Online Implementation of an Event History Calendar with formr: A Tutorial

Larissa L. Wieczorek and Cyril S. Tata
University of Goettingen

Lars Penke and Tanja M. Gerlach
University of Goettingen and Leibniz ScienceCampus Primate Cognition
Author Note

Larissa L. Wieczorek, Department of Psychology, University of Goettingen, Germany; Cyril S. Tata, Department of Psychology, University of Goettingen, Germany; Lars Penke, Department of Psychology, University of Goettingen, Germany and Leibniz ScienceCampus Primate Cognition; Tanja M. Gerlach, Department of Psychology, University of Goettingen, Germany and Leibniz ScienceCampus Primate Cognition.

Larissa L. Wieczorek and Cyril S. Tata wrote the first draft of the manuscript and created the online material for the tutorial. Tanja M. Gerlach and Lars Penke came up with the idea for the tutorial, gave conceptual input and advice regarding its conceptualization and provided critical revisions of the manuscript.

We thank Julie C. Driebe and Ruben C. Arslan for their support when setting up the first study employing an online event history calendar in formr. We thank Salome Johannigmann and Kim Gloystein for testing the tutorial and providing feedback for its improvement.

Correspondence concerning this article should be addressed to Larissa L. Wieczorek, University of Goettingen, Institute of Psychology, Gosslerstraße 14, 37073 Goettingen. Email: larissa.lucia@web.de. Further Contacts: Cyril S. Tata, Lars Penke, and Tanja M. Gerlach, University of Goettingen, Institute of Psychology, Gosslerstraße 14, 37073 Goettingen. Emails: ctata@uni-goettingen.de, lpenke@uni-goettingen.de, gerlach@uni-goettingen.de.
Abstract

Event history calendars (EHCs) are popular tools for retrospective data collection. Originally conceptualized as face-to-face interviews, EHCs contain various questions about the respondents’ autobiography in order to use their experiences as cues to facilitate remembering. For relationship researchers, EHCs are particularly valuable when trying to reconstruct the relational past of individuals. However, while many studies are conducted online nowadays, no online adaptation of the EHC is available yet. In this tutorial, we provide detailed instructions on how to implement an online EHC for the reconstruction of romantic relationship histories within the open source framework formr. We exemplify on ways to customize the online EHC and provide a template for researchers to adapt the tool for their own purposes.

Keywords: Event history calendar; relationship research; retrospection; online assessment; open source
Introduction

A common goal in relationship science is to investigate events and patterns of relationship-related behavior over a long period of time. Examples of such research topics are interaction or attachment patterns across and within different types of relationships (e.g. Neyer & Asendorpf, 2001; Robins, Caspi, & Moffitt, 2002; Roisman, Madsen, Hennighausen, Sroufe, & Collins, 2001; Sprecher & Felmlee, 1997), dating and sexual behavior (e.g. Asendorpf, Penke, & Back, 2011; Brown & Sinclair, 1999; Kinsey, Pomeroy, & Martin, 2003; Penke & Asendorpf, 2008; Turner et al., 1998), or the choice of long-term partners (e.g. Campbell, Chin, & Stanton, 2016; Eastwick, Harden, Shukusky, Morgan, & Joel, 2017; Gerlach, Arslan, Schultze, Reinhard, & Penke, 2017).

Arguably, the best way to assess consecutive data points that could speak to the topics above would be longitudinal study designs (Bordens & Abbott, 2010; Schaie, 2000). However, the time, effort, and costs to conduct such follow-up studies can be enormous (Bordens & Abbott, 2010; Schaie, 1983). Moreover, there are situations where it is impossible to perform a longitudinal study and follow the individuals over time, because the (start) time of interest has already passed. In such cases, researchers have to rely on retrospective data collection that is based on the participants’ autobiographical memories. Unfortunately, human memory performance is known to decline with time (Ebbinghaus, 2013; Rubin & Wenzel, 1996) and to be prone to a bunch of biases or even the possibility of false memories (e.g. Belli, 1998; Hyman & Pentland, 1996; Loftus, 1979; Ross, 1989; Schwarz, 2007). For example, a common finding in sexuality research is that men tend to systematically overestimate the number of their sexual partners when answering to regular self-reports (Brown & Sinclair, 1999; Mitchell et al., 2018). In contrast, when it comes to negative life events such as depressive episodes or adverse childhood experiences, people
(both men and women) tend to underestimate the frequency of events in retrospect (Hardt &
Rutter, 2004; Patten, 2009).

Fortunately, knowledge about the hierarchical structure of autobiographical memory
has increased and techniques for the facilitation of memory retrieval have been developed
(Belli, 1998; Berntsen & Rubin, 2004; Freedman, Thornton, Camburn, Alwin, & Young-
DeMarco, 1988). One very useful tool for retrospective data collection is the event history
calendar (EHC), sometimes also called life history calendar (see Freedman et al., 1988;
Martyn & Belli, 2002). Today, there are a large number of studies using EHCs in a face-to-
face interview (for a review, see Glasner & van der Vaart, 2009). For large-scale studies and
nowadays common use of online studies (see Bohannon, 2016; Mathur & Evans, 2005; Selm
& Jankowski, 2006), however, transferring the method to online environments would be
highly desirable. In this tutorial, after pointing out the advantages of EHC approaches for
relationship science, we aim to provide detailed instructions for setting up an online EHC
within the open source survey framework formr (Arslan & Tata, 2017).

Event History Calendars and Exemplary Research Questions

The EHC is a tool that was originally conceptualized as a live face-to-face interview
and helps respondents to recall their past by making use of the respondents’ own past
experiences as cues for remembering (Belli, Smith, Andreski, & Agrawal, 2007; Martyn &
Belli, 2002). During an EHC interview, respondents are encouraged to consider various
personally relevant events, which get documented in a calendar grid. The events collected via
the EHC serve as cues that facilitate autobiographical memory retrieval (Belli, 1998; Belli et
al., 2007; Freedman et al., 1988; Martyn & Belli, 2002). In accordance with the structure of
autobiographical memory, the EHC grid and interview are organized thematically by domains
and temporally by chronological ordering (Martyn & Belli, 2002). These domains are defined
by certain types of events (e.g. clinical treatments) or people (e.g. dating partners). In the
beginning of an EHC, it makes sense to start with the domain that will be easiest to recall and least threatening for the participant. Common first domains include extended periodical events such as residences, educational stages, or occupations. In addition, significant political or private events, such as the election of a new government or a job promotion, can be helpful in reconstructing the past (Belli, 1998; Martyn & Belli, 2002). Figure 1 shows the grid of an example of an EHC constructed for a study investigating dating behavior.

<table>
<thead>
<tr>
<th>Half-year</th>
<th>99-H</th>
<th>99-II</th>
<th>00-I</th>
<th>00-II</th>
<th>01-I</th>
<th>01-II</th>
<th>02-I</th>
<th>02-II</th>
<th>03-I</th>
<th>03-II</th>
<th>04-I</th>
<th>04-II</th>
</tr>
</thead>
<tbody>
<tr>
<td>Residence</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Living arrangement</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>School (type and grade)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Employment</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Important life events</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Long-term partners</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Relationship type (e.g. monogamous)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Short-term partners</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Figure 1. Example of an event history calendar grid that is used during EHC interviews.

