jsPSychR to create experimental paradigms with jsPsych: simulate participants and standardize the data preparation and analysis

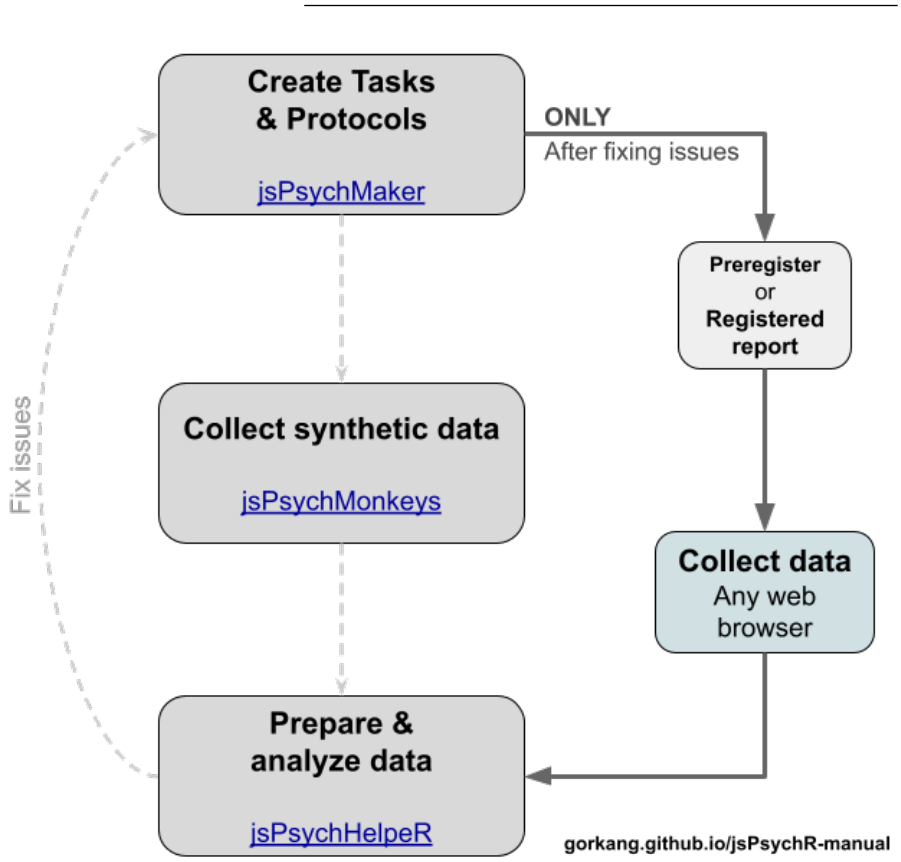
Gorka Navarrete Herman Valencia

2026-05-19

Table of contents

What is jsPsychR?

`jsPsychR` is a group of **open source** tools to help create experimental paradigms with jsPsych, simulate participants and standardize the data preparation and analysis. The final goal is to help you have the data preparation and analysis ready before collecting any real data, drastically reducing errors in your protocols, and making the move towards registered reports easier.



We have three main tools:

- jsPsychMaker: Create experimental protocols with jsPsych, randomize participants, balance between conditions, reuse already existing tasks, etc.
- jsPsychMonkeys: Release monkeys to a jsPsych experiment with {targets}, docker and {RSelenium}
- jsPsychHelpeR: Standardize and automatize data preparation, analysis and reporting of jsPsych experiments created with jsPsychMaker

And a package for the system admins:

- jsPsychAdmin: Functions to help with common administrative tasks for jsPsychR protocols

---

Everything in 3 minutes

jsPsychR creating a protocol with 3 tasks, running 10 simulated participants, and finally, data preparation… The whole process takes 3 minutes.

https://www.youtube.com/watch?v=2OXI9lzE3zU

Contributors

- Gorka Navarrete
- Herman Valencia

Citing jsPsychR

If you use the jsPsychR tools for your research, you can cite us with:

- *Navarrete G, Valencia H (2023). "Create experimental paradigms with jsPsych, simulate participants and standardize the data preparation and analysis.", 173. doi:10.5281/zenodo.8296995*

Papers published using jsPsychR

So far, the following papers using jsPsychR have been published or submitted:

- Neely-Prado, A., van Elk, M., Navarrete, G., Hola, F., & Huepe, D. (2021). Social Adaptation in Context: The Differential Role of Religiosity and Self-Esteem in Vulnerable vs. Non-vulnerable Populations–A Registered Report Study. Frontiers in Psychology, 12, 5257. https://doi.org/10.3389/fpsyg.2021.519623
- Morales, Juan Pablo and Ryan, Brenda and Polito, Vince and Vergara, Mayte and Navarrete, Gorka and Huepe, David, Can Beliefs Improve Mental Health? A Dive into Resilience During Pandemic Times in South America. http://dx.doi.org/10.2139/ssrn.4634882

1 Reproducible experiments

We use different technologies to develop experiments. Some examples are Psychopy, Qualtrics, Limesurvey, jsPsych, Gorilla, etc. Each of these has advantages and disadvantages and, in general, there are pragmatic aspects to take into account when adopting one or the other: cost, type of experiment (EEG or behavioral, lab or online), lab history and available resources, ...

In our lab, we opted for jsPsych to run behavioral experiments because it is an **open source** javascript library, based on standard web technologies, and can be used online and offline.

In the last years, we started working on a set of tools to help people without coding expertise to create jsPsych experimental protocols (jsPsychMaker), simulate participants (jspsychMonkeys) and standardize and automatize the data preparation and analysis (jsPsychHelpeR).

Our final goal is to have a big catalog of tasks available to use in the jsPsychMaker repo. Each of the tasks should run with jspsychMonkeys to create virtual participants. And each task will have a sister script in jsPsychHelpeR to fully automate data preparation (re-coding, reversing items, calculating dimensions, etc.).

1.1 Open and reproducible pipeline

To replicate an experimental protocol from a publication is not trivial. Obels et al. (2020) checked the computational reproducibility of Registered Reports in Psychology. From 62 articles meeting the inclusion criteria, only 21 had both data and code, and could be computationally reproduced. One of the main goals of jsPsychR is to be able to create, share and reproduce an experiment, its data, and data preparation and analysis without any extra effort. If recent calls for Journals to assess computational reproducibility are successful (Lindsay 2023), this should be an unavoidable aspect of researcher's work soon enough.

Furthermore, all the components of the pipeline are be Open Source, which allows reviewers, collaborators, etc. to check and run the code. This also makes it accessible to anyone with a computer connected to the internet, eliminating cost constrains.

With this system you can create a paradigm, simulate data and prepare data and analysis almost automatically.

The system output is standardized, so names of variables and the structure of the data are predictable. Finally, the plots, tables, reports and analysis are reproducible, so you can get

everything ready with simulated data, preregister or even better, go for a registered report and just relaunch the data preparation and analysis when the participant's responses arrive, with a single command.

And if you want to share the final data preparation and analysis project in a Docker container to make sure the future generations will be able to run it without dependency issues, we got you covered.

1.2 Automatization

We tried to get a few basic things right, but this is an evolving project, and some things are more complex than one would want. Please do report the issues you find:

- jsPsychMaker issues
- jsPsychMonkeys issues
- jsPsychHelpeR issues

Figure 1.1: SOURCE: https://xkcd.com/1425/

2 Quick Guide

Pre-requisites

For jsPsychHelpeR and jsPsychMaker, you will need R and RStudio desktop
For jsPsychMonkeys, you will need Docker. See the Monkeys' setup section for more detailed instructions.

2.1 jsPsychMaker: Create an experimental protocol

See the jsPsychMaker chapter for more detailed instructions.

Outline

1) Install jsPsychMaker
2) `create_protocol()` using any of the `list_available_tasks()` or your own tasks defined in csv/xls files, and edit the `config.js` to adapt the protocol settings
3) Open `index.html` in your browser

---

1) Install jsPsychMaker

Open RStudio and run the following line in the console. This will install the jsPsychMaker package from the Github repository.

```
if (!require('pak')) utils::install.packages('pak'); pak::pkg_install("gorkang/jsPsychMaker")

# If you are on Ubuntu and you get an igraph error, try: sudo apt install build-essential gfo
```

2) Create protocol

Create and test a fully working protocol with `jsPsychMaker::create_protocol()`.

Include the `canonical_tasks` you want (list the available tasks with `jsPsychMaker::list_available_tasks()`). You have more details in available-tasks. If you need new tasks, see New tasks.

```
  jsPsychMaker::create_protocol(canonical_tasks = c("AIM", "EAR", "IRI"),
                                folder_output = "~/Downloads/protocol999",
                                launch_browser = TRUE)
```

You must edit `config.js` to adapt the protocol to your needs. See experiment configuration for more details.

3) Run experiment

The experiment is ready to run on your computer. Open `index.html` in Google Chrome or your favorite (and up to date) browser.

2.2 jsPsychMonkeys: Simulate participants

See the jsPsychMonkeys chapter for more detailed instructions.

---

jsPsychMonkeys uses Selenium inside a Docker container to guarantee each session is a clean session. See how to setup your computer.

Outline

1) Install jsPsychMonkeys and Docker

2) Run Monkeys

---

1) Install jsPsychMonkeys and Docker

```
if (!require('pak')) utils::install.packages('pak'); pak::pkg_install("gorkang/jsPsychMonkeys

# If you are on Ubuntu and you get an igraph error, try: sudo apt install build-essential gfo
```

Go to Docker Desktop and install it.

2) Run Monkeys

If you are on Windows, make sure Docker Desktop is open and running before releasing the monkeys.

