimentation.

Applications of SMARTRIQS

SMARTRIQS has a great potential for becoming a fundamental tool of social scientists, as it offers researchers the ability to run interactive studies online with unprecedented ease and efficiency—without having to learn any programming language, installing any software, or paying for third-party services. Social scientists will find many potential uses of SMARTRIQS (see Table 3).

First and foremost, social scientists will be able to run interactive experiments online, which will allow them to recruit participants from larger and more diverse samples than when conducting conventional lab experiments. They can also easily convert their existing (non-interactive) Qualtrics surveys into interactive ones. Being able to match participants in real-time also makes many instances of deception unnecessary, and allows for studying human

Table 3 ■ Potential applications of SMARTRIQS across disciplines

Fields / disciplines	Sample applications
Cognitive Science & Linguistics	Coordination & joint action; Text- and sentiment analysis in real-time communication
Economics	Allocation & reciprocity (e.g., Dictator Game, Ultimatum Game); Collective decision-making (e.g., Public Goods Game); Hierarchical beliefs & expectations (e.g., p-Beauty contest)
Finance	Simple market interactions (e.g., multiple buyers and sellers); Investment decisions (e.g., Trust/Investment Game); Auctions (e.g. blind, Vickrey)
Management & Organizational Research	Group processes (e.g., collaboration, competition); Work & principal-agent problems (effort allocation & provision); Negotiation and strategic interactions
Marketing & Consumer Research	Buyer-seller interactions (e.g., endowment effect studies); Consumer focus group studies
Political Science	Persuasion & spread of beliefs; Political focus group studies; Voting behavior
Social Psychology	Moral behavior (e.g., dishonesty, cheating, punishment); Identity & group dynamics (e.g., minimal group paradigms); Compliance to, and enforcement of social norms; Impression management & social signaling
Miscellaneous applications	Educational applications (in-class experiments); Field applications (field studies with participant interaction); Lab studies (computerized interaction in conventional labs)

behavior in more naturalistic contexts.

Economists and researchers studying game theory can easily set up various "classic" economic games (e.g., Dictator Game, Public Goods Game), as well as novel, custom experimental designs. Researchers interested in experimental finance can conduct simple market experiments, and will be able to simulate investment decisions and auctions online.

There are plenty of potential applications in managerial and organizational contexts as well: For example, researchers can study various group processes such as collaboration and competition, task allocation, and effort provision within groups. The advanced chat feature allows scientists to study various aspects of communication: persuasion, negotiation, impression management, or even the linguistic properties of conversations, which are relevant not only in organizational research but also in political science or consumer research.

SMARTRIQS allows social psychologists to study a wide range of phenomena in online contexts, including but not limited to moral behavior, group dynamics, social norms, and impression management strategies. Psychologists interested in personality and individual differences can supplement real-time interactions with various inventories to study how personality traits and attitudes correlate with social behavior.

Importantly, the applications of SMARTRIQS are not limited to experiments conducted on Amazon Mechanical Turk or similar crowdsourcing platforms. Since SMARTRIQS has minimal technical requirements, and does not require participants to install any software—any device with an Internet access and a browser that supports HTML5 and JavaScript suffices—researchers can recruit participants *in virtually any context*: in conventional computer labs, in classrooms, or even in the field. This makes it possible to conduct studies with real-time interaction in places where it would have been challenging before (e.g., developing countries, remote locations, events). SMARTRIQS can also be used for educational purposes: Teachers can set up simple interactive experiments, then have their students complete these studies in class. With the built-in Results-Reports function of Qualtrics, teachers can even display the results to students in real-time.

SMARTRIQS can also be combined with other useful tools developed for Qualtrics. For example, researchers who study dynamic cognitive processes can supplement SMARTRIQS surveys with a tool that captures mouse cursor trajectories (Mathur & Reichling, 2019). Similarly, researchers who want to measure how much time participants spend on-screen versus off-screen (a crucial metric of participant attention and engagement) can supplement SMARTRIQS with TaskMaster, a simple tool that tracks participants' activity (Permut, Fisher, & Oppenheimer, 2019).

Finally, since SMARTRIQS is open-source and provides a standardized generic framework for interactive studies, it will not only serve as a useful experimental tool but also contribute to, and propagate, open research practices.

Step-by-Step Tutorial

This section provides a comprehensive guide to using SMARTRIQS and walks the reader through the step-by-step instructions for setting up a particular study: the Dictator Game. In this classic dyadic interaction, participants are randomly assigned to either the role of the "Allocator" or the "Recipient." At the beginning of the experiment the Allocator is endowed with an amount (e.g., 100 tokens). Then, she can transfer any amount out of her endowment to the Recipient. The Recipient does not make any decision, simply receives whatever amount is transferred to her. Variations of this experiment are widely used to measure social preferences and attitudes towards others (Forsythe, Horowitz, Savin, & Sefton, 1994). To demonstrate the live chat feature of SMARTRIQS, this section also covers how to allow chat between participants.

Prerequisites

Researchers need the following before they start designing interactive studies with SMARTRIQS:

1. **An institutional subscription to the Qualtrics Survey Software**. Free or trial accounts are not supported. This tutorial does not assume any prior experience with Qualtrics. However, having some familiarity with certain Qualtrics concepts (e.g., Survey Flow, Embedded Data, Branch Logic, Piped Text) is recommended, as it will make it easier to follow this guide and to understand how SMARTRIQS works.

   For those who are new to Qualtrics or wish to refresh their knowledge, there are plenty of free tutorials available online. For example, Dare McNamara has two excellent video tutorials: Beginner Qualtrics Training (10 minutes) and Advanced Qualtrics Training (17 minutes).
2. **A SMARTRIQS researcher ID**, which can be obtained by submitting the registration form. Researchers have to provide their full name and email address, and accept the Data Policy, in order to receive their unique researcher ID. Researchers may also provide additional information about their affiliation, status, and field of study, but these are all optional, not required for obtaining a researcher ID.
3. **One or more SMARTRIQS survey templates** ("Qualtrics Survey Format" files, or "QSF" for short). The purpose of these templates is to offer researchers a solid starting point when designing new studies, so that they don't have to set up everything from scratch. Importantly, all of these templates contain the generic scripts that allow interaction between participants. While researchers can also add these scripts (source code: https://github.com/andras-molnar/smartriqs) manually to any blank Qualtrics survey, it is strongly recommended to start with one of the SMARTRIQS templates, which already have all the necessary components. The survey templates (QSF files) can be downloaded from the OSF repository.

Importing survey templates (QSF files) to Qualtrics

Throughout this tutorial we will use the "Generic Interactive Survey Template" (GIST). However, researchers can start with any template that works best for them. The GIST has every SMARTRIQS feature (including chat) and using it is recommended for custom studies that are very different from other templates. First, download the "Generic_Interactive_Survey_Template_GIST.qsf" file from the OSF repository. Then, log in to Qualtrics, click on "Create new project" (top right corner of the main page), then select "Survey" under "Create your own" and click on "From a File / Choose file." In the pop-up window, select the QSF file you wish to import ("Generic_Interactive_Survey_Template_GIST.qsf" in this tutorial) and click "Open." Finally, rename the project and click on "Get Started." For more information about how to import QSF files to Qualtrics, visit the corresponding Qualtrics support page.

Matching participants: the MATCH block

Before participants can chat or interact with each other, they have to be matched first. In SMARTRIQS surveys this happens in the "MATCH block." SMARTRIQS offers a lot of flexibility in customizing interactions (e.g., group size, number of stages, participant roles), and most of these settings must be set before the MATCH block. To set up a MATCH block, open the imported survey and then open the Survey Flow (top left of the main page). On the top of the Survey Flow there are two green panels titled "Required parameters" and "Optional parameters" (see Figure 1). Note that blank Qualtrics surveys do not have these panels, only the SMARTRIQS survey templates contain these. When creating an interactive survey from scratch (instead of importing an existing SMARTRIQS template), the researcher has to add these panels manually.

The two panels contain the Embedded Data fields (olive green), which are essentially the variables and parameters used in Qualtrics surveys. SMARTRIQS studies use Embedded Data for two purposes: 1) as parameters that define the characteristics of the interaction, and 2) as variables that store participants' responses. Throughout the subsequent sections of paper, **boldface** text indicates Embedded Data fields. After scrolling down in the Survey Flow, there will be several grey panels and additional green panels with more Embedded Data. Each grey panel represents a Question Block. Blocks are either built-in SMARTRIQS blocks that are responsible for various types of interaction (MATCH, CHAT, SEND, GET, and COMPLETE) or standard Qualtrics blocks that contain regular survey items (e.g., instructions, decisions, questionnaires). The latter can be added and edited manually in the Survey Editor. The first panel of Embedded Data contains the required parameters for matching participants. These parameters must be set before the MATCH block:

- **researcherID**: Enter the SMARTRIQS researcher ID here (obtained by submitting the registration form).
- **studyID**: Enter the name of the study here. The name can be any combination of alphanumeric characters (0–9, A–Z, a–z), dash (-) or underscore (_), up to a length of 256 characters. For example: *Dictator_Game_Demo-1*. Note that names are case-sensitive.
- **groupID**: This is the field that identifies groups within a study. Leave this field blank, this will be automatically populated by SMARTRIQS.
- **participantID**: Keep the default value in this field (*$e://Field/ResponseID*) in order to assign the built-in Qualtrics response ID to participants (recommended). This guarantees that everyone will receive a random, unique, and anonymous ID.
- **groupSize**: Determine how many participants should be in each group. SMARTRIQS supports group interaction up to 8 participants per group. Set this to 2 for dyadic interactions.
- **numStages**: Determine the number of decisions to be transmitted across participants. This usually corresponds to experimental stages. If participants take turns or make multiple decisions, there should be multiple stages. In the Dictator Game example, we set this field to 1, since there is only one decision that needs to be transmitted (the Allocator's transfer).
- **roles**: Determine the set of available roles within each group, where the roles are separated by commas. For example: *Allocator,Recipient*. Note that there is no space between the comma and the roles. Since SMARTRIQS uses these roles to identify participants within groups, it is important that each participant should have a unique "role" within their group, even if their tasks are identical (e.g., simple group chat). Also note that the number of roles must be the same as the value of the **groupSize** parameter (2 in this example). Roles are case-sensitive: Make sure to use the correct cases when setting up role-specific blocks in the Survey Flow (see section "Setting up role-dependent questions and blocks: Branch Logic").
- **participantRole**: Determine the role of the participant. To assign roles randomly within groups, enter *random* here. To learn more about custom role assignment methods, visit https://smartriqs.com/randomization.
- **timeOutLog**: SMARTRIQS saves error logs in this field, for example, when participants drop out from the study. Leave this field blank, this will be automatically populated.