Past studies provided evidence for the capability of EHCs to improve scope and quality of data, especially regarding data as delicate as sexual behavior (e.g. Luke, Clark, & Zulu, 2011; Martyn, Reifsnider, & Murray, 2006). Now let us think about some research questions relevant to relationship science that could be answered with a custom-tailored EHC. When using an EHC to collect personally relevant information about different kind of relationships, including friends, family, co-workers, or romantic partners, we could ask: Does relationship quality and attachment differ across these relationships? Where and how did participants meet their later friends or romantic partners? How long did these relationships last and who ended them? An EHC also collecting detailed information about living
arrangements could track these over the course of the participants’ lives and document if there are other life circumstances that coincide with specific living arrangements: How do stages of education relate to specific living arrangements? And how about professional transitions and income situations? Similar approaches are conceivable for research on sexual behavior: Does moving to bigger cities go along with having more casual relationships or one-night stands? What circumstances accompany the experience of a fulfilled sexual life? Due to its flexibility, a custom-tailored EHC may be instrumental in collecting the data needed to retrospectively answer these kinds of questions.

The Advantage of an Open Source Online Implementation

As outlined above, EHCs were originally conceptualized as face-to-face interviews. However, a considerable amount of nowadays relationship research is based on the use of online surveys (Bohannon, 2016; Mathur & Evans, 2005; Selm & Jankowski, 2006). Online surveys are quite popular, because they possess a number of favorable qualities (Kraut et al., 2004; Riva, Teruzzi, & Anolli, 2003; Selm & Jankowski, 2006; Sue & Ritter, 2011; Wright, 2005): They allow access to large and potentially more diverse populations; sometimes, even rare populations can be reached. Moreover, online surveys are usually less expensive than laboratory or field studies, enable fast data collection, potentially save time and work effort of the researcher, and may facilitate sharing of sensitive information by providing a higher degree of anonymity for the participants. To summarize, there are several good reasons why researchers decide to run studies online.

To our knowledge, there are no online implementations of EHC studies yet. It is noteworthy to mention that computer-assisted EHCs already exist (Belli et al., 2007; Brüderl, Castiglioni, Ludwig, Pforr, & Schmiedeberg, 2017). However, these adaptations still require an interviewer, instead of enabling participants to complete the calendar on their own. Moreover, they are based on programming languages that are not easy to comprehend for
everyone and thus difficult to customize\textsuperscript{1}. In order to combine the methodological benefits of
calendar-based questionnaires and online data collection, we aimed to transfer EHCs from the
laboratory to the internet. While doing so, it is desirable to avoid the use of software with
rights reserved to the copyright holder. Hence, we implemented a solution that uses the
already existing framework of the open source software formr (www.formr.org; Arslan &
Tata, 2017). By using an open source software tool, we aimed to make the tool accessible to a
broad community of researchers and invite them to join in supporting its collaborative
development. Moreover, as formr makes it very easy to share study materials, it may help
support more transparent science and open methodology, by enabling other researchers to
reproduce exactly what someone did and replicate the results someone else found (see
Kraker, Leony, Reinhardt, & Beham, 2011; Open Science Collaboration, 2015).

**Tutorial: Implementing an Event History Calendar in formr**

**Study Example and Information about the Required Software**

**Study example.** For the purpose of the tutorial, we will make use of an exemplary
research goal, which will guide us through the steps of creating an online EHC. Imagine you
would like to collect data of all partners and ex-partners a person had. Furthermore, you aim
to include only participants that had at least two different romantic relationships, since this enables you to carry out some within-subject comparisons. In order to support the participants
during their recall of former partners and the corresponding relationship lifespans – this could
mean going back in time for several years – you plan to make use of an EHC during the
online assessment. This example EHC is supposed to collect information about residences,
occupations, and important life events of the participants, starting from the date of their first
relationship until today. The three listed calendar domains will be implemented in the study

\textsuperscript{1} Belli et al. (2007) programmed an EHC in house, with copyright 2000 to the Regents of the University of
Michigan. Brüderl et al. (2017) integrated the EHC as a Java application to their interview within the Panel
Analysis of Intimate Relationships and Family Dynamics (pairfam).
in the given order, and followed by a fourth domain that collects the dates and names of the participant’s partners and ex-partners.

Several research topics could be investigated following this specific data collection. Examples of such research issues are specified further below. For now, it is sufficient to keep in mind that the goal of our example EHC is to collect residences, occupations, important life events, and finally the names and relationship dates of all current and former partners of the participants. Please note that every entry to the EHC will be called event in order to increase readability – regardless of whether this entry describes a specific and discrete event, such as the birth of a child, or an extended episode of the person’s life, such as a residence, a job, or an illness.

![Figure 2. Example for a Gantt diagram, generated from fictive information that could have been collected within the event history calendar section of a survey. Current events have blue bars and past events have gray ones. The fictive participant first lived in Montreal where she visited High school. Now she lives in London and works at the West End Theater after a short episode of working for Pizza Hut. Her current partner is Chris; he is the third partner after Adam and Ben.](image-url)
About Gantt charts and mermaid.js. Mermaid.js\(^2\) is a markdown-like scripting language that can be used to generate charts from text via JavaScript (Sveidqvist, 2018). The so-called Gantt diagram is a type of bar chart that was originally developed to illustrate project schedules. However, it can be used to illustrate start and end dates of events, too. Events visualized in Gantt charts created with mermaid.js can be further categorized into different domains. Depending on the input, one receives a graphic that illustrates the temporal extension of various events, which are separated by their domain, but may refer to the same time frame. This way, temporal overlapping and order of events becomes visible, while clarity is ensured due to the separate sections for every domain. Figure 2 shows a Gantt chart created from fictive data, as it could result from a participant of our study example.

About formr. Formr is an open source software developed by Arslan and Tata (2017). It provides an online framework to set up online studies of varying complexity. Currently hosted by the University of Goettingen\(^3\), formr features three main components, namely a survey framework, a study control framework (Arslan, Walther, & Tata, 2018), and a utility R package (Arslan, 2018). The survey framework provides a regular questionnaire interface and allows researchers to pose questions to users and collect the corresponding responses. The items of a formr questionnaire or survey are specified in an item table, which simply is a spreadsheet that can be uploaded to the survey framework. The spreadsheet can be created locally as a Microsoft Excel file, OpenDocument spreadsheet, comma-separated value (CSV) or tab-separated value (TSV) file, or it can be created online using Google Spreadsheets\(^4\). It is recommended to use Google Spreadsheets, mainly because it enables researchers to collaboratively create studies online and facilitates rapid reloading of newer versions of the item table when debugging. Nevertheless, we will work with local Excel files in this tutorial because it is not sure if Google Spreadsheets will remain the same over the

\(^{2}\) https://mermaidjs.github.io/

\(^{3}\) https://formr.org

\(^{4}\) https://spreadsheets.google.com
next years. The study control framework of formr allows researchers to administer a study while performing complex tasks. Such tasks include managing access to a study, sending out reminders via email or text messages, giving feedback to users, writing R scripts for data analysis and overviews, or customizing who answers which questions and when. While the first component, the survey framework, is provided by most other software out there, the control framework is unique to formr. Finally, formr is accompanied by a utility R package\(^5\) (Arslan, 2018). It is independent of the survey and control framework, but helps to organize the collected data or to make further on-site analysis by providing R helper functions.