Use the `uid` parameter to set the participants' numeric id's, e.g. `uid = 1:10` would launch monkeys 1 to 10.

Use the `local_folder_tasks` parameter to indicate the location of the `jsPsychMakeR` protocol. If you are on Windows, `local_folder_tasks` value should be something similar to `C:/Users/myusername/Downloads/protocol999`.

```
jsPsychMonkeys::release_the_monkeys(uid = 1:10,
                                    local_folder_tasks = "~/Downloads/protocol999/")
```

If the protocol was running from a local folder, the Monkey's responses will be copied to a subfolder `.data/` inside the `local_folder_tasks`. In the example above, `~/Downloads/protocol999/.data`. If the protocol was running on the server (see the `server_folder_tasks` parameter), the data will be in the protocols' `.data/` folder inside the server.

2.3 jsPsychHelpeR: Prepare data

See the jsPsychHelpeR chapter for more detailed instructions.

Outline

1) Install jsPsychHelpeR
2) Create new project
3) Run data preparation

---

1) Install jsPsychHelpeR

- Install jsPsychHelpeR from Github.

```
if (!require('pak')) utils::install.packages('pak'); pak::pkg_install("gorkang/jsPsychHelpeR"

# If you are on Ubuntu and you get an igraph error, try: sudo apt install build-essential gf
```

2) Create new project

Create and setup a new RStudio project for your data. Before doing this, you need to locate the raw data for the jsPsychMaker project.

In this example, our raw data is in `~/Downloads/protocol999/.data/` and we want the new data preparation project to be in `~/Downloads/jsPsychHelpeR999/`

```
jsPsychHelpeR::run_initial_setup(pid = '999',
                                 data_location = '~/Downloads/protocol999/.data/',
                                 folder = '~/Downloads/jsPsychHelpeR999/')
```

After this, **a new RStudio project will open**.

3) Run data preparation

Run the data preparation **in the new RStudio project** with `targets::tar_make()`

```
# Restore all the necessary packages using renv
renv::restore(prompt = FALSE)

# Run data preparation
targets::tar_make()
```

If you are curious, running `targets::tar_visnetwork(targets_only = TRUE)` will show the whole data preparation targets tree. Open the file `run.R` for more details.

---

3 jsPsychMaker

---

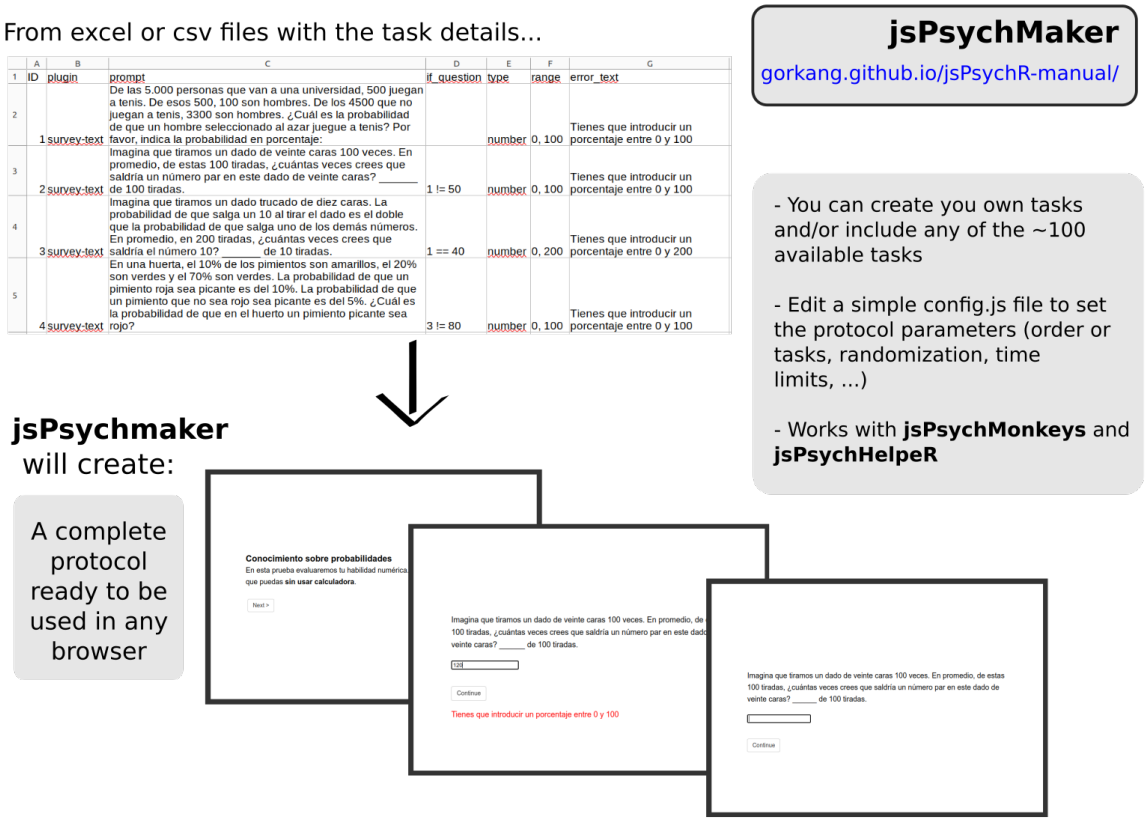
jsPsychMaker: Create experiments with jsPsych, randomize participants, etc.

In brief

Using jsPsychMaker to build experimental protocols helps you with a few things:

- Create full protocols using:
  - tasks already implemented
  - task that will be created reading a csv or excel file
- Configure your protocol by editing a simple config.js file
  - You can also use the jsPsychMaker Shiny APP to edit your config.js file
- Select order of tasks, randomize blocks of tasks, etc.
- Randomizes participants to groups, making sure the balance between the groups is maintained
- Allow participants to continue in the task where they left in the protocol
- Set time limits to complete the protocol
  - Automatically discard participants over the time limit, freeing the slots for new participants
- Run online and offline protocols
- Simulate participants with jsPsychMonkeys
- *Automagically* get your data prepared with jsPsychHelpeR

See QuickGuide for basic instructions.



	ID	plugin	prompt	if_question	type	range	error_text
1	1	survey-text	De las 5.000 personas que van a una universidad, 500 juegan a tenis. De esos 500, 100 son hombres. De los 4500 que no juegan a tenis, 3300 son hombres. ¿Cuál es la probabilidad de que un hombre seleccionado al azar juegue a tenis? Por favor, indica la probabilidad en porcentaje:		number	0, 100	Tienes que introducir un porcentaje entre 0 y 100
2	2	survey-text	Imagina que tiramos un dado de veinte caras 100 veces. En promedio, de estas 100 tiradas, ¿cuántas veces crees que saldría un número par en este dado de veinte caras? _____ de 100 tiradas.	1 != 50	number	0, 100	Tienes que introducir un porcentaje entre 0 y 100
3	3	survey-text	Imagina que tiramos un dado trucado de diez caras. La probabilidad de que salga un 10 al tirar el dado es el doble que la probabilidad de que salga uno de los demás números. En promedio, en 200 tiradas, ¿cuántas veces crees que saldría el número 10? _____ de 10 tiradas.	1 == 40	number	0, 200	Tienes que introducir un porcentaje entre 0 y 200
4	4	survey-text	En una huerta, el 10% de los pimientos son amarillos, el 20% son verdes y el 70% son verdes. La probabilidad de que un pimiento roja sea picante es del 10%. La probabilidad de que un pimiento que no sea rojo sea picante es del 5%. ¿Cuál es la probabilidad de que en el huerto un pimiento picante sea rojo?	3 != 80	number	0, 100	Tienes que introducir un porcentaje entre 0 y 100

3.1 Available tasks

In 2026-05-19 we have 68 tasks implemented, and 51 in development. The full details about the available tasks can be checked in this document. You can always check the full list of tasks in the Github repo.

To list the available tasks in the jsPsychMaker R package in you console, you can use the `list_available_tasks()` function. If you don't have the `jsPsychMaker` package, install it first.

```
jsPsychMaker::list_available_tasks()$tasks
```

```
  [1] "ACE"                 "AIM"                 "AntiBots"
  [4] "APrioriDiagnostic"   "APrioriScreening"    "Bank"
  [7] "BART"                "BDI"                 "BNT"
 [10] "bRCOPE"              "CAMIR"               "CAS"
 [13] "CDRISC10"            "CEL"                 "CIT"
 [16] "CMApost"             "CMApre"              "Consent"
 [19] "ConsentAudio"        "ConsentHTML"         "Cov19Q"
 [22] "COVIDCONTROL"        "CRQ"                 "CRS"
 [25] "CRT7"                "CRTMCQ4"             "CRTv"
 [28] "CS"                  "CTS2"                "CTT"
 [31] "DASS21"              "DEBRIEF"             "DEMOGR"
 [34] "DES"                 "DGLS"                "DMW"
 [37] "EAR"                 "ECRRS"               "EmpaTom"
 [40] "EQ"                  "ERQ"                 "ESM"
 [43] "ESV"                 "ESZ"                 "fauxPasEv"
 [46] "FDMQ"                "FKEA"                "GASP"
 [49] "GBS"                 "GHQ12"               "Goodbye"
 [52] "HRPVB"               "HRPVBpost"           "IBT"
 [55] "ICvsID"              "IDQ"                 "IEC"
 [58] "INFCONS"             "IRI"                 "IRS"
 [61] "ITQ"                 "LoB"                 "LOT"
 [64] "LSNS"                "MAIA"                "MAIAY"
 [67] "MCQ30"               "MDDF"                "MDMQ"
 [70] "MIS"                 "NARS"                "OBJNUM"
 [73] "OTRASRELIG"          "PBSr"                "PERMA"
 [76] "PPD"                 "PRFBM"               "PRFBMpost"
 [79] "PSC"                 "PSETPP"              "PSPPC"
 [82] "PSS"                 "PVC"                 "PWb"
 [85] "REI40"               "Report"              "RMET"
 [88] "RobToM"              "RSS"                 "RTS"
 [91] "SASS"                "SBS"                 "SCGT"
 [94] "SCSORF"              "SDG"                 "SDQ20"
 [97] "SILS"                "SPM2"                "sProQOL"
[100] "SRA"                 "SRBQP"               "SRSav"
[103] "STAI"                "STAIC"               "SWBQ"
[106] "UCLA"                "WaisMatrices"        "WaisMatricesES"
[109] "WaisWorkingMemory"   "WaisWorkingMemoryES" "WEBEXEC"
```

If you need help creating a NEW task, see the section help creating a new task.