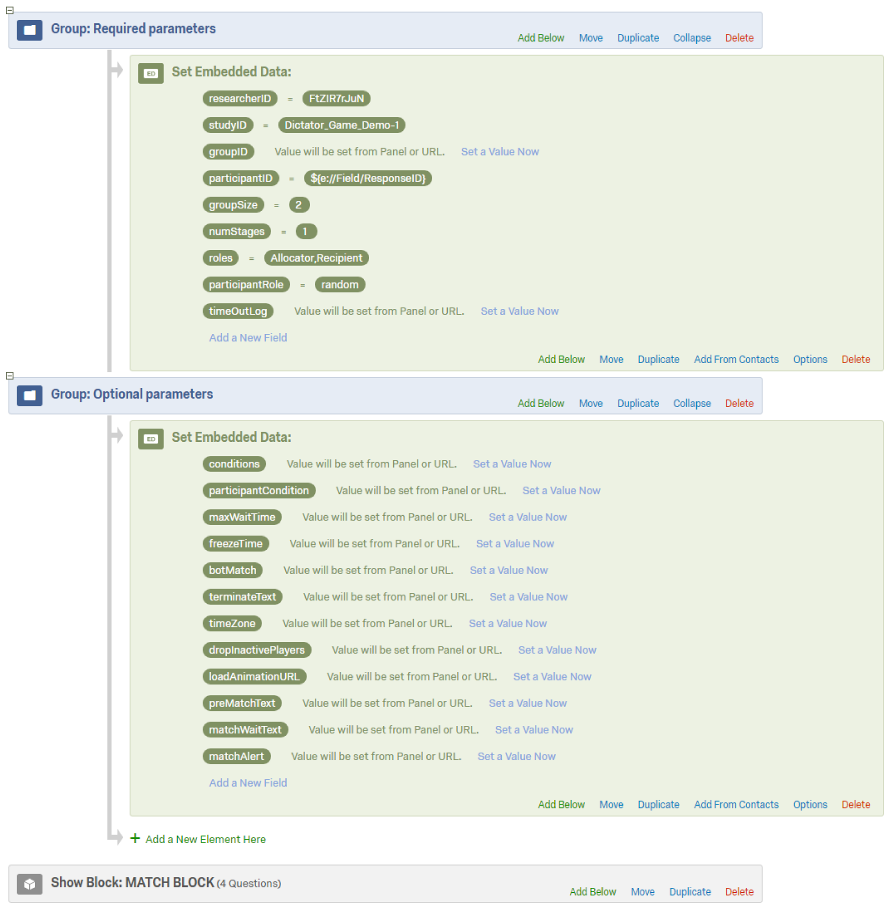
The second panel contains optional parameters, which allow researchers to customize the non-essential features of the MATCH block (e.g., messages displayed before and during matching). In this tutorial we leave these fields blank, and let SMARTRIQS apply the default values. To learn more about optional parameters, visit https://smartriqs.com/matching/#matchBlock.

Real-time communication between participants: the CHAT block

One of the most advanced features of SMARTRIQS is its built-in chat interface, which allows two or more participants to chat in real-time. The chat feature is flexible and customizable: Researchers can set up chat with or without a time limit; allow chat between the whole group or just some participants within the group; have multiple stages of chat within the study (e.g., interrupt the chat with a task, then continue the chat after participants finish the task); or customize the design of the chat window (e.g., time stamps, window size, chat instructions).

SMARTRIQS saves the content of the CHAT block, including participants' roles, messages, and time stamps (if set) to an Embedded Data field (chat log) that can be accessed by downloading the collected data. The CHAT block

Figure 1 ■ Sample screenshot of the setup of required and optional parameters before the MATCH block (Dictator Game). Note that both the “Required parameters” and “Optional parameters” panels must be placed before (above) the MATCH block in the Survey Flow.



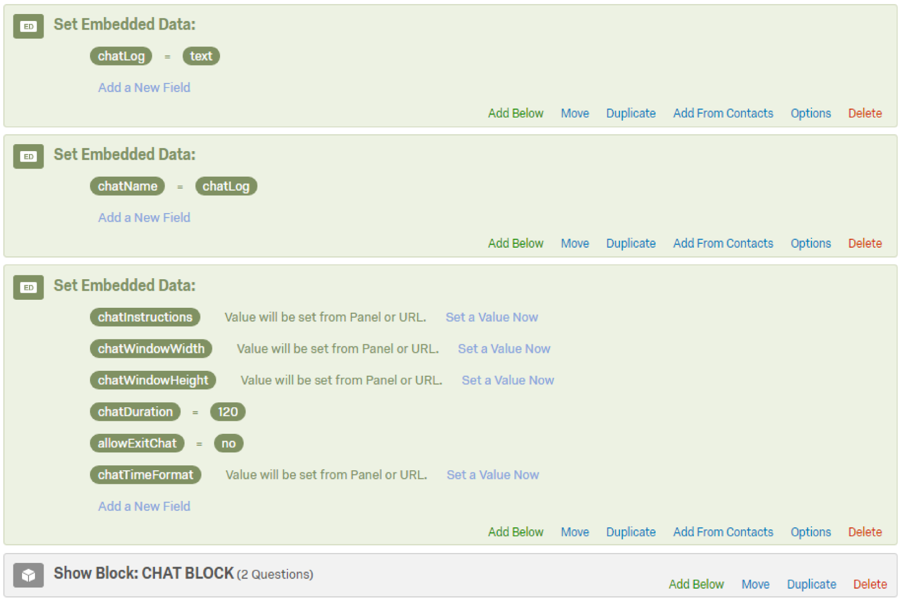
```
Group: Required parameters
    Set Embedded Data:
        researcherID = FtZIR7rJuN
        studyID = Dictator_Game_Demo-1
        groupID  Value will be set from Panel or URL.  Set a Value Now
        participantID = ${e://Field/ResponseID}
        groupSize = 2
        numStages = 1
        roles = Allocator,Recipient
        participantRole = random
        timeOutLog  Value will be set from Panel or URL.  Set a Value Now
Group: Optional parameters
    Set Embedded Data:
        conditions  Value will be set from Panel or URL.  Set a Value Now
        participantCondition  Value will be set from Panel or URL.  Set a Value Now
        maxWaitTime  Value will be set from Panel or URL.  Set a Value Now
        freezeTime  Value will be set from Panel or URL.  Set a Value Now
        botMatch  Value will be set from Panel or URL.  Set a Value Now
        terminateText  Value will be set from Panel or URL.  Set a Value Now
        timeZone  Value will be set from Panel or URL.  Set a Value Now
        dropInactivePlayers  Value will be set from Panel or URL.  Set a Value Now
        loadAnimationURL  Value will be set from Panel or URL.  Set a Value Now
        preMatchText  Value will be set from Panel or URL.  Set a Value Now
        matchWaitText  Value will be set from Panel or URL.  Set a Value Now
        matchAlert  Value will be set from Panel or URL.  Set a Value Now
Show Block: MATCH BLOCK (4 Questions)
```

has two required parameters in the Survey Flow: an Embedded Data field that stores the actual chat log (**chatLog** in this example), and **chatName**, which specifies the name of the chat log into which the chat will be saved. First, we set up the actual chat log. The name of this field can be any combination of alphanumeric characters (0–9, A–Z, a–z), dash (-) or underscore (_), up to 32 characters length. In this example we keep the default name of this field (**chatLog**). The value of this field should always be set to *text*. Then, we define the **chatName** parameter. The value of this parameter should be the *name* of the chat log field above, which is *chatLog* in this example (Figure 2).

The CHAT block has six optional parameters in the Survey Flow, which allow researchers to apply time limits and customize the chat design. In this example we keep the default design but implement a time limit of two minutes by setting the **chatDuration** parameter to 120 (seconds). This means that the chat will end after two minutes. By default—if this field is left blank—there is no time limit, and participants can chat for as long as they wish. We also set the **allowExitChat** parameter to *no*, which indicates that participants are not allowed to exit the chat before the time is up (i.e., they must chat for two minutes). By default—if this field is left blank—participants can exit any time.

To learn more about time limits, custom designs, and more advanced chat features (i.e., multiple stages, interrupted chat, private and group chat), visit https://smartriqs.com/chat. Researchers who do not want to include any chat in their studies, should delete the CHAT block from the Survey Flow, along with the Embedded Data fields above the CHAT block.

Figure 2 ■ Sample screenshot of the setup of the CHAT block. The chat log is saved in the first Embedded Data field (**chatLog**). The value of the second field (**chatName**) should always be the case-sensitive name of the chat log defined above. The third green panel contains the optional parameters for the CHAT block. Embedded Data fields should always be placed before (above) the CHAT block in the Survey Flow.