The study and control frameworks of formr are written in PHP, however, researchers can execute R code to control various study aspects via OpenCPU (Arslan et al., 2018; Ooms, 2014). Researchers familiar with R can use their favorite R packages to create dynamic survey texts, generate user feedback, or control further study aspects. In addition to the flexibility provided by R, researchers can further change the look-and-feel of their questionnaire by specifying custom Cascading styles sheets (CSS) and add front-end client functionality by writing custom JavaScript for their study. As will be shown further below, the latter is necessary to integrate mermaid.js with formr when creating an online EHC. In addition, studies created with formr can easily be combined with other survey software: Since it is possible to use the study control framework without the survey framework (but not the other way around!), formr studies can also be integrated with other software. Thus, if you would like to conduct an EHC study, but do not want to go without your favorite and familiar survey software, you could just set up a formr-based EHC as explained in this tutorial and add in questionnaires that are based on another online survey software before and after it\(^6\). Due to its ability to combine different programming languages and ease of use, formr is a very powerful tool to create a variety of innovative and complex online studies. More

\(^5\) [http://github.com/rubenarslan/formr](http://github.com/rubenarslan/formr)

\(^6\) Of course this includes the possibility of integrating software that is designed to run experimental studies online, such as the JavaScript library jsPsych ([https://www.jspsych.org/](https://www.jspsych.org/)) for behavioral online experiments.
information about formr can be obtained from the formr homepage, www.formr.org/about.

For a detailed documentation on how to use formr above and beyond the setup of an EHC as outlined in this tutorial, please visit www.formr.org/documentation and read the newly available preprint\(^7\) of Arslan et al.’s (2018) paper explaining the software\(^8\).

**Steps of Implementation: Combine formr.org and mermaid.js for an Online EHC**

In the next section, you will learn how to create an EHC that builds itself up based on the information a participant provides. In order to get an idea on how the EHC appears within an online survey, Figure 3 illustrates how the calendar displays the data from the first domain at the beginning of the survey and gets bigger with every domain and event the participant lists. We will start with an explanation on how to create a formr account, and afterwards look at some considerations that are useful when planning an online EHC. Finally and with the support survey materials that are available online, we will go through the steps of implementing the calendar.

---

\(^7\) Preprint available at [https://psyarxiv.com/pjasu](https://psyarxiv.com/pjasu)

\(^8\) In addition, when actually setting up your study, you might like to join the formr mailing list [https://groups.google.com/forum/#!forum/formr](https://groups.google.com/forum/#!forum/formr) to ask and answer questions.
**Figure 3.** Flowchart illustrating the continuous setup of the event history calendar: With every survey page a participant completed, a new domain is added to the Gantt chart.

**Step 1: Creating a formr account.** In order to use formr and its extended features, you will first need to obtain an administrator account. To do so, visit [https://formr.org](https://formr.org) and click on the *Sign up* link. Enter your email, choose a password and click on the *Sign up* button. If your registration was successful, you should receive an email confirming your registration. To complete the registration process, you will need to send a request to formr administrators to grant you administrator rights: Simply send a short message to one of the
email addresses that are listed when going to Documentation → Getting Started in formr.

Once the rights are granted, your account is ready to set-up your study.

**Step 2: Considerations in advance.** Before implementing the EHC in your online study, you need to think about the following:

- Which are the event “domains” you want to ask the participants about? Depending on the variables of interest, this could be anything. As Martyn and Belli (2002) note, only the most important domains should be included on the EHC, since large amounts of data are being collected. You should also think about the order: The EHC should start with a very general domain that is easy to remember and end with the most specific one, which often contains your variable of interest. In our example, the domains in their order are **Residences**, **Occupations**, **Events** (the participant can list any event that was personally important for her or him), and finally **Relationships**.

- How many events can a participant enter per domain? The number of possible events needs to be predefined. Think about a number of events that makes sense to you: On the one hand, participants should have the possibility to provide a lot of information. On the other hand, too many episodes in the diagram will look crowded. For the **Residences** and **Occupations** domain in our example study, participants will be required to name at least one event each. In contrast, listing events in the **Events** domain will be optional. Finally, participants will be asked to list at least two partners in the **Relationships** domain.

- What temporal categorization is sensible for your data collection? Should the participants date their events exactly to the day (which is usually utopian when collecting data retrospectively) or is it rather practical to ask participants to indicate

---

9 In our example, the number of events will be limited to 10 per domain. However, this is only a suggestion in the purpose of this tutorial and researchers are free to set the lower and upper limits of the number of events individually.
the week, month, quarter, or year of an event? In our example, participants will be asked to indicate the month and year in which events started and ended.

**Step 3: The Survey spreadsheet.** As outlined above, surveys in formr are designed using spreadsheets. For the purpose of this tutorial, we will use an Excel spreadsheet that already contains all item definitions that are required to replicate our exemplary study. All files used in this manuscript are hosted with the open science framework (OSF). Download the spreadsheet located at [https://osf.io/afzwt/](https://osf.io/afzwt/) and save it locally on your computer. The file’s name might not be changed, since formr creates the survey name from it and the code in the JavaScript settings (which will be explained later) refers to this name. Changing the spreadsheet’s name would result to an inoperable study unless the R and JavaScript code snippets are modified accordingly. If you prefer to work with Google Spreadsheets, you can clone the spreadsheet in your Google account. The provided spreadsheet contains a column with some explanations, to provide some orientation within the sheet. We will now have a look at the various items in the spreadsheet to understand how they contribute to the buildup of the EHC.

Remember that an EHC is organized thematically by domain and temporally by chronologically ordered events (Martyn & Belli, 2002). In our online implementation, every event collected in the calendar requires (1) a title (name of the event), (2) a start date, (3) an end date, or (4) an indicator specifying if the event has not yet ended. There should be a possibility of adding more than one event per domain, but a cap on the number of events needs to be specified.

*Define the earliest date.* Before we define the different domains, we need to define at what date participants can start to add events. Please open the downloaded spreadsheet: After a general instruction on the EHC, the item of the type month in row 3 requests the date of the participants’ first romantic relationship. The lower and upper limit of the month item
definition validates that users can choose dates not older than 20 years ago and not younger than the current day. In row 5, a calculate item with the name “first_date” stores the year and month that were indicated before via a function in the value column.

*Creating the Residences domain.* As you can see in the spreadsheet, the instruction in row 8 refers to the “first_date” variable that we created before via R code. When a participant opens the corresponding survey page, she or he will see the indicated date of the first relationship instead of the R code. Continuing with the spreadsheet, let us have a look on row 9 until row 50 concerning the *Residences* domain and compare them with Figure 4 for an explanation.
Figure 4. Spreadsheet defining items for the domain *Residences*. Participants will be allowed to enter a maximum of 10 residences.