Below, a table with an overview of the available tasks in the Google Sheet. If a task is here but not in the R package, you can open an issue to let us know.

short_name	Nombre
AIM	Grupos Socieeconómicos
BART	Balloon Analogue Risk Task
BNT	Berlin Numeracy test
Bank	Detalles bancarios
CAS	COVID-19 anxiety scale
CEL	Cuestionario Estilos de Liderazgo
CIT	Test de inferencia contrafactual
COVIDCONTROL	Covid related Questions
CRS	The Centrality of Religiosity Scale
CRT7	Cognitive reflection test
CRTMCQ4	Cognitive reflection test - 4 alternatives
CRTv	Verbal Cognitive Reflection Test
CS	Escala de Compasión
Consent	Consentimiento Informado de Participación
ConsentHTML	Consentimiento Informado de Participación
Cov19Q	Cuestionario Covid 19
DASS21	Escalas de depresión ansiedad y estrés
DEBRIEF	Debrief at end of experiment
DEMOGR	Demographic scale
EAR	Escala de Autoestima de Rosenberg
ERQ	Emotion Regulation Questionnaire
ESM	Escala Subjetiva de Memoria
ESV	Empathy Stimulus Validation
ESZ	Escala abreviada de Sobrecarga del Cuidador de Zarit
EmpaTom	Empatia, Teoria de la mente y Compasión
GBS	Global Beliefs Screening
GHQ12	12-item General health questionnaire
Goodbye	Goodbye task
HRPVB	High risk perception of vaginal birth
HRPVBpost	High risk perception of vaginal birth (POST)
IBT	Impulsive buying Tendency
ICvsID	Impartial Charity vs. Instrumental Damage
IDQ	Cuestionario de Identificación
IEC	Rotters internal-External control scale

short_name	Nombre
INFCONS	Cesarean information brochures
IRI	Interpersonal reactivity index
IRS	Importance of Rationality Scale
MDDF	MORAL DISGUST DUMBFOUNDING
MDMQ	Melbourne Decision Making Questionnaire
MIS	Magical ideation scale
OBJNUM	Lipkus numeracy Test
OTRASRELIG	Varias preguntas sobre creencias religiosas
PBSr	Paranormal Belief Scale
PERMA	Perma Profiler
PRFBM	Preferencia of birth mode in normal pregnancy
PRFBMpost	Preferencia of birth mode in normal pregnancy (post)
PSC	Prosociality
PSETPP	Percepciones sobre el embarazo, trabajo de parto y parto (Fear of birth scale)
PSPPC	Percepciones sobre el parto por cesárea
PSS	Perceived Stress Scale
PVC	Preguntas vacuna COVID-19
PWb	Psychological well being-scale
REI40	Rational-Experiential intentory-40
RTS	The Revised Transliminality Scale
Report	Tarea para solicitar autorización y detalles para enviar reporte automatizado con result
SASS	Social Adaptation Self-evaluation Scale
SBS	Supernatural Belief Scale
SCSORF	Santa Clara Strength of Religious Faith Questionnaire
SDG	Socio Demográfico General
SILS	School Implementation Leadership Scale
SRA	Self-Reported Altruism Scale
SRBQP	Self-regulation and belief questionnaire in pandemic
SRSav	Springfield Religiosity Scale (Abbreviated Version)
STAI	State-Trait Anxiety Inventory
SWBQ	Spiritual wellbeing questionnaire
WEBEXEC	Web based executive funtion questionnaire
bRCOPE	Escala de afrontamiento religioso Brief-RCOPE
sProQOL	Short Professional Quality of Life Scale

3.2 Experiment configuration

In the `config.js` file you can find the main parameters to control how your experiment works.

You can edit the config file in one of the following two ways:

A) Go to `folder_output` and edit `config.js`.

B) Use the jsPsychMaker_config Shiny APP and copy the generated config.js file in the `folder_output` folder.

If you use the app, you will need to copy the generated config.js file to your protocol folder. The Shiny app can also help you create a parametrized consent form (see the Consent tab).

3.2.1 Main parameters

- `pid = 999;`: Number of protocol
- `online = true;`: true if the protocol runs in a server, false if it runs locally
- `max_participants = 3;`: If you have `between participants` conditions (participants are assigned to only one of a number of conditions), this is the max number of participants per condition
- `random_id = false;`: true if you want to assign a random id to participants, false if the participant needs to input an id
- `max_time = "24:00:00";`: Maximum time to complete the protocol (HH:MM:SS; Hours:Minutes:Seconds)
- `accept_discarded = true;`: If a participant is discarded (i.e. exceeded the max_time), should we allow them to continue, given there are available slots?
- `debug_mode = false;`: When testing the protocol:
- shows DEBUG messages
- creates the DB tables if they don't exist
- Avoids randomization (e.g. order of items) so the `jsPsychMonkeys` can have a reproducible behavior
- `language = "English";` Language to use for the protocol. Either Spanish or English

3.2.2 Order of tasks

```
first_tasks = ['Consent'];//  The protocol will start with these tasks in sequential order
last_tasks = ['Goodbye'];//  Last block of tasks presented (in sequential order)
```

Create as many blocks as needed:

```
randomly_ordered_tasks_1 = ['TASK1', 'TASK2']; //  Block of tasks in random order
randomly_ordered_tasks_2 = ['TASK3']; //  Block of tasks in random order
secuentially_ordered_tasks_1 = ['TASK5', 'TASK4']; // Block of tasks in sequential order
```

The final array of tasks can be build combining the above blocks. The order of the tasks in the arrays starting with "random" will be randomized.

```
tasks = ['first_tasks',
         'randomly_ordered_tasks_1',
         'secuentially_ordered_tasks_1',
         'randomly_ordered_tasks_2',
         'last_tasks'];
```

3.2.3 Between-subject tasks

The variable `all_conditions` in `config.js` let's you define the Independent Variables (IV) and levels for the between-subject tasks:

If there is no between-subject task:

```
all_conditions = {"protocol": {"type": ["survey"]}};
```

If there are between-subject tasks:

```
all_conditions = {"NAMETASK": {"name_IV": ["name_level1", "name_level2"]}};
```

`jsPsychR` will randomize participants to the different conditions keeping the unbalance between conditions to the minimum possible.

3.3 online-offline protocols

jsPsych uses standard web technologies (HTML, CSS y Javascript), so that protocols should run in any modern browser (updated, please). We recommend Google Chrome just because our test suite runs with Google Chrome, so we will catch its specific issues earlier.

3.3.1 Offline

If you want to run a protocol locally (on your computer, on a lab computer), you need to:

- set `online = false;` in the `config.js` file
- double click index.html

`jsPsychR` will use `IndexedDB` to store the participants' progress and balance between conditions. The output csv files will be Downloaded to the Download folder of the computer where the protocol runs.

3.3.1.1 CORS ERRORS

If any of the tasks imports an html file, the Offline protocol will give a CORS error.

There are ways to disable web security in your browser, but it **MUST only be done if your experiment computer runs offline**, otherwise you will be exposed to very bad things.

See how to run chrome disabling web security to avoid CORS error:

- `google-chrome --disable-web-security --user-data-dir="~/"`

3.3.2 Online

Tu run a protocol online, set `online = true;` in the `config.js` file. You will need a couple more things:

- MySQL running in your server
- A file `.secrets_mysql.php` with the content below
- Define the route to `.secrets_mysql.php` in `controllers/php/mysql.php`
  - `require_once '../../.secrets_mysql.php';`

– THIS FILE **MUST NOT** BE PUBLICLY VISIBLE FROM THE BROWSER

- **Upload the files to the server** :)

```
<?php

/* DO NOT UPLOAD TO PUBLIC REPO */

  $servername = "127.0.0.1";
  $username = "USERNAME OF THE DATABASE";
  $password = "PASSWORD OF THE DATABASE";
  $dbname = "NAME OF THE DB";

?>
```

`jsPsychR` will use `MySQL` to store the participants' progress and balance between conditions. The output csv files will be Downloaded in the `.data/` folder inside the protocol folder in the server.

Before launching the final experiment, make sure you start with a clean slate! That can be summarized in 3 simple steps:

1. Check the configuration for you experiment (`config.js`) and make sure all is well. Some of the critical bits are:

```
pid = 999; // SHOULD have your project ID!
online = true; // true is good
max_participants = 100; // Max participants per contition [number]
max_time = "24:00:00"; // Max time to complete the protocol [HH:MM:SS]
debug_mode = false; // SHOULD be false
```

2. Check that the `.data/` folder for your protocol is empty in the server. You will likely have remains of the piloting and Monkeys.
3. Clean up the MySQL data associated to your protocol.

```
SET @PID = 999; // HERE YOUR PROTOCOL ID!

delete from experimental_condition where id_protocol=@PID;
delete from user where id_protocol=@PID;
delete from user_condition where id_protocol=@PID;
delete from user_task where id_protocol=@PID;
delete from task where id_protocol=@PID;
delete from protocol where id_protocol=@PID;
```

You will most likely need help from the server admin to perform these steps.