Set Embedded Data:
chatLog = text
Add a New Field
Add Below Move Duplicate Add From Contacts Options Delete

Set Embedded Data:
chatName = chatLog
Add a New Field
Add Below Move Duplicate Add From Contacts Options Delete

Set Embedded Data:
chatInstructions Value will be set from Panel or URL. Set a Value Now
chatWindowWidth Value will be set from Panel or URL. Set a Value Now
chatWindowHeight Value will be set from Panel or URL. Set a Value Now
chatDuration = 120
allowExitChat = no
chatTimeFormat Value will be set from Panel or URL. Set a Value Now
Add a New Field
Add Below Move Duplicate Add From Contacts Options Delete

Show Block: CHAT BLOCK (2 Questions)
Add Below Move Duplicate Delete

Adding instructions and recording responses: Question Blocks

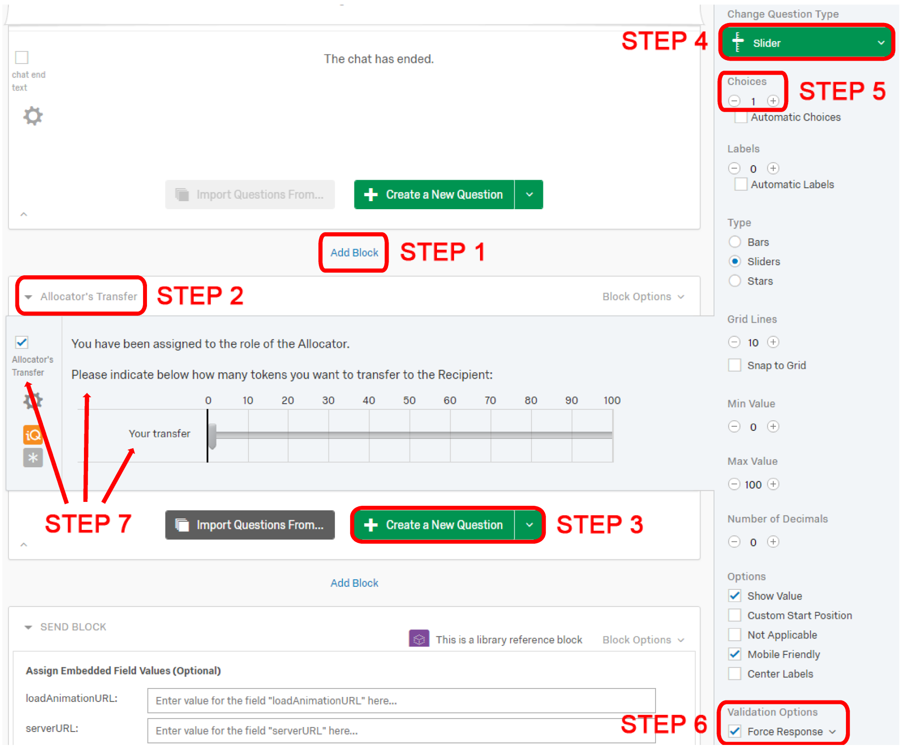
At this point, participants can already chat with each other. If the study does not require any other response to be transmitted, the SEND and GET blocks should be deleted from the Survey Flow. In that case, the survey is ready for testing and data collection. However, if the study requires responses to be transmitted (e.g., the Allocator's transfer), researchers should add new Question Block(s). In this example, we add a block that allows the Allocator to make a decision (see Figure 3):

1. Click on "Save Flow" to close the Survey Flow, then scroll down to the bottom of the CHAT block in the Survey Editor and click on "Add Block." This adds a blank Question Block to the survey.
2. Rename the new block by clicking on its name (e.g., "Allocator's Transfer").
3. Click on "Create a New Question."
4. Click on the green button labeled "Multiple choice" on the right side of the screen (below "Change Question Type") and select "Slider."
5. Change "Choices" to 1.
6. Check "Force Response" to prevent participants from proceeding without making a decision. It is best practice to always force responses in interactive studies. To learn more about forced responses and other types of response validation, visit the corresponding Qualtrics support page.
7. Label the question (e.g., "Allocator's Transfer") and the choice ("Your transfer"), and add some instructions by editing the question text ("You have been assigned to the role of the Allocator. Please indicate below how many tokens you want to transfer to the Recipient").

Next, we repeat the above procedure for the Recipient. In the Dictator Game the Recipient does not make any decisions, she simply receives the Allocator's transfer. However, we still need to add another Question Block, in which we inform the Recipient about the transfer. Add a new Question Block and rename it to "Recipient's Payment." Create a new question, and change the question type to "Descriptive Text." Label the question (e.g., "Recipient's Payment") and add the following text: "The Allocator has decided to transfer you ${e://Field/transfer} tokens" (see Figure 4).

The expression following the $ sign is a Qualtrics Piped Text, which represents a dynamic text that depends on a variable or a previous response. When participants take the survey, they see the actual value of this variable (e.g., 50), instead of the expression. In this example, this Piped

Figure 3 ■ Adding a new block and question in the Qualtrics Survey Editor. This example shows the setup of a question that records the Allocator's transfer in the Dictator Game (slider scale from 0 to 100). To change the default value (0), drag the slider to the desired position.



The chat has ended.

chat end text

Allocator's Transfer

You have been assigned to the role of the Allocator.

Please indicate below how many tokens you want to transfer to the Recipient:

0 10 20 30 40 50 60 70 80 90 100

Your transfer

Assign Embedded Field Values (Optional)

loadAnimationURL: Enter value for the field "loadAnimationURL" here...

serverURL: Enter value for the field "serverURL" here...

Change Question Type: Slider

Choices: 1
Automatic Choices

Labels: 0
Automatic Labels

Type: Bars [✓ Sliders] Stars

Grid Lines: 10
Snap to Grid

Min Value: 0

Max Value: 100

Number of Decimals: 0

Options:
✓ Show Value
Custom Start Position
Not Applicable
✓ Mobile Friendly
Center Labels

Validation Options:
✓ Force Response

Text depends on an Embedded Data field called **transfer**, which has not been defined yet. We will add this field to the Survey Flow when we set up the GET block.

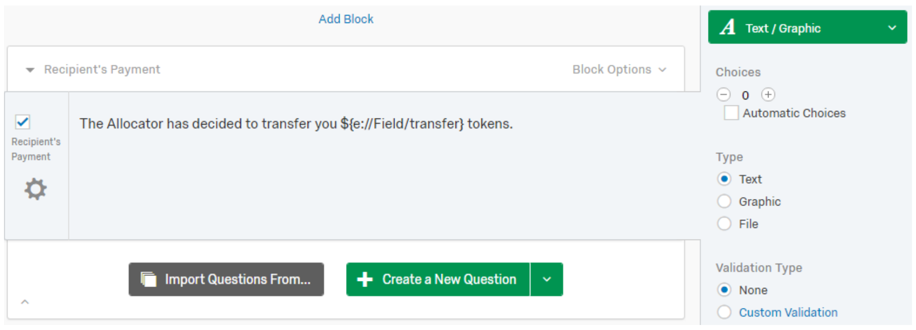
Setting up role-dependent questions and blocks: Branch Logic

In most studies participants are presented different instructions and make different decisions, depending on their roles. In the Dictator Game the Allocator transfers an amount, while the Recipient does not make any decision. This means that we should not display every block to everyone. Instead, we should present the "Allocator's Transfer" block to Allocators only, and the "Recipient's Payment" block to Recipients only. To achieve this, we utilize the Qualtrics feature called Branch logic, which allows for displaying only certain blocks to participants, conditioned on some criteria.

Open the Survey Flow, and scroll down to the section with the two new blocks created in the previous section. Click on "Add Below" on the panel of the CHAT block and select "Branch." Then click on "Add a Condition," and select "Embedded Data" from the drop-down menu. Enter *participantRole* in the first empty field, and enter *Allocator* in the second empty field. Note that these are case-sensitive. Then click on "Move" on the panel of the "Allocator's Transfer" block, and while holding the left mouse button, drag this block under the branch. Repeat this process for the Recipient: Add a new branch below the CHAT block, add a condition, and select "Embedded Data." Enter *participantRole* in the first empty field and *Recipient* in the second empty field. Finally, drag the "Recipient's Payment" block under this new branch (see Figure 5).

The Dictator Game is an example with a single decision and "one-shot" interaction, so we do not have to add more questions. However, one of SMARTRIQS' greatest strengths is the ability to facilitate multi-stage interactions. For example, researchers might want to have participants to make consecutive decisions. For practical examples, see the demo studies with more complex Branch Logic (e.g., Trust Game, Ultimatum Game, or Third-Party Punishment) at https://smartriqs.com/demos. The corresponding QSF templates can be downloaded from the OSF repository.

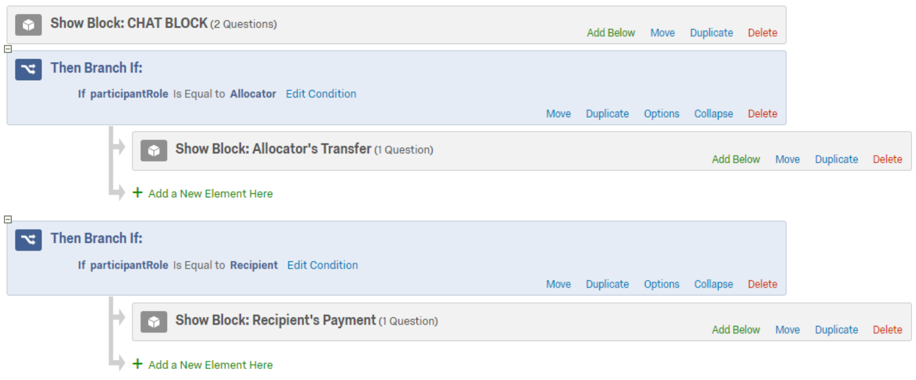
Figure 4 ■ Sample survey question (Recipient's payment)


Add Block

Recipient's Payment — Block Options

The Allocator has decided to transfer you ${e://Field/transfer} tokens.

Figure 5 ■ Using Branch Logic to conditionally display blocks to participants. In this example we use the **participantRole** variable as the condition. Depending on the value of this variable (i.e, the participant's role), we display either only the transfer block or only the payment block.