Figure 4 illustrates how the items of the *Residences* domain are defined within the spreadsheet. As you see, four items are defined to collect the information (1)-(4) that is
required to describe one event (in this case one residence) as delineated above. In formr parlance, this translates to (1) an item of the type \textit{text} to enter a residence, (2) an item of the type \textit{month} to enter a start date, (3) a second \textit{month} item to enter an end date, and (4) an item of the type \textit{mc_multiple} to indicate an event as ongoing, i.e. not ended. The \textit{month} item for the end date and the \textit{mc_multiple} item for current events are optional (as marked by “*” in the \textit{optional} column), because these two are mutually exclusive in terms of content: An event has either ended and thus has an end date or is ongoing, but never both.

As noted before, the number of events a participant can enter needs to be predefined. To redefine the number of events that a participant can list within the domain, the four mentioned item rows need to be repeated for every potential event entry. In our case we decided to limit the number of events to 10 (see spreadsheet). This means that the four items in the spreadsheet with the names “address\_R1”, “datefrom\_R1”, “dateto\_R1”, and “current\_address\_R1” defining one event need to be repeated nine times. Note that the item names in the \textit{name} column of the spreadsheet need to be adapted when repeating the rows (e.g. “address\_R1”, “address\_R2”, “address\_R3”, etc.) because formr does not accept two items with the same name.

Even though we have predefined 10 events, a participant should only see and need to fill out as many residences as she or he had. Thus, we need a mechanism that controls how many item groups are displayed to the participants at a time. In the spreadsheet, we use the \textit{showif} column to hide the items (rows) not needed to be shown and add an item of type \textit{number} with the CSS special class called \textit{counter} at the end of the item definitions in row 49. Formr translates this \textit{number} item with the special class into buttons that can be clicked by the participant to add or remove item rows. This \textit{counter} needs a default value in the \textit{value} column to work. This value dictates “how many event rows” are shown to the participant by default. In our example, its default value is “1” because every participant needs to indicate at
least one residence. As you can see in the showif column, all events after the first one are assigned a show-if condition, defining that the set of items for a further event is only displayed if the counter variable exceeded a certain value. In practice, this means that the items collecting the title and dates of a further residence are only shown if the participant requested to add a new one. Moreover, note that if you would like to predefine more or less events, you would need to adapt the number of repetitions of the four rows defining one event, as well as the upper limit of the counter variable (lower and upper limit are defined by values next to the number item in the type column). Finally, every survey page needs to end with a submit item, a button that permits the participant to save his or her responses and move on to the next page of the survey. Figure 5 gives an impression of how the items defined in the spreadsheet are displayed to the participant.
Figure 5. Example for the user interface in formr with items collecting the information required for the event history calendar. The example shows items collecting residences and their temporal categorization as defined in Figure 4. Events are collected starting from February 2011 because the fictive participant indicated this as the date of the first romantic relationship.

Residences:

Please list all your residences where you have lived since 2011-02.
Note: We only explore the period of time since your first romantic relationship.
Thus, 2011-02 is the earliest date you can select in your answers. Places where you lived before this date are not to be listed.

<table>
<thead>
<tr>
<th>Residence</th>
<th>Move in date</th>
<th>Move out date</th>
</tr>
</thead>
<tbody>
<tr>
<td>Montreal</td>
<td>Feb 2011</td>
<td>Jun 2013</td>
</tr>
<tr>
<td>London</td>
<td>Jul 2013</td>
<td></td>
</tr>
</tbody>
</table>

Would you like to add a residence? [ ] [ ]

Go on!

Figure 6. Output of an event history calendar with fictive data collected during the Residences domain, corresponding to the data input that is shown in Figure 5.

Rows 51 and 52 of the spreadsheet define a validation mechanism for the Residences domain. It checks if all residences have labels and if the move-out date is always greater than the move-in date. Row 53 defines a formr note item that presents the calendar to the user for this domain. The label column of this row contains R code that aggregates data entered by a participant (on her or his residences) and creates the relevant markdown structure required by
mermaid.js to draw up the EHC. This markdown structure is read internally by JavaScript as
specified in the study settings (more on this later). The resulting calendar graphic, generated
from the input of the Residences domain and exemplarily displayed in Figure 6, will be
presented to the participants when processing the next domain on a new survey page. The full
R code, together with short explanations in comments highlighted in black is given below:

```{r}
# Function that collects an event’s title, start date, end date, and
# the information if it is current; the collected information is pasted
# in the format that can be read by mermaid.js

gannt_event_row <- function (i, data, col_title, col_datefrom,
    col_dateto, col_current) {
    empty <- ''

    if(!(paste0(col_title, i) %in% colnames(data))) {
        return (empty)
    }

    title <- last(data[paste0(col_title, i)])
    if (is.na(title) || title == '') {
        return (empty)
    }

    is_current <- last(data[paste0(col_current, i)]) == 1
    datefrom <- last(data[paste0(col_datefrom, i)])
    dateto <- ifelse(is_current, format(Sys.Date(), "%Y-%m-%d"),
        last(data[paste0(col_dateto, i)]))
    tag <- ifelse(is_current, "active", "done");
    label <- paste0("a", i)
    paste(title, ":", tag, ",", label, ",", datefrom, ",", dateto, " ",
        sep = " ")
}

# Vector that stores all information of the Residences domain: The
# function specified above runs in a loop for every entry of a
# participant
residences <- NULL;
for (x in 1:last(My_EHC_survey$count_SR)) {
    residences <- c(residences, gannt_event_row(x, My_EHC_survey,
        'residence_R', 'datefrom_R', 'dateto_R', 'current_residence_R'))
}

# Below, the text input for mermaid.js is given; it includes the vector
# with the information of the Residences domain

<div class="formr-mermaid">

Gantt

title Overview of residences

dateFormat YYYY-MM-DD

section Residences

`r paste(residences, collapse="\n")`
</div>
```
Creating the Occupations domain. For the next domain(s), the steps that were required for the Residences domain need to be repeated: Again, all items/events need to be predefined. Of course, all labels and variable names need to be adapted to the new domain. In our example, the second domain is the domain Occupations and its items are defined in row 54-96 of the spreadsheet. The four items required for one event will now be named differently. For example, we use the names “occupationtitle_O1”, “datefrom_O1”, “dateto_O1”, “current_occupation_O1” for the first occupation, followed by names with rising numbers for the items predefining the occupations from one to 10. After all items, including a further counter item for occupations in row 95, are defined in a manner that is parallel to the first domain, the rows defining a validation mechanism are inserted. Again, all variable names need to be changed in order to (a) differ from the Residence section, and (b) specifically refer to the Occupations section. After defining the items necessary for the Occupations domain, we use R to aggregate the collected data and display an updated EHC showing information from the Residences and Occupations domains. Again, short explanations are provided within comments that are highlighted in black. R code that is new compared to the R code printing only the first domain in a calendar is written in bold font:
```{r}
# ONLINE EVENT HISTORY CALENDAR