3.4 Language

We started implementing the basic blocks to be able to switch a protocol's language from Spanish to English with the parameter `language` in the `config.js` file.

This will change the protocol's hardwired messages (see `config_messages.js`), and will use the desired version of the task, if available. So far we only prepared a handful of tasks in multiple languages.

An example of a multilingual task would be `Consent.js`.

We have a Translations block:

```
// Translations ------------------------------------------------------------

switch (language) {

  case "Spanish":

    Consent_000 = ['<p><left><b><big>Consentimiento informado</big></b><br /></p>'];
    Consent_001_choices = ['acepto participar', 'rechazo participar'];
    Consent_001_end = 'Gracias por tu tiempo. Puedes cerrar esta página.';
    break;

  case "English":

    Consent_000 = ['<p><left><b><big>Informed consent</big></b><br /></p>'];
    Consent_001_choices = ['I agree to participate', 'I reject to participate'];
    Consent_001_end = 'Thanks for your time. You can close this page.';
    break;

}
```

And then a Task block, where the logic of the experiment is unique, and we use the variables created in the Translation block for the things the users will see in their screens:

```
// Task --------------------------------------------------------------------

questions = ( typeof questions != 'undefined' && questions instanceof Array ) ? questions :
```

```
questions.push( check_fullscreen('Consent') );
Consent = [];    //temporal timeline


var instruction_screen_experiment = {
    type: 'instructions',
    pages: Consent_000,
    data: {trialid: 'Consent_000', procedure: 'Consent'},
    show_clickable_nav: true,
    on_trial_start: function(){
        bloquear_enter = 0;
    }
};


// Reads consent from media/consent/consent-placeholder.js
var question01 = {
  type: 'html-button-response',
  stimulus: intro_CONSENT,
  choices: Consent_001_choices,
  prompt: "<BR><BR>",
  // If 'rechazo participar' is pressed, end experiment
  on_finish: function(data){
    if(jsPsych.data.get().values().find(x => x.trialid === 'Consent_001').button_pressed == 1
      jsPsych.endExperiment(Consent_001_end);
    }
  },
    data: {
    trialid: 'Consent_001',
    procedure: 'Consent'
   }
};
Consent.push(question01);


Consent.unshift(instruction_screen_experiment);
Consent.push.apply(questions, Consent);
call_function("Consent");
```

3.5 Need help implementing a task!

If you need help creating a NEW task, see the section help creating a new task.

3.6 Developing tasks

Remember to place an `if (debug_mode === false)` before the randomization of the item order so when running in debug_mode, the items are not randomized. This is important so the behaviour of the jsPsychMonkeys is reproducible:

```
if (debug_mode === false) NAMETASK = jsPsych.randomization.repeat(NAMETASK,1);
```

3.7 Technical aspects

We currently use jsPsych 6.3 as the default, and started to implement the last stable jsPsych (7.3) recently. To choose a version for the protocols you create, use the parameter `jsPsych_version`. For example, `jsPsych_version = 7`. Things are not fully tested in the v7 so, use with care.

There is a migration guide and a Github issue with migration questions.

3.7.1 Misc

When index.html is launched:

- Checks if there are available slots

When an uid is assigned:

- `questions` array is created
- `between-participants` conditions are assigned and stored in the DB (MySQL if online, IndexedDB if offline)

Each question, timeline or conditional question needs to have a:

```
data: {trialid: 'NameTask_001', procedure: 'NameTask'}
```

The `trialid` identifies the trial, and the `procedure` makes possible to find that trial so participants can continue the tasks where they left, know when participants finished the tasks, etc. This is done in MySQL if online, IndexedDB if offline.

When running online tasks with between-participants variables, the system that balances conditions can change between-participants condition after the participants accept the consent form. If any of the items is not in a timeline (e.g. Instructions) and stores the `condition_between`, it may not be up to date. See Bayesian39 task for an example.

`trialid`'s need to have a standardized structure, which generally conforms with `NameTask_3DigitNumber`. When using conditional items the structure can be a bit more complex, but not much. We use the following rules to check for non-complying trialid's:

```
^[a-zA-Z0-9]{1,100}_[0-9]{2,3}$ -> `NameTask_2or3DigitNumber`, for example `BNT_001`
^[a-zA-Z0-9]{1,100}_[0-9]{2,3}_[0-9]{1,3}$ -> `NameTask_2or3DigitNumber_1to3DigitsSuffix`, fo
^[a-zA-Z0-9]{1,100}_[0-9]{2,3}_if$ -> `NameTask_2or3DigitNumber`, for example `BNT_002_if`
^[a-zA-Z0-9]{1,100}_[0-9]{2,3}_[0-9]{1,3}_if$  -> `NameTask_2or3DigitNumber`, for example `BI
```

3.7.2 jsPsychMaker main changes on a task

1. Start of a task

```
questions = ( typeof questions != 'undefined' && questions instanceof Array ) ? questions :
questions.push( check_fullscreen('NameOfTask') );
NameOfTask = [];
```

2. Each item

```
data: {trialid: 'NameOfTask_01', procedure: 'NameOfTask'}
```

3. End of experiment

```
if (debug_mode == 'false') NameOfTask = jsPsych.randomization.repeat(NameOfTask, 1);
NameOfTask.unshift(instruction_screen_experiment);
questions.push.apply(questions, NameOfTask)

questions.push({
    type: 'call-function',
    data: {trialid: 'NameOfTask_000', procedure: 'NameOfTask'},
    func: function(){
      if (online) {
```

```
        var data = jsPsych.data.get().filter({procedure: 'NameOfTask'}).csv();
      } else {
        var data = jsPsych.data.get().filter({procedure: 'NameOfTask'}).json();
      }
      saveData(data, online, 'NameOfTask');
    }
});
```

3.7.3 Conditional questions

```
var question001 = {
  type: 'survey-multi-choice-vertical',
  questions: [{prompt: '<div class="justified">¿Usted se ha vacunado contra el coronavirus /
  data: {trialid: 'PVC_001', procedure: 'PVC'}
};
PVC.push(question001);

var question001_1 = {
  type: 'survey-multi-choice-vertical',
  questions: [{prompt: '<div class="justified">¿Usted se va a vacunar contra el coronavirus
  data: {trialid: 'PVC_001_1', procedure: 'PVC'}
};

var if_question001_1 = {
  timeline: [question001_1],
  data: {trialid: 'PVC_001_1_if', procedure: 'PVC'},
  conditional_function: function(){
    let data = (JSON.parse((jsPsych.data.get().values().find(x => x.trialid === 'PVC_001'))[
    if((data) ==  'No'){
      return true;
    } else {
      return false;
    }
  }
};
PVC.push(if_question001_1);
```

3.8 Common ERRORS

If you get the following error in the console: `Uncaught TypeError: Cannot read properties of undefined (reading 'procedure')`

Run this in the console:

```
for (var i = 0; i < questions.length; i++) {
  console.log(i + questions[i].data["procedure"])
}
```

It will stop in one of the items. Go to the console, check the array `questions` and go to the number that failed.

When you know the task and item that fails, you probably need to add:

```
`data: {trialid: 'TASKNAME_ITEMNUMBER', procedure: 'TASKNAME'}
```

4 jsPsychMonkeys

---

jsPsychMonkeys: Release Monkeys to a jsPsych experiment using the R package {targets}, docker and {RSelenium}.

In brief

With jsPsychMonkeys you can:

- Simulate participants online and offline
- Simulate participants sequentially and in parallel
- Ask your Monkeys to take pictures of each screen of the protocol
- Make the behavior of the Monkeys reproducible setting a random seed associated with their unique id
- Store logs of the process, including console logs with errors
- Watch your participants randomly click things in VNC (you will need to install realvnc)

See QuickGuide for basic instructions.

Setup

First, install Docker Desktop.

Ubuntu

You may need to install some system libraries first:

- `sudo apt install libssl-dev libcurl4-openssl-dev libxml2-dev docker`
- If the Monkeys do their work but no csv's appear, make sure your the docker user has write access to the `~/Downloads` folder.

In some versions of Ubuntu, you do not need to install docker. If anything fails, installing Docker desktop and leaving it open could help. Alternatively, maybe you need docker as non-root user?, but be careful, as this leaves your system more vulnerable.

Windows

- Install docker desktop
- Update wsl (in a command prompt): `wsl - update`

Mac

- Install docker desktop

4.1 How to simulate participants

If you are on Windows, make sure Docker Desktop is open and running before releasing the monkeys.

To run a monkey locally:

```
jsPsychMonkeys::release_the_monkeys(uid = 1,
                                    local_folder_tasks = "~/Downloads/protocol999/")
```

To run a monkey on a server:

```
jsPsychMonkeys::release_the_monkeys(uid = 1,
                                    server_folder_tasks = "999",
                                    credentials_folder = "~/.vault/")
```

`credentials_folder` must contain `SERVER_PATH.R` and `.credentials`. See below for the expected content of those files.