Show Block: CHAT BLOCK (2 Questions)

Then Branch If:
If participantRole Is Equal to Allocator — Edit Condition

Show Block: Allocator's Transfer (1 Question)

Add a New Element Here

Then Branch If:
If participantRole Is Equal to Recipient — Edit Condition

Show Block: Recipient's Payment (1 Question)

Add a New Element Here

Transmitting responses across surveys: the SEND and GET blocks

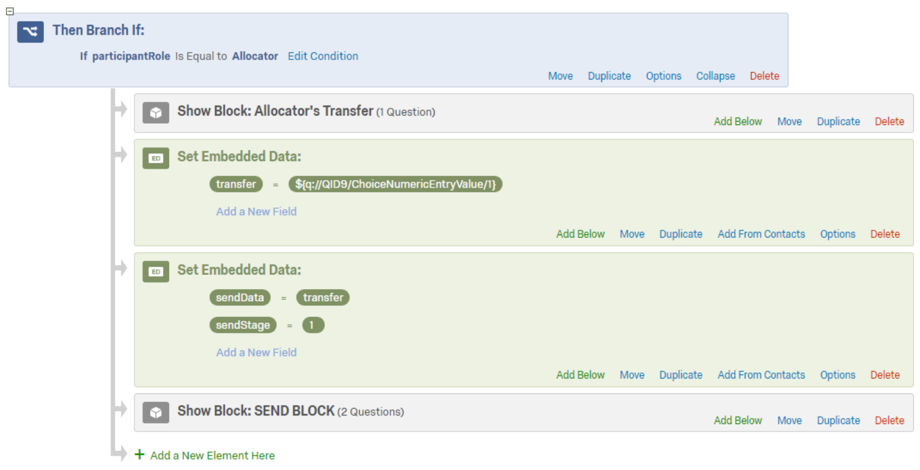
So far we have added all the necessary questions to the survey, however, these are still individual responses, which need to be transmitted across participants. SMARTRIQS uses two blocks to achieve this: the SEND block submits responses to the SMARTRIQS server, while the GET block retrieves responses from the server. In the Dictator Game, we want to send the Allocator's transfer to the server, so that the Recipient can retrieve it.

Sending data: the SEND block

Open the Survey Flow and scroll down to the SEND block. There are two green panels above the SEND block: The top panel has one Embedded Data field (**response**), while the bottom one has two fields (**sendData** and **sendStage**). All three are required parameters, and they should always be set before (above) the SEND block. There are no optional parameters for the SEND block. The green panels and the SEND block should always be placed after (below) the question block that contains the response to be transmitted. In this example, the response to be transmitted is in the "Allocator's Transfer" block, so move the two green panels and the SEND block under the Allocator's branch and below the "Allocator's Transfer" block (Figure 6).

The next step is setting up the values of the three required Embedded Data fields. By default, the name of the first field is **response** but it is best practice to change this to something more informative and specific (max. 32 alphanumeric characters: 0-9, A-Z, a-z, -, _). Not changing the name can not only make data analysis difficult but also lead to errors in the Survey Flow. In this example, we rename this field to **transfer**. The value of this field should always be a Piped Text that refers to a particular response provided by the participant. This response can be of any type: multiple choice, scale, open-ended text, etc. In this example, the Piped Text should refer to the Allocator's de-

Figure 6 ■ Sample screenshot of the setup of the SEND block (Dictator Game).



Then Branch If:

If participantRole Is Equal to Allocator Edit Condition

Move Duplicate Options Collapse Delete

Show Block: Allocator's Transfer (1 Question) Add Below Move Duplicate Delete

Set Embedded Data:

transfer = ${q://QID9/ChoiceNumericEntryValue/1}

Add a New Field

Add Below Move Duplicate Add From Contacts Options Delete

Set Embedded Data:

sendData = transfer

sendStage = 1

Add a New Field

Add Below Move Duplicate Add From Contacts Options Delete

Show Block: SEND BLOCK (2 Questions) Add Below Move Duplicate Delete

+ Add a New Element Here

cision. Click on "Set a Value Now," then click on the blue arrow, and select "Insert Piped Text → Survey Question." The pop-up menu will show all the eligible questions and other variables than can be inserted as Piped Text. Find the question that has the response to be transmitted (in this example: "Allocator's Transfer"). When this question is highlighted, click on the response to be transmitted ("Your Transfer," see Figure 7).

Note that "Your Transfer" is the manually set *label* of the response, not the response itself. It is worth reiterating that the use of intuitive and informative labels is best practice. In this instance, it will make it easier to insert the correct response using Piped Text.

The second field (**sendData**) indicates which previously defined Embedded Data field should be sent to the server. The value of this field should always be the *name of another Embedded Data field*—it should *not* refer directly to a question response. In this example, change the value of **sendData** to *transfer*. This indicates that the SEND block will access the value stored in the **transfer** field, and send that value to the SMARTRIQS server. The third field (**sendStage**) specifies which experimental stage does the selected response correspond to. The value of this field should be a positive integer that is less than or equal to the **numStages** parameter, which was defined before the MATCH block. Since the Dictator Game has only one stage (**numStages** = 1), set **sendStage** = 1.

Retrieving data: the GET block

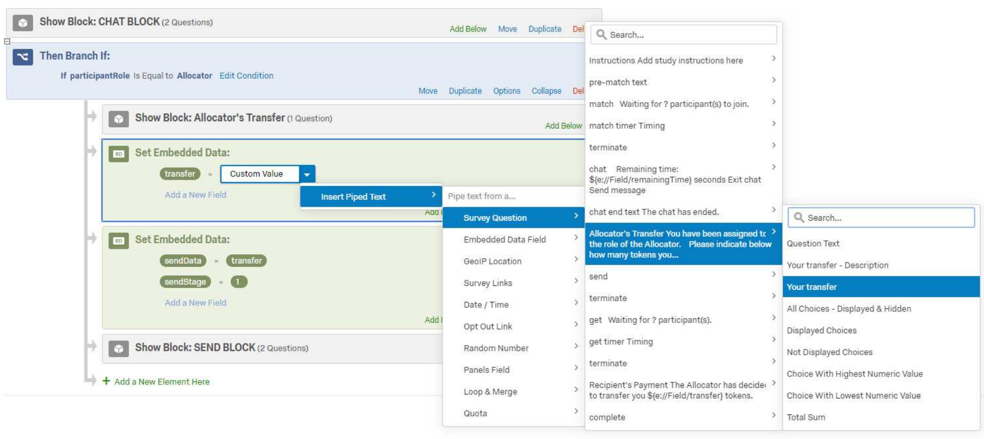
The function of the GET block is to retrieve responses from the SMARTRIQS server. A response cannot be retrieved if it was not previously transmitted to the server via a SEND block. In this example, the GET block retrieves the Allocator's transfer from the server and saves this response to the Recipient's survey. Qualtrics can then display the response to the Recipient. Open the Survey Flow and scroll down to the GET block. There are two green panels above the SEND block: the top one has three required fields of Embedded Data (**getData**, **getStage**, and **saveData**), while the bottom one has nine optional fields. The green panels should always be placed before (above) the GET block. Always place the GET block before (above) the question block(s) in which the retrieved response is displayed. Move the panels and the GET block under the Recipient branch, but *above* the "Recipient's Payment" block (Figure 8).

The **getData** field specifies which response should be retrieved from the server. The value of this field should be *the role of the participant* whose response is retrieved (e.g., *Allocator*). The **getStage** field specifies which stage the response is retrieved from. This should be a positive integer that is less than or equal to the **numStages** parameter. Since the Allocator's response is transmitted in Stage 1, set **getStage** = 1. The **saveData** field specifies the *name of the Embedded Data field* into which the retrieved response is saved. This Embedded Data field should be added manually to the Survey Flow. To do this, click on "Add a New Field," and name the new field, leaving its value blank. Then use the name of this new field as the value for **saveData**. For example, create a new field named **transfer**, and set the value of **saveData** to *transfer*.

The GET block has nine optional parameters. The **defaultData** parameter specifies a default (computer-generated) response, which is applied if the original (human) response cannot be retrieved. This occurs in two cases:

1. If the other "participant" is a bot.

Figure 7 ■ Setting up a SEND block: Inserting a response as Piped Text.


Show Block: CHAT BLOCK (2 Questions)
Add Below Move Duplicate Del
Then Branch If:
If participantRole Is Equal to Allocator Edit Condition
Move Duplicate Options Collapse Del
Show Block: Allocator's Transfer (1 Question)
Add Below
Set Embedded Data:
transfer = Custom Value
Add a New Field
Insert Piped Text
Set Embedded Data:
sendData = transfer
sendStage = 1
Add a New Field
Show Block: SEND BLOCK (2 Questions)
Add a New Element Here
Pipe text from a...
Survey Question
Embedded Data Field
GeoIP Location
Survey Links
Date / Time
Opt Out Link
Random Number
Panels Field
Loop & Merge
Quota
Search...
Instructions Add study instructions here
pre-match text
match Waiting for ? participant(s) to join.
match timer Timing
terminate
chat Remaining time: ${e://Field/remainingTime} seconds Exit chat Send message
chat end text The chat has ended.
Allocator's Transfer You have been assigned to the role of the Allocator. Please indicate below how many tokens you...
send
terminate
get Waiting for ? participant(s).
get timer Timing
terminate
Recipient's Payment The Allocator has decided to transfer you ${e://Field/transfer} tokens.
complete
Search...
Question Text
Your transfer - Description
Your transfer
All Choices - Displayed & Hidden
Displayed Choices
Not Displayed Choices
Choice With Highest Numeric Value
Choice With Lowest Numeric Value
Total Sum

2. If the other participant has "timed out," that is, if he or she has not submitted a response before the maximum waiting time expired (see **maxWaitTime**).

Note that to avoid any deception, participants in SMARTRIQS are always informed in real-time whether they have been matched with other participants or bots. This is a built-in feature that cannot be customized. By default, there are no bots or default responses in SMARTRIQS surveys. To learn more about using bots and default responses, visit https://smartriqs.com/bots.