```{r}
gantt_event_row <- function (i, data, col_title, col_datefrom, col_dateto, col_current) {
  empty <- ''

  if(!(paste0(col_title, i) %in% colnames(data))) {
    return (empty)
  }

  title <- last(data[[paste0(col_title, i)]]);
  if (is.na(title) || title == '') {
    return (empty)
  }

  is_current <- last(data[[paste0(col_current, i)]]); == 1
  datefrom <- last(data[[paste0(col_datefrom, i)]]);
  dateto <- ifelse(is_current, format(Sys.Date(), "%Y-%m-%d"),
                     last(data[[paste0(col_dateto, i)]]));
  tag <- ifelse(is_current, "active", "done");
  label <- paste0("a", i)

  paste(title, ":", tag, ",", label, ",", datefrom, ",", dateto, " ",
        sep = " ")
}

residences <- NULL;
for (x in 1:last(My_EHC_survey$count_SR)) {
  residences <- c(residences, gantt_event_row(x, My_EHC_survey,
                                               'residence_M', 'datefrom_M', 'dateto_M', 'current_residence_M'))
}

# A second loop and vector are added to store the information of the 
# Occupations domain; again, the function specified above runs in a loop 
# for every entry of a participant

occupations <- NULL;
for (x in 1:last(My_EHC_survey$count_SO)) {
  occupations <- c(occupations, gantt_event_row(x, My_EHC_survey,
                                               'occupationtitle_O', 'datefrom_O', 'dateto_O', 'current_occupation_O'))
}

# Below, a new text input for mermaid.js is given: The title is 
# extended and the vector with the information of the Occupations domain 
# is added

<div class="formr-mermaid">
Gantt
title Overview of residences and occupations
dateFormat YYYY-MM-DD
section Residences
  `r paste(residences, collapse="\n")`
section Occupations
  `r paste(occupations, collapse="\n")`
</div>
Special case: Creating the Events domain. You might want to give participants the opportunity to name single date events as opposed to extended episodes. These kinds of events only require the first two pieces of information, (1) an event title and (2) a date. Therefore only two items, (1) an item of the type text and (2) an item of the type month need to be defined in the spreadsheet, with names following the scheme “eventtitle_E1”, “datefrom_E1”, and so on (see row 100-120 of the spreadsheet). The second month item (the end date) and the indicator of a current event, as they were used for extended events, can be left out. Additionally, we decided to make the Events domain in our example study optional – participants can list important life events but skip this section if nothing special happened in the relevant time interval. This is why all items of this domain except the counter (which is renamed and redefined again in order to refer to the Events domain) are marked as optional with “*” within the optional column of the spreadsheet. Both special characteristics – single date events and optional nature of the domain – require a new function and some adaptations of the R code printing the Gantt diagram. Again, short explanations are provided within comments that are highlighted in black and the new code is presented in bold font. To save space, the R code collecting information from the first two domains (with extended events) is replaced by “…”:
... [Function collecting information of the extended events]

```r
gannt_events <- function (i, data, col_title, col_datefrom) {
  empty <- NULL

  if(!paste0(col_title, i) %in% colnames(data) || is.na(data[paste0(col_title, i)])) {
    return (empty)
  }

  title <- last(data[paste0(col_title, i)])
  if (is.na(title) || title == '') {
    return (empty)
  }

  datefrom <- last(data[paste0(col_datefrom, i)])
  tag <- "done"
  label <- paste0("a", i)

  paste(title, ":", tag, ":", label, ":", datefrom, ":", datefrom, " ", sep = " ")
}
```

... [Loops and vectors storing information of the extended events]

```r
events <- NULL;
for (x in 1:last(My_EHC_survey$count_SE)) {
  events <- c(events, gannt_events(x, My_EHC_survey, "eventtitle_E", "datefrom_E"))
}
```

# Below, a new text input for mermaid.js is given: The title is extended and, provided that the participant added any events, the vector with the information of the Events domain is added

```mermaid
Gantt
    title Overview of residences, occupations, and events
    dateFormat YYYY-MM-DD
    section Residences
        `r paste(residences, collapse="\n")
    section Occupations
        `r paste(occupations, collapse="\n")
    `r ifelse(!is.null(events), paste(c("section Events", events), collapse="\n"), "")
```

A new function is used here to parse the entered *Events domain* data and generate the text structure required for the Gantt chart. Note here that the “start date” of the event is used as the end date to follow the pattern of the two previous domains. (Actually, mermaid.js is able to print Gantt graphs with a start date only, but it is necessary to insert an end date to keep the loop in the code above running.) Moreover, the function and the code of the *Events* section within the Gantt chart are specified in a way, that the domain is only printed if the participant added any life events. Otherwise, this part is just skipped.

*Creating the Relationships domain.* Until here, we have learned how to define the items and the Gantt charts of the three domains *Residences, Occupations*, and *Events*. In our study example, a fourth and final domain - the *Relationships* domain - is planned to collect information (i.e. names and relationship dates) about the participants’ current and former romantic partners. In contrast to the previous domains, participants will have to name at least two events (i.e. partners). Figure 7 illustrates the item definitions of our last domain in row 122-160 of the spreadsheet. As with the other events, each relationship is defined by four items with names following the scheme “partner_P1”, “datefrom_P1”, “dateto_P1”, and “current_partner_P1”. Please keep in mind that all partner names are stored in variables named “partner_P1”, “partner_P2”, “partner_P3”, etc. – we will keep working with these later when we assess more information on those partners. In order to increase the number of required event entries from one to two, the four items of both the first and second relationship are not bound to a *showif* condition. Instead, the *showif* conditions are attached to the items of the relationships three – 10, starting with “count_SP > 2” in the *showif* column of the items of the third relationship. The default value of the counter variable is changed to “2”. 
Since the extension of the R code for the Gantt chart is equivalent to the *Occupations* domain, it will not be further explained in this tutorial. Just remember that all variable names again need to be adapted in order to refer to the *Relationships* domain. At this point, our EHC
is complete and the Gantt chart displays all four domains. The survey ends with a final *submit* item.

*Prospect: Adding further domains.* The four domains of the example EHC are relatively generic and can probably be used for many studies dealing with relationship-related topics. Nonetheless, the content of the domains could be substituted and researchers can readily add further domains in the same manner in which the four domains in the example study were created. For instance, a fifth or sixth calendar domain could be used to collect and visualize more specific relationship events and aspects. Naming just a few, one could assess health-related or psychosocial issues of the participant and her or his partner, sexual affairs, marriage, or the birth of children by adding corresponding domains to the calendar. As a next step, an additional survey could be created to gather more information about each partner and relationship the participants listed. The implementation of corresponding items will be explained further below. First, we will proceed with the implementation of the EHC itself.

*Step 4: Survey upload and settings in formr.* The spreadsheet created above now needs to be imported to formr. To do so, visit https://formr.org/login and enter your login credentials. On success, you will be redirected to the admin section of formr. Go to *Surveys → Create new Survey* in the top menu. Depending on the format in which you saved the spreadsheet, you either need to upload it from your local computer or you import a Google spreadsheet with a link to the online file. If you decide to import a Google spreadsheet, enter the survey name “My_EHC_survey”. Please follow this nomination as this is used to refer to variables of the spreadsheet’s R code snippets and in the CSS and JavaScript code snippets, too. A change in this name would mean that you need to modify the code snippets accordingly. Moreover, please make sure that you have set appropriate permissions on your Google spreadsheet to be readable for everyone with the sheet’s link. Click the *Import* button to complete the upload of the spreadsheet.
To permit users navigating back and forth within the EHC study and make changes to already given answers, we need to configure our created survey to allow paging. Go to Surveys → My_EHC_survey. This will lead you to the settings of that survey. Scroll to the bottom of the page, tick the Enable Paging checkbox, and save your settings. Please be aware of the fact that you cannot reverse this function, once it is enabled. There are many survey types where participants should not be allowed to edit their answers. However, during an EHC study this might be very useful because it is likely that participants remember events from an earlier domain while working on a later one or realize that they mixed up some dates.