4.2 Parameters available

There are a few parameters for `jsPsychMonkeys::release_the_monkeys()` that can be useful:

- `uid_URL = TRUE`: The uid is passed in the URL (e.g. `&uid=1`)
- `local_folder_tasks = rep("Downloads/tests/test_prototol", 25)`: Passing a vector of multiple protocols will make the Monkeys to complete all of them.
- `times_repeat_protocol`: How many times a monkey should complete the same protocol (useful for longitudinal protocols or to speed up things)
- `time_to_sleep_before_repeating_protocol`: How many seconds to wait before reattempting to complete the protocol
- `keep_alive = TRUE` Keep the docker container alive after completing the tasks
- `DEBUG = TRUE` Activate DEBUG mode. Lot's of stuff will show up in the console.
- `open_VNC = TRUE` Activate DEBUG mode and open a VNC container to see the Monkeys' progress.
- `screenshot = TRUE` The Monkeys will take a picture of all the pages they see. The .png files are stored in `outputs/screenshots`
- `debug_file = TRUE` Activate DEBUG mode and store all the console output in the `outputs/log`
- `big_container = TRUE` Sets the Shared memory size (/dev/shm) to 2 gigabytes. This is useful to avoid long/complex protocols to crash
- `disable_web_security = TRUE` If you are running a local protocol that loads external files (e.g. consent form in a html file), you may need this. Only works with Google Chrome.
- `console_logs = TRUE` Store the browser's console logs. Only works with Google Chrome
- `forced_random_wait = TRUE` Will wait a randomly sampled number of seconds on page 4
- `forced_seed = 11` Set a random seed so the Monkeys' behavior will be fully reproducible
- `forced_refresh = 20` Refresh browser in page 20 (if TRUE is given, it will refresh in a randomly sampled page)
- `sequential_parallel` Choose between `sequential`, the default, or `parallel`
- `number_of_cores` Number of cores for parallel monkeys. The default is half of the available cores

4.2.1 Parameters details

- `local_folder_tasks`: If the folder is not accessible to Docker (anything outside the Download folder), jsPsychMonkeys will create a copy of the protocol in `Downloads/JSPSYCH/`

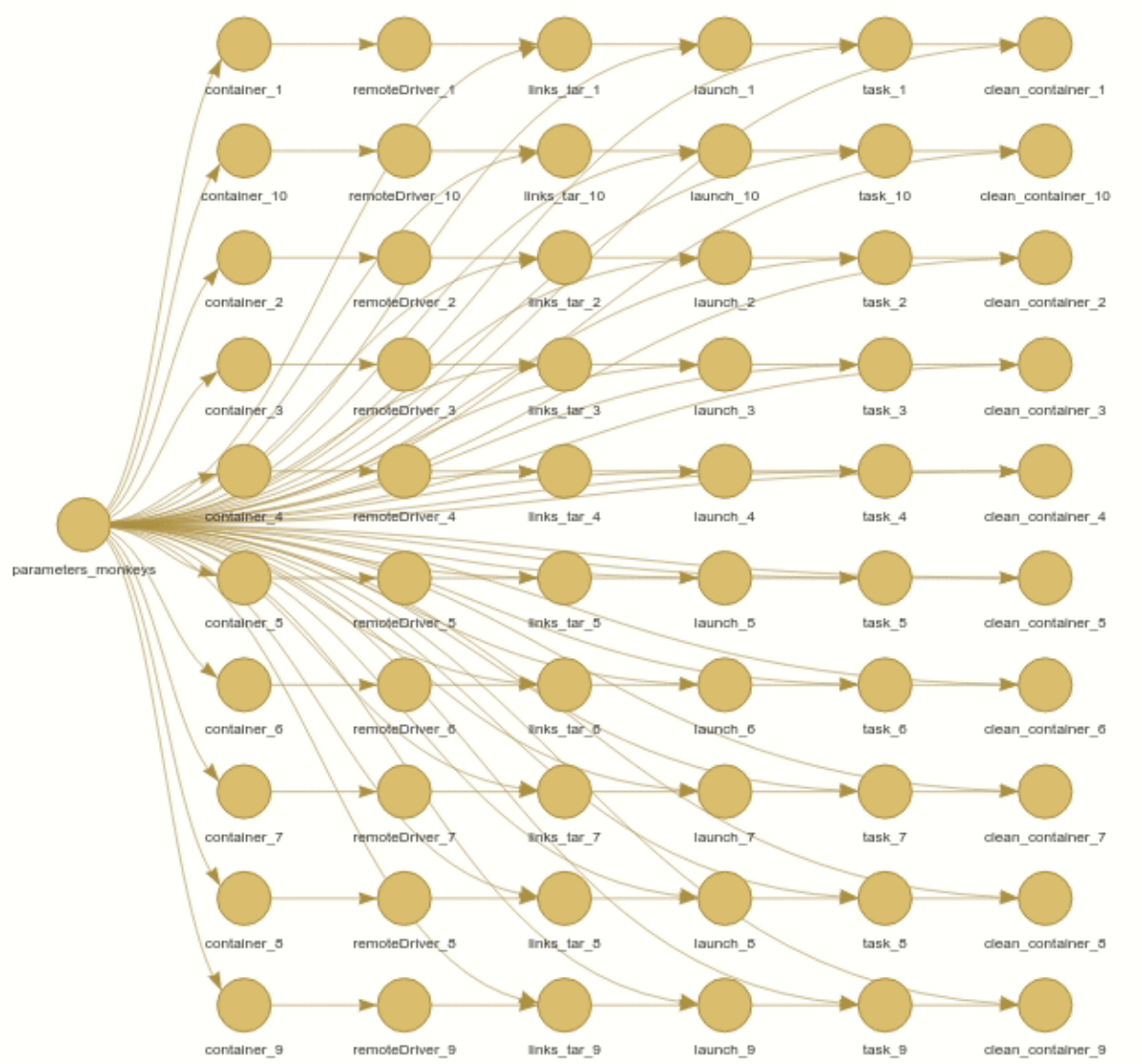
4.3 Release a horde of Monkeys!

If you want a horde of Monkeys, you can set up `sequential_parallel = "parallel"` and choose how many monkeys will run in parallel with `number_of_cores`:

```
jsPsychMonkeys::release_the_monkeys(uid = "1",
                                    local_folder_tasks = "~/Downloads/protocol999/",
                                    sequential_parallel = "parallel",
                                    number_of_cores = 4)
```

---

10 Monkeys completing a protocol in parallel:



4.4 Issues

If the setup configuration steps didn't work... You may need to do one of the things below:

- Switch to Ubuntu :-)
- Run participants manually

4.5 Technical aspects

4.5.1 Launch Monkeys on a server

You will need two files for the configuration in the hidden and NOT SHARED `.vault/` folder:

- `.vault/SERVER_PATH.R`: contains the path where the protocols are located in your server: `server_path = "http://URL_OF_YOUR_SERVER/PROTOCOLS_GENERAL_FOLDER/"`
- `.vault/.credentials`: contains a list with the user and password for the server:

```
list(IP = "IP ADDRESS OF SERVER",
     main_FOLDER = "/MAIN/FOLDER/AND/PATH/TO/FILES/IN/THE/SERVER/",
     user = "YOUR SERVER USERNAME",
     password = "YOUR VERY STRONG SERVER PASSWORD")
```

With the `server_folder_tasks` you will set the sub-folder where the protocol is located. In the example below the Monkeys would go to, `http://URL_OF_YOUR_SERVER/PROTOCOLS_GENERAL_FOLDER/999`

4.5.2 Alternatives

Since jsPsych 7.1 there is a simulation mode available, which should be much faster than the good ol' Monkeys. Once we migrate to jsPSych 7.x, we may retire this section.

5 jsPsychHelpeR

---

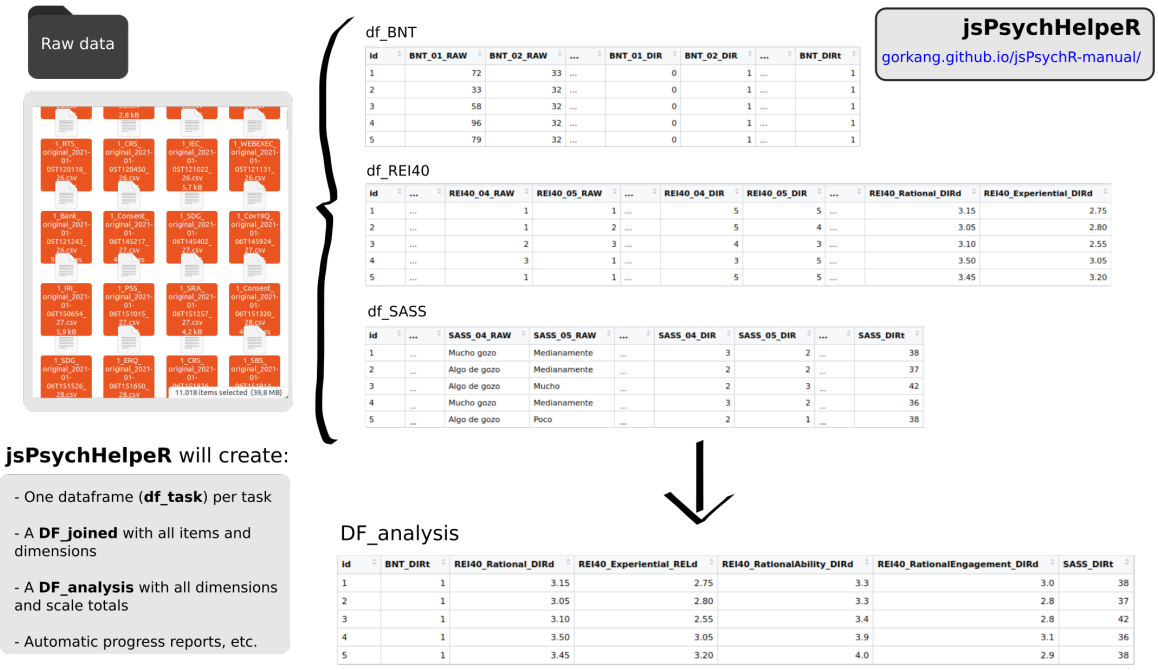
jsPsychHelpeR: Standardize and automatize data preparation and analysis of jsPsych experiments created with jsPsychMaker.