The **maxWaitTime** parameter specifies the maximum waiting time limit (in seconds). Participants will wait in the GET block until the response is successfully retrieved, or until they reach this time limit. If they reach the time limit before retrieving a response, the survey is either terminated (the default setting), or a default (computer-generated) response is applied (if using bots or using default responses is allowed). If this optional parameter is left blank, participants will wait up to 3 minutes for each response. The **getWaitText** parameter customizes the message displayed to participants while they are waiting in the GET block, while the other six optional parameters allow researchers to do mathematical operations on retrieved responses (for example, to calculate the average or the maximum of retrieved responses). In the Dictator Game example there is no need for these optional settings or mathematical operations, so leave these fields blank. To learn more about optional parameters and operations, visit https://smartriqs.com/sending-retrieving/#getBlock.

Concluding the study: the COMPLETE block

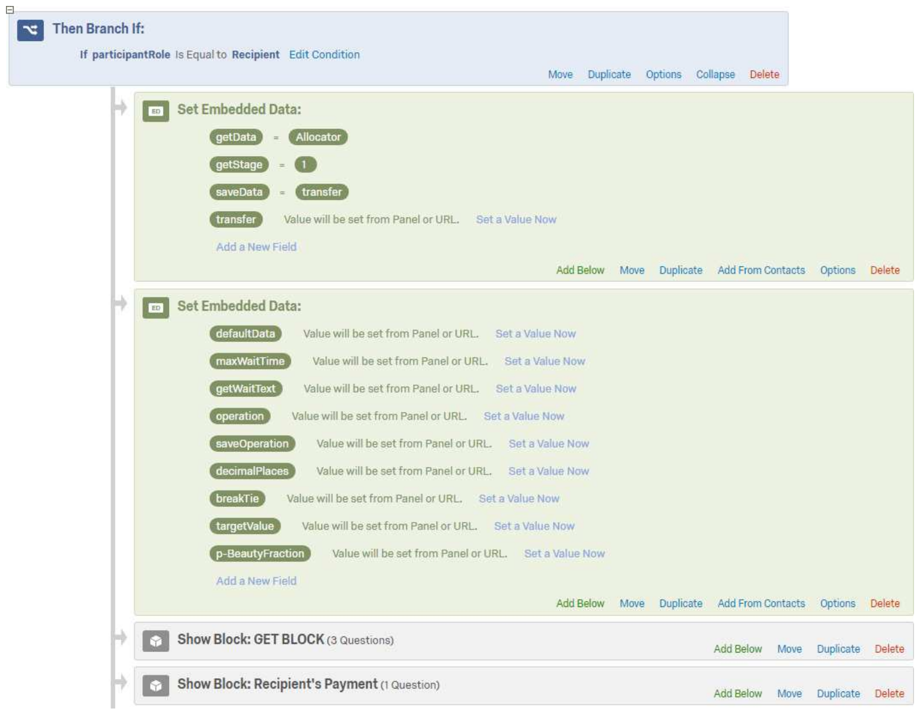
The COMPLETE block concludes the survey and indicates that the participant has completed the study. It also records issues in the **timeOutLog** field (e.g., participant dropouts and response timeouts). If a participant has not experienced any issue during the study, the **timeOutLog** field will take the following value: "OK – no issues." Otherwise, the field will contain a brief description of the issue encountered (e.g., "Allocator timed out in stage 1").

The COMPLETE block does not require any parameters and cannot be customized, so there is no need to define any Embedded Data fields before this block. Do not put any blocks *after* the COMPLETE block, otherwise SMARTRIQS could incorrectly indicate that a participant has completed the study, even if that participant actually dropped out at some point after the COMPLETE block. The only exception is the End of Survey Element which should always be placed after the COMPLETE block. This element is optional: Researchers can include it to display a custom end of survey message or to redirect participants to another page (Figure 9).

Launching studies and monitoring data collection

To launch the study, save and exit the Survey Flow, then click on the green "Publish" button (top right), and click "Publish" again. Qualtrics will generate a public URL address ("Anonymous Survey Link"), which gives access to the survey. If the study has been already published, the URL can be accessed under "Distributions → Anonymous Link." Thoroughly test any survey and ensure that all features work as expected before distributing the study link to participants. To test the Dictator Game, open the link in two tabs (or on two devices), and complete the study in both roles. If everything has been set up properly, the study should conclude without any issues, and the data should be available under the Data & Analysis menu in Qualtrics. Otherwise, an error message will be displayed, describing the issue. To optimize participant experience,

Figure 8 ■ Sample screenshot of the setup of the GET block: required parameters (top green panel) and optional parameters (bottom green panel). These panels and the GET block should always be placed before (above) the block in which the retrieved response is displayed ("Recipient's Payment" in this example).



Then Branch If:
If participantRole Is Equal to Recipient Edit Condition

Set Embedded Data:
getData = Allocator
getStage = 1
saveData = transfer
transfer Value will be set from Panel or URL. Set a Value Now
Add a New Field

Set Embedded Data:
defaultData Value will be set from Panel or URL. Set a Value Now
maxWaitTime Value will be set from Panel or URL. Set a Value Now
getWaitText Value will be set from Panel or URL. Set a Value Now
operation Value will be set from Panel or URL. Set a Value Now
saveOperation Value will be set from Panel or URL. Set a Value Now
decimalPlaces Value will be set from Panel or URL. Set a Value Now
breakTie Value will be set from Panel or URL. Set a Value Now
targetValue Value will be set from Panel or URL. Set a Value Now
p-BeautyFraction Value will be set from Panel or URL. Set a Value Now
Add a New Field

Show Block: GET BLOCK (3 Questions)
Show Block: Recipient's Payment (1 Question)

Move Duplicate Options Collapse Delete
Add Below Move Duplicate Add From Contacts Options Delete
Add Below Move Duplicate Add From Contacts Options Delete
Add Below Move Duplicate Delete
Add Below Move Duplicate Delete

and to minimize the risk of having technical issues, please read the best practices and other useful tips for data collection at https://smartriqs.com/best-practices.

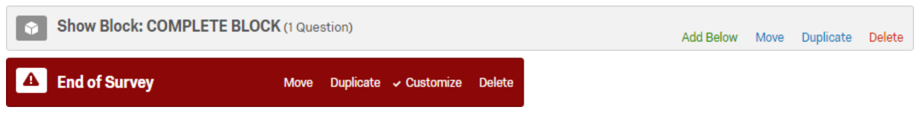
SMARTRIQS has a "progress monitor" website that allows researchers to monitor data collection and participant activity. In addition to displaying the status of each group, the progress monitor also shows the activity and responses of individual participants in real-time (see Figure 10).

To access the progress monitor, go to https://server.smartriqs.com/php/monitor.html and enter the **researcherID** and **studyID**.

Advanced settings and features

SMARTRIQS also allows researchers to run more complex interactive studies, including but not limited to: multiple conditions within studies, group interaction up to 8 people per group, multiple stages, or repeated interaction between participants. This section briefly introduces some of the most important advanced settings of SMARTRIQS. More information, along with practical examples can be found at https://smartriqs.com/getting-started.

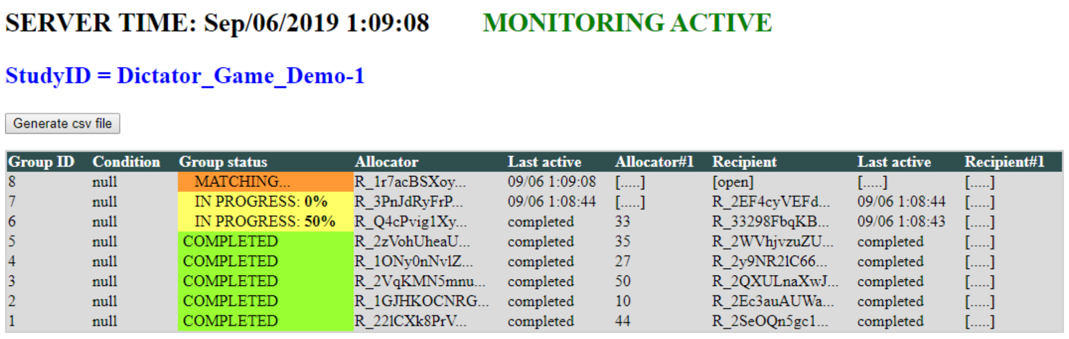
Figure 9 ■ The COMPLETE block and an optional End of Survey Element. The COMPLETE block should always be the last block in SMARTRIQS surveys, unless, if there is a custom End of Survey Element. In that case (as in this example), the End of Survey Element should be placed after the COMPLETE block.



Show Block: COMPLETE BLOCK (1 Question)
End of Survey

Add Below Move Duplicate Delete
Move Duplicate ✓ Customize Delete

Figure 10 ■ Sample screenshot of the SMARTRIQS progress monitor. Each row represents a group. Column 1 shows the group ID, column 2 shows the condition (if set), and column 3 shows the group status. Columns 4–6 show the Allocator's ID, the Allocator's time of last activity, and the Allocator's stage 1 decision (transfer to the Recipient). Columns 7–8 show the Recipient's ID and the Recipient's last activity. Column 9 is blank since Recipients do not submit any decision in the Dictator Game.