**Step 5: Setting up the EHC study: The formr run.** From this point on, we will operate within the study control framework of formr. After creating and importing your questionnaire spreadsheet as a formr survey, the next step will be to set up the control structure of your study, in formr parlance a run. Go to Runs → Create new Run. Enter the name of the run, e.g. “EHC-study”. Please note that within formr, each possible run name can only be used once - if a name is already taken, a warning message will be shown and you need to choose another name. The just mentioned name will be used in this tutorial in order to refer to the exemplary study. On success, you will be redirected to the study control interface. While a survey represents a part of the study, the run represents the whole study structure, which usually includes one or more surveys. It comprises sub components known as run units. Each unit carries out a specific task and can be added to the run with the click of the mouse on the appropriate run unit button located at the bottom of the study control interface. For example, an Email unit sends out emails, a Survey unit displays a questionnaire (survey) and so on. For more on formr run units consult the formr documentation. As a minimal set up, we will need a Survey unit to display our questionnaire and a Stop Point unit to end the study and display a message to the participant when the study is over. If you are not already

---

10 [www.formr.org/documentation](http://www.formr.org/documentation)
there, go to Runs → EHC-study. At the bottom of the page, you should see the panel depicted in Figure 8.

Figure 8. Panel with buttons that can be clicked to add units to the formr run.

Adding the run units: Click on the Add survey button and use the dropdown to select the name of the survey you created in the previous section. In our case that will be “My_EHC_survey“.

Next, click on the Add a stop point button and enter some text you would like to display to the participant at the end of the study. It could be something as simple as “Thank you for participating!” or some complex feedback with interactive charts (written in R). It is up to the researcher what goes in there.

With the above set up of our study, we will now add some settings to the formr run. As mentioned earlier, formr gives you the possibility to add custom CSS and JavaScript to your study. The CSS setting will enhance the look and feel of the items displayed to the user while the JavaScript setting will contain code responsible for performing some client side validation and also loading the mermaid.js library responsible for displaying the EHC.

CSS Setting: Still in the run control interface of your run, click on the Settings button located at the left menu. Go to the CSS tab, copy and paste the code snippet located at https://osf.io/5nzgm/. This brings some of the items into position and defines the colors used in the EHC Gantt chart.
JavaScript Setting: Now go to the JS tab of the settings page and paste the code snippet located at https://osf.io/3vzwh/. This JavaScript code is essential to make the study work. Therefore, we will have a closer look on it: We are using mermaid.js for displaying the EHC on the browser. This JavaScript library accepts a Gantt chart definition in text form – as we produced it with the R code in the survey spreadsheet – and outputs the required graphics. The following code provides the minimal JavaScript setup, which is required to run our EHC study since it connects formr with mermaid.js:

```javascript
// retrieving mermaid.js and applying it to formr text with the class "formr-mermaid"
var script = document.createElement('script');
script.onload = function () {
    mermaid.initialize({startOnLoad:false});
    $(document).ready(function() {
        $($.".item-note .formr-mermaid").each(function() {
            var $this = $(this);
            $this.text($this.text());
            $this.show();
        });
        mermaid.init({noteMargin: 10}, ".item-note .formr-mermaid");
    });
    script.src = "https://unpkg.com/mermaid@8.0.0-rc.8/dist/mermaid.js";
    document.head.appendChild(script);
}
```

The code above searches for an HTML element in the page with the class name "formr-mermaid”. This element has been generated with R code in the spreadsheet (see code defined during the spreadsheet creation) and contains the required Gantt chart definition in text form. The mermaid.js library is applied to this element, converting its text content into a graphic displaying the EHC. Please note that the web address for the mermaid.js script (displayed in the last row of the code above) changes from time to time and therefore may need to be updated.

Furthermore, you can edit the labels of the buttons to skip forward and backward within the EHC survey with help of the following code in the JavaScript settings of the run:
Every page is given a meaningful label to assist the participant navigate back and forth in the study. Moreover, you can adapt the labels “Page” and “Back to” via the first two lines of code, e.g., if you would like to run your survey in a language that is not English.

Finally, we added some JavaScript code that performs a validation of the participant’s date input, which goes beyond the validation mechanism that was defined within the spreadsheet. Roughly summarized, the validation code causes error messages to pop up if the participant enters invalid data, such as an end date that is earlier than the start date of the same event. Generally, JavaScript gives you the flexibility to add more, sometimes fancy details to your formr study.

Figure 9. Publicness settings in formr. By default, the study can be accessed by the study administrator and people with an access code.
Step 6: Testing the study. Now that the run is created and all its settings are configured - here, this should be the case right after uploading the spreadsheet to formr, adding it as a survey to the run and configuring the CSS and JavaScript settings - the next step is to test the study. In the control framework of formr, you can determine the publicness of your study. The buttons for the publicness settings, as they are depicted in the formr control framework, are shown in Figure 9. By default, publicness is set to access for you and everyone with an access code that you can distribute. With this setting, you can test and change your study without being afraid that users will see it. However, you have the possibility to send access codes to people you know to let them run the study on trial. To test the study on your own, click on the Test Run button on the left side of the study control panel. This will enable you to click through the study and see how it appears and if everything is working. Deviating from real users, you can make use of the monkey mode and skip run sections. The monkey mode is a function in formr that automatically fills out the survey pages - this can save you a lot of time when testing large questionnaires again and again! Furthermore, the test run might display warning messages to you, which will not be apparent for real users.

Every time you start a test run, a new user/test session is created. You can view all existing test sessions by clicking on the Old Guinea Pigs button right under the Test Run button. Session codes of test users contain animal names and the letters “XXX”. This will enable you later to differentiate between test and real users. Within the user overview, you can delete old test session, send users to certain positions in the run, and test the study again with their session code. By right-clicking on the symbol in the Action column and choosing the option Copy link address, you can retrieve the access code to this test and send it to someone.
Step 7: Running the study for real. After testing the run and finding everything to work correctly, your study is ready to go public. Depending on who you want to reach, you can either choose the inner or outer left of the two left buttons of the publicness settings (see Figure 9). In the first case, everyone with the official study link - which is found right on top of the study control panel, next to the run name - can enroll to the study. When choosing this option, you will need to distribute the link via e-mail or online platforms. Probably, most researchers will prefer this publicness setting because it provides them with control over the study access. In the second case, the study link will be publicly posted to the study page in former and everyone visiting the page can enroll. Of course, the study link can be distributed via additional ways in this case, too.

Managing users. To get directed to the user overview, click on the Users button on the left side of the control panel. Actually, this leads you to the same page as the Old Guinea Pigs button, except that search function does not filter for the “XXX”, which is specifying test sessions as opposed to real sessions. Again, you can track the users’ progress on the basis of their run position, send them to another run position, or delete their session.