In brief

jsPsychHelpeR will lend you a hand automatizing and standardizing your data preparation and analysis.

- Use a completely open, reproducible and automatic process to prepare your data
- Data preparation ready for > 70 tasks (see here the available tasks)
- Get tidy output dataframes for each task, and for the whole protocol
- Include tests for common issues
- Automatic reports with progress, descriptives, codebook, etc.
- Create a fully reproducible Docker container with the project's data preparation and analysis
- Create a blinded dataframe to be able to perform blinded analyses. See 10.1038/526187a and 10.1177/25152459221128319

See QuickGuide for basic instructions.



5.1 How to prepare data

Our goal is that each jsPsychMaker task has a sister script on jsPsychHelpeR to help prepare the data automatically. If a task you need does not have one, you can try to create the script yourself and do a pull request in the jsPsychHelpeR repo, or see the section help creating a new task.

If you already ran a pilot experiment, simply:

1. Install `jsPsychHelpeR`:

```
if (!require('remotes')) utils::install.packages('remotes'); remotes::install_github('gorkang
```

2. `jsPsychHelpeR::run_initial_setup()` will:

- Try to make sure you have all the dependencies, folders, etc.
- Copy the data to the data/pid folder
- Move data with sensitive tasks to the .vault folder

- Create a customized `_targets.R` file adapted to the data of your protocol, so data preparation can run automagically

```
jsPsychHelpeR::run_initial_setup(pid = '999',
                                 data_location = '~/Downloads/JSPSYCH/999/',
                                 folder = '~/Downloads/jsPsychHelpeR_999/')
```

If you have the sFTP credentials for the server, it will:

- Download all the data from your protocol (you will need the FTP credentials and set `download_files = TRUE`)

- Download and zip a copy of the full protocol without the data (you will need the FTP credentials and set `download_task_script = TRUE`)

This should work on Ubuntu, if you have the FTP credentials, and `sshpass` and `rsync` installed.

```
jsPsychHelpeR::run_initial_setup(pid = '999',
                                 download_files = TRUE,
                                 download_task_script = TRUE,
                                 folder = "~/Downloads/jsPsychHelpeR_999")
```

5.1.1 Targets pipeline

To help make the pipeline reproducible and more efficient, we use the targets package (Landau 2021b). A few basic things to know:

- **The whole process can be reproduced running `targets::tar_make()`**
- A nice visualization of all the pre-processing steps can be seen with `targets::tar_visnetwork(targets_o = TRUE)`
- The file `_targets.R` contains the important parameters and calls to all the functions used when running `targets::tar_make()`

To see more detail about any specific step, you can:

1. Go to the relevant function in `_targets.R` (cursor on a function, then F2)

2. Load the input parameters of the function with `debug_function(NAME_OF FUNCTION)`. Alternatively, manually use `targets::tar_load(NAME_OF_TARGET)`
3. Run the code step by step as you would normally do

5.2 Basics

jsPsychHelpeR uses as input a data created with a jsPsychMaker experimental protocol.

5.2.1 Inputs

The input data folder will be named after the protocol_id, for example `999/` and needs to be placed in the `data/` folder of the jsPsychHelpeR project `data/YOUR_PROJECT_NUMBER`:

- The data folder can contain either multiple .csv files, or a single .zip file

There will be a single .csv file for each participant and task of the protocol. For example:

- 999_Consent_original_2022-04-02T205622_1.csv:
  - [project: 999]_[experimento: Consent]_[version: original]_[datetime: 2022-04-02T205622]_[participant id: 1]

5.2.2 Outputs

When the pipeline successfully runs with `targets::tar_make()`, a number of outputs will be created.

All the outputs can be found in the `/outputs` folder. The only exception is the sensitive data and reports, which can be found in `.vault/outputs`. WARNING: The '.vault/' folder **MUST NOT** be made public.

5.2.2.1 Output folders

The outputs will be organized in different folders:

- **Data frames** for different stages of data processing can be found in `outputs/data`
- **Temporary files for manual correction** are in `outputs/data/manual_correction` (the final manual correction files must be place by the user in `data/manual_correction`). WARNING: These will be overwritten each time the pipeline runs

- **Plots**, **tables** and **reports** are in `outputs/plots`, `outputs/tables`and `outputs/reports` respectively.
- **Test outputs** are in `outputs/tests_outputs`
- **Anonymized Raw data** will be moved to `.vault/data_vault/`

5.2.2.2 Output dataframes

There will be a single data frame (df) for each of the tasks in `outputs/data`, plus a data frame (DF) for each of the steps of the data preparation, and a dictionary file listing all the available tasks. We store the files in two formats, csv and rds:

- **DF_raw.csv**: All the `data/project_id/` csv files combined on a single file. We only add the columns "project", "experimento", "version", "datetime", "id" by parsing the filenames
- **DF_clean.csv**: Clean version of the raw file ready to process the individual tasks
- **df_ShortNameOfTask.csv**: One df for each of the tasks of the protocol after being processed with the `prepare_ShortNameOfTask()` functions
- **DF_joined.csv**: all the processed tasks joined in a single DF
- **DF_analysis**: only the total scores and dimensions from `DF_joined` (columns ending in `_DIRt`, `_STDt`, `_DIRd`, `_RELd`, `STDd`). Can be visually explored using the shiny app in `Rmd/app.R`
- **DF_analysis_blinded**: If the `DVars` parameter of `create_DF_analysis()` is not empty, jsPsychHelpeR will create DF_analysis_blinded where the DVars will be scrambled so te data analysts can perform blinded analysis
- **DICCIONARY_tasks.csv**: list of all tasks in the protocol

5.2.2.3 Output dataframes column names

All the output processed data frames columns are named in a standardized way:

- **ShortNameOfTask_ItemNumber_RAW**: raw responses of participants for individual items
- **ShortNameOfTask_ItemNumber_DIR**: processed raw responses following the task correction instructions (e.g. inverting certain items, converting strings to numbers, computing accuracy…)
- **ShortNameOfTask_RAW_NA**: number of missing data (NA) in the RAW responses

- **ShortNameOfTask_DIR_NA**: number of missing data (NA) in the DIR responses. If it is not equal to `ShortNameOfTask_RAW_NA` there is something wrong in the items correction.
- **ShortNameOfTask_DimensionName_DIRd**: scores for a specific dimension (`d`) in a task, calculated following task correction instructions (e.g. summing or averaging certain items)
- **ShortNameOfTask_DimensionName_RELd**: scores for a specific dimension (`d`) in a task, calculated following task correction instructions AND after filtering items with low reliability. See Reliability section for more information.
- **ShortNameOfTask_DimensionName_STDd**: standardized score for a dimension (`d`)
- **ShortNameOfTask_DIRt**: total (`t`) score for a task calculated following task correction instructions (e.g. summing or averaging all items)
- **ShortNameOfTask_STDt**: standardized (`t`) score for a task

5.3 Errors in the pipeline

See the targets manual for more information.

We include `tar_option_set(workspace_on_error = TRUE)` in `_targets_options.R` so if there is an error in our pipeline, `targets` will automatically save the workspace. This allows you to go to the relevant target and debug interactivelly.

If you get an error:

1. List the available workspaces (e.g. DF_clean):

```
tar_workspaces()
```

2. Load the errored workspace:

```
tar_workspace(DF_clean)
```

5.4 Advanced

5.4.1 Need help preparing new task

If you need help preparing a NEW task, see the section help with new tasks.

5.4.2 Create your own reports

You can use any of the template reports in the `_targets.R` file, or create your own reports.

We will start opening one of the template reports: `rstudioapi::navigateToFile("doc/report_analysis.Rmd"`

- Edit the RMarkdown file to adapt it to your needs.
- If you already did `targets::tar_make()`, when running `targets::tar_load(DF_analysis)` the dataframe `DF_analysis` will load in your Environment.

Go back to the `_targets.R` file:

- Look for `# Analysis report` and uncomment the following lines:

```
# tar_render(report_analysis, "doc/report_analysis.Rmd",
#            output_file = paste0("../outputs/reports/report_analysis.html")),
```

When you finished editing and uncomented the tar_render command, go back to the `run.R` file:

- `targets::tar_make()`

5.4.3 Create new tasks

To create the correction script for a new task, you start with:

- `create_new_task(short_name_task = "NAMETASK", get_info_googledoc = TRUE)`

This will:

- create a new file from a template correction script (`R_tasks/prepare_TEMPLATE.R`)
- adapt it to your `short_name_task` to make everything as standardized as possible
- open the new `prepare_NAMETASK.R` file

If the parameter `get_info_googledoc = TRUE`:

- The NEW tasks document is checked.
- If the document has been filled properly, it will show in the console standardized strings (ready to be copy/pasted to the new `prepare_NAMETASK.R` file) about:
  - dimension names
  - items corresponding to each dimension

- dimension calculation
- inverse items
- numeric conversion of items

You can also use `get_dimensions_googledoc()` as a standalone function:

```
get_dimensions_googledoc(short_name_text = "MLQ")
```

All the `prepare_NAMEOFTASK.R` scripts on the `R_tasks/` folder have been created starting from the same template. The only exception are the experimental tasks and some surveys with particularities that require more complex adaptations.