SERVER TIME: Sep/06/2019 1:09:08 **MONITORING ACTIVE**

StudyID = Dictator_Game_Demo-1

Group ID	Condition	Group status	Allocator	Last active	Allocator#1	Recipient	Last active	Recipient#1
8	null	MATCHING...	R_1r7acBSXoy...	09/06 1:09:08	[.....]	[open]	[.....]	[.....]
7	null	IN PROGRESS: **0%**	R_3PnJdRyFrP...	09/06 1:08:44	[.....]	R_2EF4cyVEFd...	09/06 1:08:44	[.....]
6	null	IN PROGRESS: **50%**	R_Q4cPvig1Xy...	completed	33	R_33298FbqKB...	09/06 1:08:43	[.....]
5	null	COMPLETED	R_2zVohUheaU...	completed	35	R_2WVhjvzuZU...	completed	[.....]
4	null	COMPLETED	R_1ONy0nNvlZ...	completed	27	R_2y9NR21C66...	completed	[.....]
3	null	COMPLETED	R_2VqKMN5mnu...	completed	50	R_2QXULnaXwJ...	completed	[.....]
2	null	COMPLETED	R_1GJHKOCNRG...	completed	10	R_2Ec3auAUWa...	completed	[.....]
1	null	COMPLETED	R_221CXk8PrV...	completed	44	R_2SeOQn5gc1...	completed	[.....]

Multiple conditions

Researchers can set up multiple conditions within a survey by filling in the optional parameters **conditions** and **participantCondition** before the MATCH block. SMARTRIQS will then assign groups to conditions, based on these specifications, see https://smartriqs.com/randomization. Once participants are assigned to conditions, researchers can decide what should happen in each condition by using Branch Logic in the Survey Flow. For example, imagine a modified Dictator Game with three possible levels: *low* (10 tokens), *medium* (100 tokens), and *high* (1000 tokens), indicating that the Allocator would either allocate 10, 100, or 1000 tokens, depending on the condition. To achieve this, set **conditions** = *low,medium,high* (note that there is no space between the commas and the name of the conditions) and **participantCondition** = *random*. Then, use either Branch Logic in the Survey Flow or Display Logic in the Survey Editor to determine which version of the Dictator Game is displayed to which participant. To learn more about how to implement studies such as the above example, see the "Dictator Game, 3 conditions" survey at https://smartriqs.com/demos and download the corresponding template from the OSF repository.

Larger groups (3–8 participants per group)

SMARTRIQS supports group interactions up to 8 participants per group. Researchers can assign: (a) the same role with an identical task to everyone (e.g., auction, collective decision), (b) unique roles with different tasks to each participant (e.g., negotiation), or (c) any combination of the above. Note that the *names* of the roles should always be unique to each participant within a group, even if their tasks are identical. To increase the group size, modify the value of the **groupSize** field to the desired number, and then add this many roles to the **roles** field, separating them by commas. For example, in a study with groups of four, where participants are assigned to the roles of *Blue*, *Red*, *Green* and *Yellow*, set **groupSize** = 4, and **roles** = *Blue,Red,Green,Yellow* (note that there is no space between role titles).

When having groups of 3–8 participants, it is also possible to set up private and group chats. While private chats are between selected participants only (excluding at least one participant), group chats include everyone in the group. Researchers can customize which participants, with whom, when, and for how long, chat in a study. As with dyadic chat, it is possible to interrupt private and group chats with unrelated tasks. In later stages, participants can also join those private chats from which they were excluded from before. To learn more about how to set up private and group chats, visit https://smartriqs.com/chat. For demos, see the "Communication" section at https://smartriqs.com/demos. The "Group Interaction" section showcases surveys with larger groups. The corresponding templates can be downloaded from the OSF repository.

Turn-taking and multiple stages

In many studies, participants take turns or have to respond to their partner's choices. For example, the Ultimatum Game (Güth, Schmittberger, & Schwarze, 1982) has two consecutive stages. The first stage is identical to the Dictator Game: The Allocator decides how to split a sum of money between herself and the Recipient. In the second stage, however, the Recipient can decide whether to accept or to reject the Allocator's offer. Rejecting the offer leaves both empty-handed.

To implement the above, duplicate the Dictator Game survey created in the previous section, then rename the **studyID**. Note that each study should have a unique **studyID**. Change the **numStages** to 2, indicating that there are two responses to be transmitted. Then, add a new question block in which the Recipient reacts to the offer, and add a new SEND block in the Survey Flow below this new question block. Finally, pipe in the Recipient's response into a new embedded data field **reaction**, then set **sendData** = *reaction* and **sendStage** = 2 (see Figure A1 in the Appendix).

Next, go to the Allocator's branch, and add a new Embedded Data field named **reaction** under the SEND block. Also set **getData** = *Recipient*, **getStage** = 2, and **saveData** = *reaction*, to indicate that the Recipient's reaction should be retrieved and saved into the field **reaction**. Then insert a new GET block under these fields. Finally, use either Branch Logic in the Survey Flow or Display Logic in the Survey Editor to display the final payoffs, conditioned on whether the offer was accepted or rejected (see Figure A2 in the Appendix).

The "Trust Game" and "Third-Party Punishment" demos at https://smartriqs.com/demos also rely on sequential interaction and multiple stages. These demos, along with the Ultimatum Game demo, were designed to help researchers learn how to implement sequential interactions and multiple stages. Corresponding survey templates are also available at the OSF repository.

Simultaneous responses and waiting rooms

In some cases researchers might want to ensure that participants start certain stages of an experiment at the exact same time (e.g., group chat, effort task). Also, if participants have to read long instructions before a task, it is likely that some of them will spend considerably more time reading instructions than others, which introduces asynchrony between participants. To ensure that participants start stages simultaneously, researchers can set up "waiting rooms" in studies. When participants enter a waiting room, they cannot proceed to the next stage before everyone else in their group joins them in the waiting room.

To set up a waiting room, insert a SEND block and a GET block before the task that participants should start simultaneously. Set the following parameters before the SEND block: **participantStatus** = *ready*, **sendData** = *participantStatus*, and **sendStage** = 1 (this must be a number that is not used in any other SEND block). Importantly, each waiting room counts as a separate stage—separate from other decisions—which means that researchers should not use its stage number when sending or retrieving other responses. For example, if there is a waiting room in the Dictator Game (before the Allocator makes a decision) then the waiting room is Stage 1 and the Allocator's transfer is Stage 2. Or, if there is a waiting room in the Ultimatum Game (before the Allocator's initial offer) then the waiting room is Stage 1, the Allocator's offer is Stage 2, and the Recipient's response is Stage 3.

Next, set the following parameters before the GET block: **getData** = *Allocator,Recipient* (the roles in the group), **getStage** = 1 (this number should match the one defined above), and **saveData** = *null,null*.[1] Finally, set a custom message that participants will see in the waiting room. This will reduce participant concerns related to their status in the study. Setting **getWaitText** = *The task will start soon. Please wait for the other participant.*, for example, will display this message while participants wait to start (see Figure A3 in the Appendix).

Learn more about setting up waiting rooms by reviewing the "Effort Competition with Waiting Room" demo at https://smartriqs.com/demos and downloading the corresponding survey template from the OSF repository.

Transmitting sensitive and personal data: custom private servers

The SMARTRIQS Data Policy prohibits researchers from submitting any personal or sensitive data to the default SMARTRIQS server. Since the server is hosted at Amazon Web Services—which constitutes as a third-party—participant's confidentiality cannot be guaranteed. Researchers are allowed to submit any *anonymous data* (e.g., participant IDs, decisions, roles, chat logs, or open-ended text), as long as these do not contain any personal identifiers or personal addresses. In addition, some funding agencies might forbid research involving non-private servers due to potential data privacy issues. Researchers who wish to use SMARTRIQS for submitting personal data or other identifiers, should set up their own server on a secure, private web-server. Finally, by using a custom server researchers can also freely modify the server-side scripts, in case they wish to modify the built-in settings of SMARTRIQS (e.g., maximum group size), or add new features to it (e.g., video chat). A step-by-step guide to setting up a custom SMARTRIQS server is available at https://smartriqs.com/custom-server.

[1] Note that here the retrieved "responses" are not actual responses that we want to save. They are simply status indicators that determine whether the participant can proceed. As such, in this case we do not have to refer to any other Embedded Data when setting up the **saveData** parameter. Use *null* instead, and separate them by commas (no space between).

Author's note

I thank Ankita Sastry for her help in developing SMARTRIQS and Peggy He, Jinny Hwang, Denise Lin, and Sommer Schneller for their assistance in testing SMARTRIQS. I also thank Denis Cousineau, Russell Golman, Nik Gurney, Einav Hart, Mark Hurlstone, Daisung Jang, Melis Kirgil, Jonathan Lee, Stephanie Permut, Eric VanEpps, Chao Wang, Simon van der Zandt, and an anonymous reviewer for their valuable feedback and suggestions.

References

Arechar, A. A., Gächter, S., & Molleman, L. (2018). Conducting interactive experiments online. *Experimental Economics*, *21*(1), 99–131. doi:10.1007/s10683-017-9527-2

Balietti, S. (2017). nodeGame: Real-time, synchronous, online experiments in the browser. *Behavior Research Methods*, *49*(5), 1696–1715. doi:10.3758/s13428-016-0824-z

Berinsky, A. J., Huber, G. A., & Lenz, G. S. (2012). Evaluating Online Labor Markets for Experimental Research: Amazon.com's Mechanical Turk. *Political Analysis*, *20*(3), 351–368. doi:10.1093/pan/mpr057

Bohannon, J. (2016). Mechanical Turk upends social sciences. *Science*, *352*(6291), 1263–1264. doi:10.1126/science.352.6291.1263

Casari, M., & Cason, T. N. (2009). The strategy method lowers measured trustworthy behavior. *Economics Letters*, *103*(3), 157–159. doi:10.1016/j.econlet.2009.03.012

Chandler, J., Rosenzweig, C., Moss, A. J., Robinson, J., & Litman, L. (2019). Online panels in social science research: Expanding sampling methods beyond Mechanical Turk. *Behavior Research Methods*, *51*(5), 2022–2038. doi:10.3758/s13428-019-01273-7

Chen, D. L., Schonger, M., & Wickens, C. (2016). oTree—An open-source platform for laboratory, online, and field experiments. *Journal of Behavioral and Experimental Finance*, *9*, 88–97. doi:10.1016/j.jbef.2015.12.001

Collaboration*, O. S. (2015). Estimating the reproducibility of psychological science. *Science*, *349*(6251), aac4716–aac4716. doi:10.1126/science.aac4716

Fischbacher, U. (2007). z-Tree: Zurich toolbox for ready-made economic experiments. *Experimental Economics*, *10*(2), 171–178. doi:10.1007/s10683-006-9159-4

Forsythe, R., Horowitz, J. L., Savin, N., & Sefton, M. (1994). Fairness in Simple Bargaining Experiments. *Games and Economic Behavior*, *6*(3), 347–369. doi:10.1006/game.1994.1021

Giamattei, M., Molleman, L., Seyed Yahosseini, K., & Gachter, S. (2019). LIONESS Lab – A Free Web-Based Platform for Conducting Interactive Experiments Online. *SSRN Electronic Journal*. doi:10.2139/ssrn.3329384

Gosling, S. D., & Mason, W. (2015). Internet Research in Psychology. *Annual Review of Psychology*, *66*(1), 877–902. doi:10.1146/annurev-psych-010814-015321

Güth, W., Schmittberger, R., & Schwarze, B. (1982). An experimental analysis of ultimatum bargaining. *Journal of Economic Behavior & Organization*, *3*(4), 367–388. doi:10.1016/0167-2681(82)90011-7

Hawkins, R. X. D. (2015). Conducting real-time multiplayer experiments on the web. *Behavior Research Methods*, *47*(4), 966–976. doi:10.3758/s13428-014-0515-6

Hendriks, A. (2012). *SoPHIE - Software Platform for Human Interaction Experiments*, University of Osnabrueck.