Viewing and exporting results. Formr saves the user data in a result chart, with one row for every participant or observation, and one column for each of the study variables from the survey spreadsheet plus information about the processing times. Results are stored separately per survey. This is especially helpful when many variables are assessed and result tables would get very large otherwise. To access the results of one formr survey, go to the corresponding survey page. In our case, this means going to Surveys → My_EHC_survey. To view the results online, click the Show Results button on the left side of the page. To export the results, click on the Export Results button underneath and choose one of the file formats, depending on your preferences or the preferences of your statistical program.

https://formr.org/studies
Beyond the Calendar: Embedding the Tool into a Larger Study

General Options to Use and Extend the Event History Calendar

With the steps instructed so far, you know everything necessary to create a minimal set up for an EHC study. So far, the information that can be collected with this EHC would enable researchers to count the number of long-term partners and calculate the corresponding relationship durations. It would also be possible to look at the time spans of residences or occupations and check whether moves temporally coincide with breakups. As noted before, further domains could be added to collect further and more specific information.

However, you might want to embed the calendar into a larger study that includes further questionnaires. For instance, you will most likely be interested in gathering some demographic data of the participants. In addition, there are several research questions going beyond the EHC domains we have generated in our example study and thus going beyond the examples outlined above. Imagine that your goal was to assess further attributes of the partners and relationships listed in the EHC. These attributes could be compared across the relationships with respect to similarity or variability: Do the different former and current partners of a participant have certain traits in common? How (dis-)similar are the different relationships, e.g. with regard to their quality or the power balance as perceived by the participant? In general, embedding the EHC in a larger study becomes feasible by simply uploading and adding further surveys to your formr run. This can either happen in the form of additional Survey units or alternatively with External link units directing to surveys that are implemented with other survey software. In the next section, we will learn how to create a second formr survey that gathers further details about all partners and relationships a participant listed during the EHC survey.
Gather More Information about Relationships Listed in the Calendar

Let us consider the exemplary and extended research goal from above in more detail: To find out if all current as well as former partners and relationships of one person have certain attributes in common, information about every partner and relationship a participant listed needs to be obtained. For example, one could examine specific personality characteristics of the partners and ex-partners, ask if any children were or are planned in the relationship, or who initiated the break-up in the case of ex-relationships. This way, the EHC would be the first step in a study by collecting the names and dates of all partners and ex-partners, followed by the assessment of more details on the corresponding relationships. The steps that are required to create a second survey that gathers more information about the relationships will be explained next. In order to distinguish these steps from the steps of implementing the EHC itself, they will be ordered by letters instead of numbers.

Step A: Creating a new spreadsheet. In the beginning, we need to create a new spreadsheet. Please download the spreadsheet named “Relationship_details” located at https://osf.io/txpd9/ and save it locally or add it to Google Spreadsheets. Basically, this spreadsheet is just another item table (former survey definition). After completing all steps explained in this section, all items in the “Relationship_details” spreadsheet should be repeated for each partner a participant listed during the EHC survey. Therefore, the new survey needs to refer to our first survey called “My_EHC_survey” and retrieve some of the variables stored in the corresponding result file. When opening the “Relationship_details” spreadsheet, you will find calculate items in the first five rows. These item definitions are illustrated in Figure 10 and add some useful adaptations to the survey via their functions defined in the value column, as will be explained below.
Figure 10. Spreadsheet defining calculate items that are used to tailor the survey’s appearance based on the partner’s name and relationship status (i.e. being a current partner or ex-partner).

**Defining calculate items.** First, the calculate items enable us to insert the correct partner name for each partner about whom the questions are posed. Remember that, in our first survey, partner names were stored in variables named “partner_P1”, “partner_P2”, etc. The first calculate item in row 2 recognizes how often the survey has been processed and thus stores the value “1” for the first partner, “2” for the second partner, etc. In row 3, the second calculate item refers to the ”My_EHC_survey” survey and merges the string “partner_P” with these numbers, resulting in the names of the variables that store the partner names. Together, the two calculate items can retrieve the correct name of each listed partner and ex-partner. Thus, when inserting the variable “name” into the spreadsheet’s label column of an item via R code, the correct partner name will be displayed in its place when a participant processes the survey.

Second, the calculate items enable us to switch between presence and past tense, depending on the fact whether questions refer to a current partner or to an ex-partner. Going to row 4, the calculate item retrieves and stores the information if a partner is a current

<table>
<thead>
<tr>
<th>type</th>
<th>name</th>
<th>showif</th>
<th>optional</th>
<th>label</th>
<th>columns: choice 1-4</th>
<th>value</th>
</tr>
</thead>
<tbody>
<tr>
<td>calculate</td>
<td>person_nr</td>
<td>Variable stores for how many partners the current survey has been processed</td>
<td></td>
<td>row(Relationship_details)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>calculate</td>
<td>name</td>
<td>Variable retrieves partner names from the EHC survey, starting with name stored in the variable „partner_P1“</td>
<td></td>
<td></td>
<td></td>
<td>toString(last(My_EHC_survey)</td>
</tr>
<tr>
<td>calculate</td>
<td>current</td>
<td>Variable stores value „1“ if partner was marked as current in the EHC survey</td>
<td></td>
<td></td>
<td></td>
<td>false(toString(last(My_EHC_survey)</td>
</tr>
<tr>
<td>calculate</td>
<td>had</td>
<td>Variables calculate words in present tense for current partners and words in past tense for ex-partners</td>
<td></td>
<td></td>
<td></td>
<td>false(current == 1,”has”, ”had”)</td>
</tr>
<tr>
<td>calculate</td>
<td>were</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>false(current == 1,”are”, ”were”)</td>
</tr>
</tbody>
</table>
partner or an ex-partner from the EHC survey. This information can be used by the last two 
*calculate* items: If partners are current, the *ifelse* conditions in the *value* column produce the 
words written in present tense. Else, the words in past tense are produced. Thus, when 
inserting the variables “had” or “were” into the spreadsheet’s *label* column of an item via R 
code, the correct tense of the words will be displayed in its place when a participant processes 
the survey.

*Define relationship-specific questions.* After all required *calculate* items are defined, 
we can move on to the actual questions about the partners and relationships. To provide a 
temporal orientation and facilitate the recall of this information, we can reprint the complete 
EHC diagram to the top of the surveys page by pasting the R code into the *label* column of a 
*note* item, which is the first item of the survey page: This R code resembles the code that was 
used to print the complete calendar at the end of the EHC survey (thus visualizing the four 
domains *Residences, Occupations, Events,* and *Relationships*), except that it specifies the file 
“My_EHC_survey” as the origin of all variables. When not specified, formr would search for 
the variables in the current survey sheet – which is “Relationship_details” – without finding 
them.

Below the EHC graphic, questions are posed that will be repeated for every listed 
relationship. To put it into more detail, our example spreadsheet defines one set of items 
concerning the attributes of each partner or ex-partner in row 8-12. For example, one item 
asks how trustworthy a partner is. Within the item definitions, R code refers to the *calculate* 
items we created before. Whenever a participant opens the survey page, the values stored in 
these *calculate* variables are shown in the place of the R code. Thus, if a participant listed a 
partner named Peter, the mentioned item will read “What do you think, how trustworthy is 
Peter?” A second survey page in our example study starts again with the EHC diagram on top 
and includes a set of items concerning the relationship or ex-relationship, defined in row 15-
23 of the spreadsheet. For instance, one item asks about the power distribution within current and past relationships. With help of the calculate items and R code, the corresponding variable will show the verb form of “have” depending on the relationship status of the partner or ex-partner in question. If the question concerns a current partner, the item will read “In your relationship, who has more power?” Otherwise, the item will read “In your relationship, who had more power?”