When you finish implementing the correction script, please do a Pull request so we can add you script to the pool. If you have not already, please help us filling up details about the task in the NEW tasks document.

5.4.4 Adapting new tasks

`get_dimensions_googledoc` will show you how to adapt the `prepare_TASK()` script, but you will need to know how it works to be able to edit the relevant bits. Also, sometimes `get_dimensions_googledoc` won't get all the details of the task right, or there could be non-standard elements to it. Here, we will describe some of the elements of the template to help understand how it works.

Remember you should **ALWAYS** start with `create_new_task(short_name_task = "NAMETASK")` so your task template works well with `jsPsychHelpeR`.

There are three chunks you will need to adapt to have a fully working preparation script.

- `[ADAPT 1/3]: Items to ignore and reverse, dimensions`
- `[ADAPT 2/3]: RAW to DIR for individual items`
- `[ADAPT 3/3]: Scales and dimensions calculations`

5.4.4.1 Items to ignore and reverse, dimensions

```
# [ADAPT 1/3]: Items to ignore and reverse, dimensions -----------------------
# ****************************************************************************

description_task = "" # Brief description here

items_to_ignore = c("000") # Ignore these items: If nothing to ignore, keep as is
items_to_reverse = c("000") # Reverse these items: If nothing to reverse, keep as is
```

```
## NameDimension1, NameDimension2 should be the names of the dimensions
## Inside each c() create a vector of the item numbers for the dimension
## Add lines as needed. If there are no dimensions, keep as is
items_dimensions = list(
  NameDimension1 = c("000"),
  NameDimension2 = c("000")
)

# [END ADAPT 1/3]: ************************************************************
# *****************************************************************************
```

5.4.4.2 RAW to DIR for individual items

```
DF_long_DIR =
    DF_long_RAW %>%
    select(id, trialid, RAW) %>%



  # [ADAPT 2/3]: RAW to DIR for individual items -------------------------------
  # *****************************************************************************

    # Transformations
    mutate(
      DIR =
        case_when(
          RAW == "Nunca" ~ 1,
          RAW == "Poco" ~ 2,
          RAW == "Medianamente" ~ 3,
          RAW == "Bastante" ~ 4,
          RAW == "Mucho" ~ 5,
          is.na(RAW) ~ NA_real_,
          grepl(items_to_ignore, trialid) ~ NA_real_,
          TRUE ~ 9999
        )
    ) %>%

    # Invert items
    mutate(
      DIR =
```

```
        case_when(
          DIR == 9999 ~ DIR, # To keep the missing values unchanged
          trialid %in% paste0(short_name_scale_str, "_", items_to_reverse) ~ (6 - DIR),
          TRUE ~ DIR
        )
    )

  # [END ADAPT 2/3]: ************************************************************
  # *****************************************************************************
```

5.4.4.3 Scales and dimensions calculations

```
# [ADAPT 3/3]: Scales and dimensions calculations ----------------------------
# *****************************************************************************

# Reliability ----------------------------------------------------------------
# REL1 = auto_reliability(DF_wide_RAW, short_name_scale = short_name_scale_str, items = items
# items_RELd1 = REL1$item_selection_string


# [USE STANDARD NAMES FOR Scales and dimensions: names_list$name_DIRd[1], names_list$name_DI
# CHECK with: create_formulas(type = "dimensions_DIR", functions = "sum", names_dimensions)
DF_wide_RAW_DIR =
  DF_wide_RAW %>%
  mutate(

    # [CHECK] Using correct formula? rowMeans() / rowSums()

    # Score Dimensions (see standardized_names(help_names = TRUE) for instructions)
    !!names_list$name_DIRd[1] := rowMeans(select(., paste0(short_name_scale_str, "_", items_
    !!names_list$name_DIRd[2] := rowSums(select(., paste0(short_name_scale_str, "_", items_di
    # Dimension should correspond to a specific item
    !!names_list$name_DIRd[3] := get(paste0(short_name_scale_str, "_", items_dimensions[[3]]


    # Reliability Dimensions (see standardized_names(help_names = TRUE) for instructions)
    # !!names_list$name_RELd[1] := rowMeans(select(., paste0(short_name_scale_str, "_", items

    # Score Scale
    !!names_list$name_DIRt := rowSums(select(., matches("_DIR$")), na.rm = TRUE)
```

```
  )

# [END ADAPT 3/3]: ************************************************************
# *****************************************************************************
```

5.4.5 Tasks with non-standard elements

There are a number of tasks that do not fit a simple *sum all the items* preparation pipeline. Here, a number of them that could be useful when encountering the same particularities again:

- AIM: Has an associated cvs file look-up file to score
- WaisMatrices: direct to standardized scores using a specific table
- ITQ, CAMIR, ...: other interesting issues solved

5.4.6 DEBUG tasks

At the begining of each of the `R_tasks/prepare_NAMETASK.R` scripts you will find a commented `debug_function(prepare_NAMETASK)` line.

When running it, it will load the input parameters for the task. From there, you can work inside of the preparation scipt as you would normally do in a R script.

If you get the error `"Error in debug_function(prepare_NAMETASK) : could not find function 'debug_function'`debug_function()`does nor work"` you will need to load all the functions in the `R/` folder first.

You can do this in one of three ways:

- `CONTROL + P` shortcut will work if the `run_initial_setup()` completed correctly (at least on Ubuntu systems).
- Run `targets::tar_load_globals()`
- Or directly, source all the scripts in the `R/` folder: `invisible(lapply(list.files("./R", full.names = TRUE, pattern = ".R$"), source))`

5.4.7 Docker containers

The function `jsPsychHelpeR::create_docker_container()` will create a fully reproducible docker container with the data preparation and analysis for a specific project.

The container can be easily shared or stored to allow others to run the data preparation and analysis for you project without worrying about dependencies, versions of packages, etc.

See more information about the setup in the admin section.

The gist of it is, after you have the full data preparation and analysis for you project ready, to create the container image and share it, just run:

```
# 1) Set your project ID
  PID = 999

# 2) Create docker image
  jsPsychHelpeR::create_docker_container(PID = PID)

# 3) SHARE your docker image

  # Using Dockerhub
    system(paste0("docker push gorkang/jspsychhelper:pid", PID))

  # Using a .tar file
    system(paste0("docker save gorkang/jspsychhelper:pid", PID, " | zip > pid", PID, ".tar.zi
```

To load and run the container image (if you are using Windows, see here):

```
# 1) Set your project ID
  PID = 999

# 2) Get the docker image loaded into to your computer

  # Dockerhub
  system(paste0("docker pull gorkang/jspsychhelper:pid", PID))

  # .tar file
  utils::unzip(zipfile = paste0("pid", PID, ".tar.zip"), files = paste0("-"))
  system(paste0("docker load --input -"))

# 3) Run docker container
  system(paste0("docker run --rm -d --name pid", PID, " -v ~/Downloads/jsPsychHelpeR", PID, "
```

The output will be in `Downloads/jsPsychHelpeR[PID]/outputs/` after a couple of minutes. You can see the data preparation and analysis progress using docker desktop.

5.4.8 Blinded analysis

The function `create_DF_analysis()` has the parameter `DVars` to select the Dependent Variables in your data that should be scrambled to be ready for a blinded analysis. We use a simple `sort()` in those variables, so their data will be ordered from smaller to bigger, losing the relationship with the other variables in the data, but keeping their structure.

See MacCoun, R., & Perlmutter, S. (2015). Blind analysis: Hide results to seek the truth. Nature, 526(7572), 187-189 (https://doi.org/10.1038/526187a), or Sarafoglou, A., Hoogeveen, S., & Wagenmakers, E. J. (2023). Comparing analysis blinding with preregistration in the many-analysts religion project. Advances in Methods and Practices in Psychological Science, 6(1), 25152459221128319. (https://doi.org/10.1177/25152459221128319)

5.5 Helper functions

5.5.1 Reliability

You can use the `auto_reliability()` function to help you automatically filter items with low reliability (although doing this automatically is probably a bad idea). The function uses `psych::alpha()` and filters by default items with an r.drop <= 0.2. See `psych::alpha()` help for more details. **IMPORTANT**: Using `psych::omega()` is generally a better idea, see the alpha help page.

An example can be found in prepare_REI40().

The basic logic would be:

```
# Define items for a specific dimension
items_DIRd1 = c("01", "02", "03", "04", "05", "06", "07", "08", "09", "10")

# Calculate reliability
REL1 = auto_reliability(DF_wide_RAW, short_name_scale = short_name_scale_str, items = items_

# Store item selection in a variable
items_RELd1 = REL1$item_selection_string

# In the final Dimension calculation, use the item selection including only the items with a
## See `items_RELd1` below
!!names_list$name_RELd[1] := rowMeans(select(., paste0(short_name_scale_str, "_", items_RELd
```

```
# Compare it with the calculation including the original items
## See `items_DIRd1` below
!!names_list$name_DIRd[1] := rowMeans(select(., paste0(short_name_scale_str, "_", items_DIRd
```

5.6 Technical aspects

5.6.1 How trialid's are processed

See `PRFBM`:

- If more than one response per screen
  - Item: `PRFBM_04`
  - Responses: `{"daño":"Parcialmente en desacuerdo","beneficio":"Parcialmente en desacuerdo"}`
  - final trialids: `PRFBM_04_beneficio` and `PRFBM_04_daño`

5.7 Common ERRORS

5.7.1 `run_initial_setup()`:

```
x Can find server credentials in '.vault/.credentials'
x 0 tasks found for protocol 'TU NUMERO DE PROYECTO'. NOT creating _targets.R file
```

5.7.1.1 On Linux (Ubuntu):

- IF you have the server credentials:
  - Open .credentials_TEMPLATE `rstudioapi::navigateToFile(".vault/.credentials_TEMPLATE`
  - Edit the file with your server credentials
  - Rename the file to `.credentials`

- IF you DON'T have the credentials but you have the .csv results files:
  - Copy the csv files to the folder `data/YOUR_PROJECT_NUMBER`
  - Run again `run_initial_setup()`

5.7.1.2 On Mac or Windows:

- Copy the csv files to the folder `data/YOUR_PROJECT_NUMBER`
- Run again `run_initial_setup()`

5.7.2 Rendering Rmd's

> Error: ! missing files _targets/meta/meta Execution halted

It is better to run everything, including your reports, inside the pipeline (`targets::tar_make()`).