Henninger, F., Kieslich, P. J., & Hilbig, B. E. (2017). Psynteract: A flexible, cross-platform, open framework for interactive experiments. *Behavior Research Methods*, *49*(5), 1605–1614. doi:10.3758/s13428-016-0801-6

Henrich, J., Heine, S. J., & Norenzayan, A. (2010). The weirdest people in the world? *Behavioral and Brain Sciences*, *33*(2-3), 61–83. doi:10.1017/S0140525X0999152X

Hertwig, R., & Ortmann, A. (2008). Deception in Experiments: Revisiting the Arguments in Its Defense. *Ethics & Behavior*, *18*(1), 59–92. doi:10.1080/10508420701712990

Mason, W., & Suri, S. (2012). Conducting behavioral research on Amazon's Mechanical Turk. *Behavior Research Methods*, *44*(1), 1–23. doi:10.3758/s13428-011-0124-6

Mathur, M. B., & Reichling, D. B. (2019). Open-source software for mouse-tracking in Qualtrics to measure category competition. *Behavior Research Methods*. doi:10.3758/s13428-019-01258-6

Molnar, A. (2019). SMARTRIQS: A Simple Method Allowing Real-Time Respondent Interaction in Qualtrics Surveys. *Journal of Behavioral and Experimental Finance*, *22*, 161–169. doi:10.1016/j.jbef.2019.03.005

Paolacci, G., & Chandler, J. (2014). Inside the Turk. *Current Directions in Psychological Science*, *23*(3), 184–188. doi:10.1177/0963721414531598

Peer, E., Brandimarte, L., Samat, S., & Acquisti, A. (2017). Beyond the Turk: Alternative platforms for crowdsourcing behavioral research. *Journal of Experimental Social Psychology*, *70*(January), 153–163. doi:10.1016/j.jesp.2017.01.006

Permut, S., Fisher, M., & Oppenheimer, D. M. (2019). TaskMaster: A Tool for Determining When Subjects Are on Task. *Advances in Methods and Practices in Psychological Science*, *2*(2), 188–196. doi:10.1177/2515245919838479

Pettit, J., Friedman, D., Kephart, C., & Oprea, R. (2014). Software for continuous game experiments. *Experimental Economics*, *17*(4), 631–648. doi:10.1007/s10683-013-9387-3

Qualtrics. (2020). Provo, Utah, USA: Qualtrics. Retrieved from https://www.qualtrics.com

SurveyMonkey. (2020). San Mateo, California, USA: SurveyMonkey Inc. Retrieved from www.surveymonkey.com

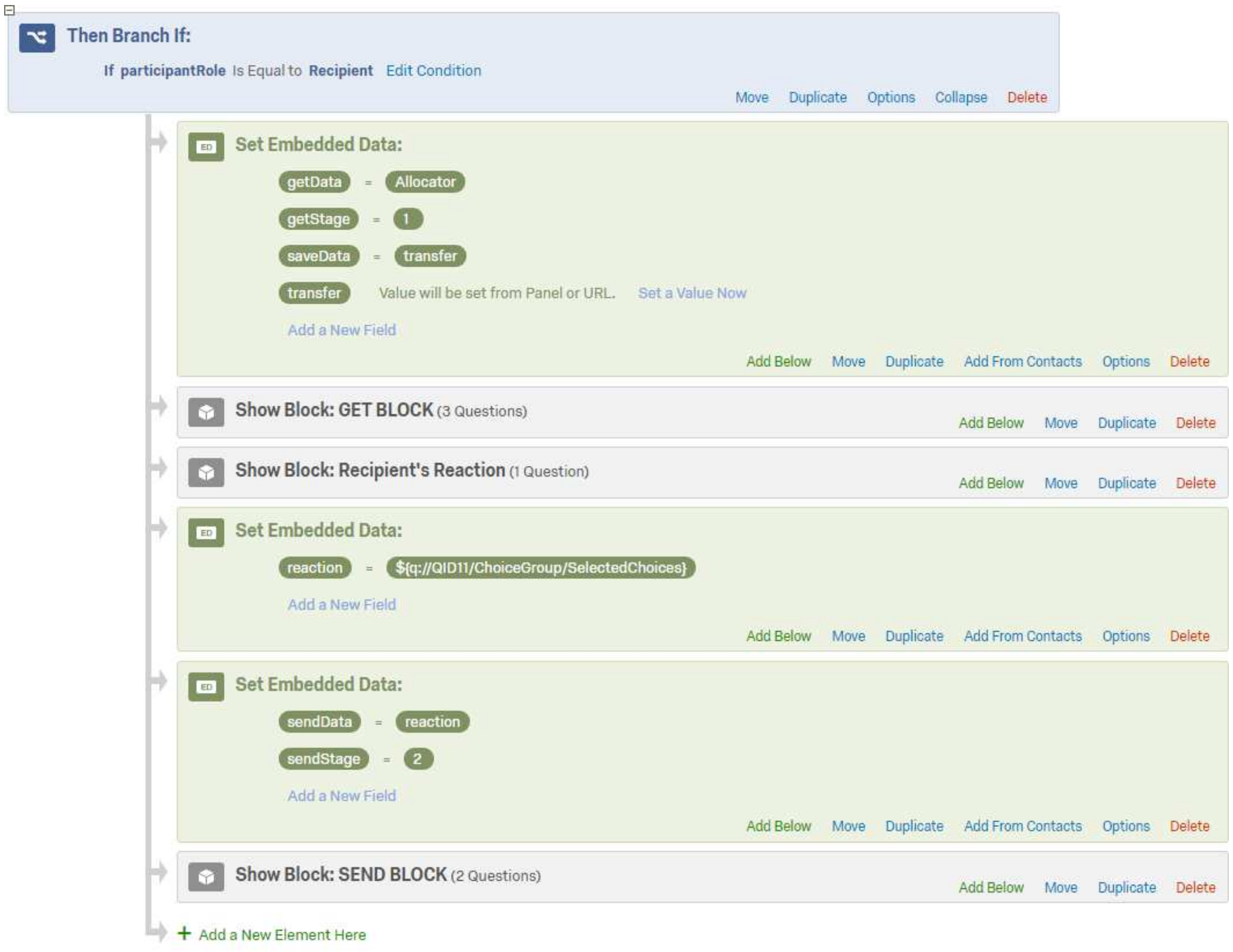
Zhou, H., & Fishbach, A. (2016). The pitfall of experimenting on the web: How unattended selective attrition leads to surprising (yet false) research conclusions. *Journal of Personality and Social Psychology*, *111*(4), 493–504. doi:10.1037/pspa0000056

Appendix

This appendix provides useful links and additional screen captures.

Useful Links

1. Live demos: https://smartriqs.com/demo
2. SMARTRIQS registration form: LINK
3. Survey templates (QSF files): https://osf.io/cgejr
4. Data collection progress monitor: https://server.smartriqs.com/php/monitor.html
5. Data Policy & Data Submission Policy Agreement: https://smartriqs.com/data-policy
6. Source code (JavaScript and PHP): https://github.com/andras-molnar/smartriqs


Then Branch If:
If participantRole Is Equal to Recipient Edit Condition

Set Embedded Data:
getData = Allocator
getStage = 1
saveData = transfer
transfer Value will be set from Panel or URL. Set a Value Now

Show Block: GET BLOCK (3 Questions)