Please note that the items given in the provided spreadsheet only represent examples. You can use the spreadsheet as a template for modification and replace or add further questions that fit your specific research goal.

**Step B: Survey upload.** When the second spreadsheet is defined, it needs to be uploaded to formr again. Please proceed as you did with the first survey. Once again, make sure that you keep the survey name “Relationship_details” to keep all references to the survey working. After the upload, you can decide if custom paging should be enabled in the survey about the relationships too and correspondingly put the tick in the survey settings or not. Within the tutorial, we go without paging in this survey, since we are interested in ratings that are given spontaneously and should not be altered later on.

**Step C: Extending the formr run.** In order to get the survey running, we need to add it to our already existing run. Please add a second Survey unit and import the “Relationship_details” survey via the dropdown of the unit. In order to let the survey run as a loop, which is getting repeated for each partner a participant listed in the EHC survey, we need to add a Skip Backward unit (see Figure 8). The following R code is stored in a file at https://osf.io/q4f9d/ – please download the file and paste the code into the code field of the unit. It defines the condition on how and when another run unit gets repeated, comments highlighted in black are added for explanation:
After pasting this, set the position of the control panel saying \textit{...skip backward to to}\n“10” and save the changes. The code checks (a) how often the \textquotedblleft Relationship_details\textquotedblright survey has run and (b) if there is a further partner that has not been rated yet. When all listed partners have been rated, the loop ends and participants are directed to the next unit, which should be the \textit{Stop Point} unit and study’s end in our case.

Please make sure that the run units have the correct run position (otherwise the study will not work). Run positions can be edited by increasing or decreasing the large numbers at the bottom of a run unit: The run should start with the \textit{Survey} unit with the survey \textquotedblleft My_EHC_survey\textquotedblright in run position \textquotedblleft 0\textquotedblright, followed by the \textit{Survey} unit with the \textquotedblleft Relationship_details\textquotedblright survey in run position \textquotedblleft 10\textquotedblright, and the \textit{Skip Backward} unit in run position \textquotedblleft 20\textquotedblright. Finally, the \textit{Stop Point} unit is placed in run position \textquotedblleft 30\textquotedblright. After editing the run positions, click on the \textit{Reorder} button at the top of the run control panel. At this point, your extended EHC study that can gather further relationship details should be ready for testing. Please follow the steps for testing the run that were explained in the section referring to the EHC implementation before.

In formr, there is a possibility to export the run structure defined above to a JSON file. This file can then be shared with other researchers who are maybe interested in collaboration or study replication. Such a JSON export containing a run definition can be imported to formr and you will not need to do the setup a run manually as explained above. The link \url{https://osf.io/kxdru/} will direct you to the JSON export of our example study. To test this JSON export, download the file from the above URL and save on your local computer, create

```
library(dplyr)
# Retrieve the number of partner variables stored in the EHC result file
person_name = My_EHC_survey %>% select(contains("partner_P")) %>%
tidyr::gather() %>% na.omit() %>% select(value) %>% pull(value)
# check if the number of processed partners is smaller than the number of listed partners
nrow(Relationship_details) < length(person_name)
```
a new formr run, click on the Import button at the top of the run control panel and select the downloaded JSON file. After clicking on the green Import button, the whole run will be built up based on the structure stored in the JSON file and you can see what the online EHC and the survey gathering relationship details should look like.

**Discussion**

In this tutorial, we provided a detailed instruction, explaining how to create an online EHC using formr.org (Arslan & Tata, 2017) and mermaid.js (Sveidqvist, 2018). In our eyes, online EHCs hold the potential to make a great contribution to relationship science by combining the methodological benefits of calendar-based questionnaires and online data collection: Online EHCs enable researchers to improve the quality retrospective data collection with the help of autobiographical memory cues (Belli, 1998; Freedman et al., 1988; Martyn & Belli, 2002). At the same time, the online implementation saves resources and allows collecting data from a large sample easily (Riva et al., 2003; Selm & Jankowski, 2006; Wright, 2005). A particular benefit of the EHC online implementation introduced in this tutorial is the use of open source software (see Kraker et al., 2011). The flexible framework of formr.org org (Arslan & Tata, 2017) can be accessed without charge and supports collaborative development as well as open methodology. Furthermore, surveys created from other survey frameworks can be integrated seamlessly to the formr study control framework.

Next to its advantages, it is important to consider some limitations of online EHCs. First of all, disadvantages applying to EHCs in general affect online EHCs as well. Probably most important, EHCs cannot replace longitudinal studies: The use of an EHC helps to improve recall, but neither guarantees for completeness of the recollections, nor fully compensates for retrospective biases (Schwarz, 2007). At least in this tutorial, the data collected with the EHC are self-reports. Please keep in mind that this aspect can lead to biased ratings, too (e.g. by idealizing current partners and devaluing ex-partners; Busby,
Holman, & Niehuis, 2009; Geher et al., 2005). As a solution to this problem, it is be possible to add reports of informants or the partners themselves. However, the explanation of the required steps would go beyond the scope of this tutorial.

In addition, the extensive autobiographical reconstruction within an EHC can become quite time consuming for the participants. This is especially true when EHCs are combined with further questionnaires. Second, the many advantages of online studies are accompanied by disadvantages, too (Kraut et al., 2004; Selm & Jankowski, 2006; Wright, 2005). For instance, older people may do not know how to operate with computers and internet (however, this population excluded from online studies decreases; Mossberger, Tolbert, & McNeal, 2007). More specific to EHCs, going without an interviewer leaves the user of the online survey all alone with the instructions of the EHC. During an EHC that is performed as a face-to-face interview, the interviewer has the chance to double check answers or to ask for more details. When employing an EHC online, it is crucial to equip the survey pages with very clear instructions that are as straightforward as possible and effectively guide participants what to do. In addition, the conversational style of the interview in face-to-face EHCs holds the potential to improve the respondents’ recall (Belli, Shay, & Stafford, 2001; Brüderl et al., 2017). In an online EHC, one can mimic the conversational style of face-to-face interviews to a certain extent, for example by using direct questions that prompt participants to reconsider their answers. Finally, it should be noted that the EHC online instrument has not been formally validated yet. However, user comments in a first study using the online EHC introduced in this tutorial support the memory structuring function of the dynamically created Gantt chart and positive participant experience when reconstructing their relationship histories (Wieczorek, 2018).

To conclude, researchers should carefully consider the advantages and disadvantages of online studies as well as the trade-off between benefits and effort for participants and
researchers resulting from the use of an EHC. If data quality can be improved with the help of a calendar instrument and online collection of data is deemed appropriate, the online EHC presented in this tutorial may be a promising and flexible tool that may crucially aid in answering cutting edge questions in the field of relationship science. Given the open source nature of the formr-based online EHC, researchers are invited to adapt it for their own research goals and to participate in its further development and optimization.
References