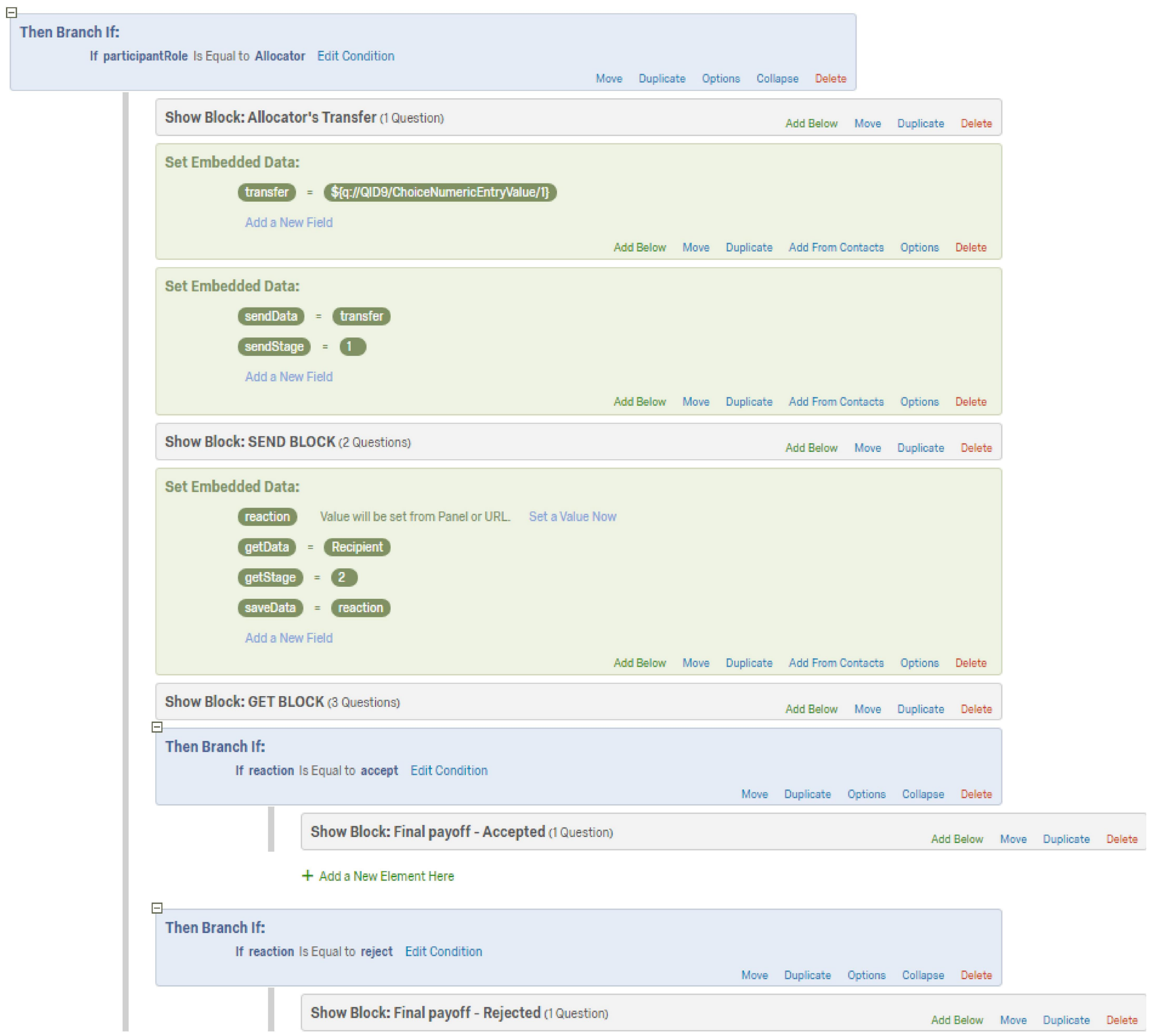
Show Block: Recipient's Reaction (1 Question)

Set Embedded Data:
reaction = ${q://QID11/ChoiceGroup/SelectedChoices}

Set Embedded Data:
sendData = reaction
sendStage = 2

Show Block: SEND BLOCK (2 Questions)

Figure A1 ■ Sample setup of the Recipient's branch in an Ultimatum Game. First, the Allocator's offer is retrieved via a GET block, then the Recipient's reaction is transmitted via a SEND block. The new panels below the "Recipient's Reaction" block were added by clicking on the "Add a New Element Here" button, and then either selecting "Embedded Data" or "Block."


Then Branch If:
If participantRole Is Equal to Allocator Edit Condition
Move Duplicate Options Collapse Delete

Show Block: Allocator's Transfer (1 Question) Add Below Move Duplicate Delete

Set Embedded Data:
transfer = ${q://QID9/ChoiceNumericEntryValue/1}
Add a New Field
Add Below Move Duplicate Add From Contacts Options Delete

Set Embedded Data:
sendData = transfer
sendStage = 1
Add a New Field
Add Below Move Duplicate Add From Contacts Options Delete

Show Block: SEND BLOCK (2 Questions) Add Below Move Duplicate Delete

Set Embedded Data:
reaction Value will be set from Panel or URL. Set a Value Now
getData = Recipient
getStage = 2
saveData = reaction
Add a New Field
Add Below Move Duplicate Add From Contacts Options Delete

Show Block: GET BLOCK (3 Questions) Add Below Move Duplicate Delete

Then Branch If:
If reaction Is Equal to accept Edit Condition
Move Duplicate Options Collapse Delete

Show Block: Final payoff - Accepted (1 Question) Add Below Move Duplicate Delete

+ Add a New Element Here

Then Branch If:
If reaction Is Equal to reject Edit Condition
Move Duplicate Options Collapse Delete

Show Block: Final payoff - Rejected (1 Question) Add Below Move Duplicate Delete

Figure A2 ■ Sample setup of the Allocator's branch in an Ultimatum Game. First, the Allocator's offer is transmitted via a SEND block, then the Recipient's reaction is retrieved via a GET block. The branches below the GET block use the retrieved response to display either the "Accepted" or the "Rejected" blocks.

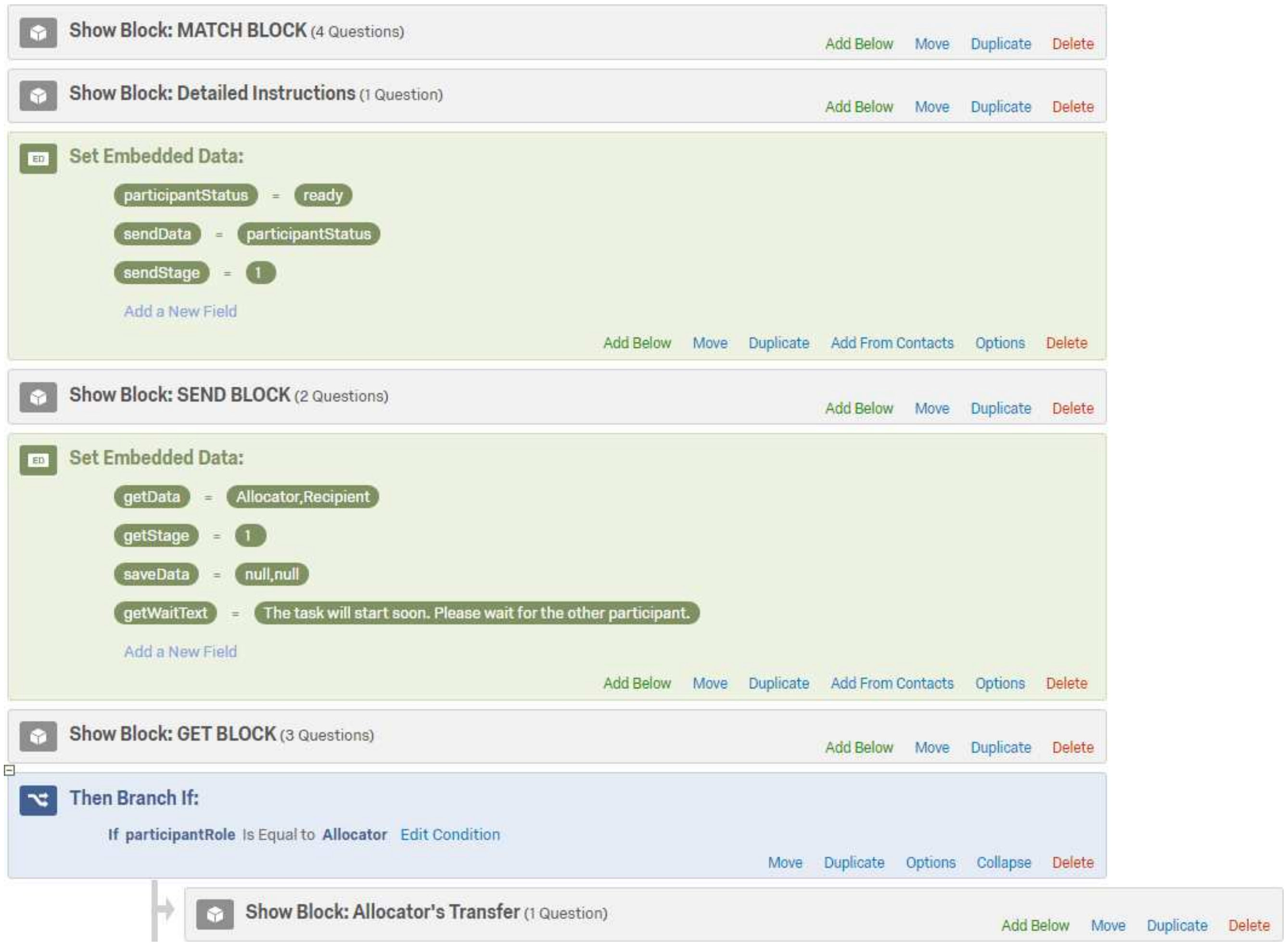

Show Block: MATCH BLOCK (4 Questions)
Show Block: Detailed Instructions (1 Question)
Set Embedded Data:
participantStatus = ready
sendData = participantStatus
sendStage = 1
Show Block: SEND BLOCK (2 Questions)
Set Embedded Data:
getData = Allocator,Recipient
getStage = 1
saveData = null,null
getWaitText = The task will start soon. Please wait for the other participant.
Show Block: GET BLOCK (3 Questions)
Then Branch If:
If participantRole Is Equal to Allocator
Show Block: Allocator's Transfer (1 Question)

Figure A3 ■ Sample setup of a waiting room. Here the waiting room is inserted after the instructions, but before the Allocator's branch. Both participants have to read the instructions first and proceed to the waiting room. Note that a waiting room counts as a separate stage, so in this example the Allocator's transfer would be already Stage 2.

Citation

Molnar, A. (2020). How to implement real-time interaction between participants in online surveys: A practical guide to SMARTRIQS. *The Quantitative Methods for Psychology*, *16*(4), 334–354. doi:10.20982/tqmp.16.4.p334

Received: 15/02/2020 ∼ Accepted: 21/04/2020