If you need to the knitr (or render) button, you will have to:

- Load DF's `DF_analysis = readr::read_rds(here::here("_targets/objects/DF_analysis"))` instead of `targets::tar_load(DF_analysis)`
- Include all the necessary `library()` calls

> Error : Could not parse knitr report Rmd/report_analysis.Rmd to detect dependencies: Scanner error: mapping values are not allowed in this context at line 6, column 17

There is something wrong in Your YAML heather.

6 jsPsychR Admins

---

jsPsychAdmin: Functions to help with common administrative tasks for jsPsychR protocols. The goal is to minimize issues and make sure tasks behave in a consistent and reproducible manner.

- Install with:

```
if (!require('pak')) utils::install.packages('pak'); pak::pkg_install("gorkang/jsPsychAdmin")
```

You will need a `.vault` folder with `.credentials`, `data_encrypted.rds` and `data_public_key.txt`.

6.0.1 Common tasks

6.0.1.1 Debug functions

Prints usage and examples, shows help, and loads usage parameters to Global environment.

```
get_parameters_of_function(name_function = "jsPsychMaker::create_protocol()",
                           load_parameters = TRUE)
```

6.0.1.2 Sync and Check all protocols

```
  # Syncs and checks all
  jsPsychAdmin::download_check_all_protocols(gmail_account = "XYZ@gmail.com",
                                             # Overwrites existing files that changed
                                             ignore_existing = FALSE,
                                             dont_ask = TRUE,
                                             show_all_messages = TRUE)
```

```
# Just syncs the protocols
jsPsychHelpeR::sync_server_local(server_folder = "",
                                 local_folder = here::here(paste0("..", "/CSCN-server/proto
                                 direction = "server_to_local",
                                 only_test = FALSE,
                                 exclude_csv = TRUE, # DO NOT INCLUDE DATA
                                 delete_nonexistent = TRUE,
                                 # If a file already exists, ignore it. Good for data, bad
                                 ignore_existing = FALSE,
                                 dont_ask = TRUE
                                 )
```

6.0.1.3 CHECK participants all protocols

```
# Uses the .credentials file + the public key to unlock the encrypted data_encrypted.rds
jsPsychAdmin::check_status_participants_protocol()
```

6.0.1.4 Clean up a DEV protocol

```
# Clean up DB and csv files for a test/protocols_DEV/ protocol # Useful when testing
# rstudioapi::navigateToFile(".vault/.credentials")
# Asks for server password
jsPsychAdmin::clean_up_dev_protocol(protocol_id = "test/protocols_DEV/31")
```

6.0.1.5 Check missing scripts

```
#  Downloads all the protocols to CSCN-server
#  Then checks:
# - for which ones we do not have preparation scripts
# - for which ones we do not have googledoc details in
# + https://docs.google.com/spreadsheets/d/1Eo0F4GcmqWZ1cghTpQlA4aHsc8kTABss-HAeimE2IqA/edi
# + https://docs.google.com/spreadsheets/d/1LAsyTZ2ZRP_xLiUBkqmawwnKWgy8OCwq4mmWrrc_rpQ/edi

jsPsychAdmin::check_missing_prepare_TASK(sync_protocols = TRUE,
                                         check_trialids = TRUE,
```

```
                                                check_new_task_tabs = TRUE,
                                                delete_nonexistent = FALSE,
                                                gmail_account = "myemail@gmail.com")

  # Will sync all protocols on server to the LOCAL folder ../CSCN-server
  # Then, will CHECK:
  # - Tasks with no prepare_TASK() script!
  # - Tasks NOT in Google Doc
  # - Check trialid's are OK
  # - Check no missing info in Google doc of NEW tasks
  jsPsychAdmin::download_check_all_protocols(gmail_account = "myemail@gmail.com")
```

6.0.1.6 Download/Upload specific protocol

```
  # UPLOAD
jsPsychHelpeR::sync_server_local(
  direction = "local_to_server",
  local_folder = "protocols_DEV/999/",
  server_folder = "test/protocols_DEV/999/",
  only_test = TRUE,
  delete_nonexistent = FALSE
)

  # DOWNLOAD
jsPsychHelpeR::sync_server_local(
  direction = "server_to_local",
  server_folder = "test/protocols_DEV/999/",
  local_folder = "protocols_DEV/999/",
  only_test = TRUE,
  delete_nonexistent = FALSE
)
```

6.0.1.7 Build jsPsychHelpeR and jsPsychMonkeys packages

Scripts to build the template projects for jsPsychHelpeR and jsPsychMonkeys

```
jsPsychAdmin::create_jsPsychHelpeR_zip()
jsPsychAdmin::create_jsPsychMonkeys_zip()
```

6.0.1.8 Create & Simulate & Prepare

```
# Extract example tasks to a folder
jsPsychMaker::copy_example_tasks(destination_folder = "~/Downloads/example_tasks/")

# Create a protocol with the example tasks
jsPsychMaker::create_protocol(
  folder_tasks = "~/Downloads/example_tasks/",
  folder_output = "~/Downloads/protocolALL999",
  launch_browser = FALSE,
  options_separator = ";"
)

# AUTOMATICALLY run Monkeys and prepare data

# Only prints in console the code
jsPsychAdmin::simulate_prepare(folder_protocol = "~/Downloads/protocolALL999",
                               n_participants = 3,
                               print_watch_run = "print")

# Runs everything
jsPsychAdmin::simulate_prepare(folder_protocol = "~/Downloads/protocolALL999",
                               n_participants = 3,
                               print_watch_run = "run")

# Run a single Monkey and open VNC to see how it goes
jsPsychAdmin::simulate_prepare(folder_protocol = "~/Downloads/protocolALL999",
                               print_watch_run = "watch")
```

6.1 Docker containers

We can create a fully reproducible docker image with the data preparation and analysis for a specific project using `jsPsychHelpeR::create_docker_container()`

Afterwards, you can use the image to run a docker container that will reproduce the analysis and results of your project.

The current version is a first attempt at this, so there is a lot to improve.

6.1.1 Install Docker

First, we need to install docker.

- Linux: follow installation instructions
- Mac: follow installation instructions
- Windows:
  - Install docker desktop
  - Update wsl (in a command prompt): wsl – update

6.1.2 Create image for a project

When a project is ready to share, you can create a self-contained docker image:

```
# Clean environment ----------------------------------------------------

  # DELETE ALL CACHE
  system("docker builder prune --all -f")

  # Clean renv cache libraries
  renv::clean()

  # Clean extra versions of libraries
  source("R/helper_functions_extra.R")
  clean_renv_cache()


# Create docker image --------------------------------------------------

  PID = 999
  jsPsychHelpeR::create_docker_container(PID = PID)
```

6.1.3 Store image

You can create a tar file with the image or directly share it through dockerhub:

1. Store image creating a pid[PID].tar.zip file TO SHARE

```
PID = 999
system(paste0("docker save gorkang/jspsychhelper:pid", PID, " | zip > pid", PID, ".tar.zip
```

2. Push image to Dockerhub

```
PID = 999
system(paste0("docker push gorkang/jspsychhelper:pid", PID))
```

6.1.4 Load image

You can load the image in your computer in two ways:

1. Using a pid[PID].tar.zip:

On linux:

```
PID = 999
utils::unzip(zipfile = paste0("pid", PID, ".tar.zip"), files = paste0("-"))
system(paste0("docker load --input -"))
```

On windows:

```
tar -xf pid999.tar.zip & docker load input -
```

2. Pull image from Dockerhub

```
PID = 999
system(paste0("docker pull gorkang/jspsychhelper:pid", PID))
```

6.1.5 Run container

Once the docker image is loaded in your system, you will be able to run the data preparation and analysis for your project inside a docker container, ensuring reproducibility. The output will be in `Downloads/jsPsychHelpeR[PID]/outputs` after a couple of minutes.

- Linux

```
  # Make sure ~/Downloads/jsPsychHelpeR999 is empty
  file.remove(list.files(paste0("~/Downloads/jsPsychHelpeR", PID, "/outputs"), recursive = TR

  # Run docker
  system(paste0("docker run --rm -d --name pid", PID, " -v ~/Downloads/jsPsychHelpeR", PID, "
```

- Windows

```
docker run --rm -d --name jspsychhelper -v %USERPROFILE%\Downloads\jsPsychHelpeR\outputs:/hom
```

6.1.6 DEBUG Container

You can DEBUG a container with the following command:

```
docker run --rm -ti -v ~/Downloads/jsPsychHelpeR999/outputs:/home/project/jsPsychHelpeR/outpu
```

Inside the container, you can acces R and debug as you would locally.

```
# See last CMD line in Dockerfile_TEMPLATE:
# $ R
source('renv/activate.R')
invisible(lapply(list.files('./R', full.names = TRUE, pattern = '.R$'), source))
setup_folders(pid = 999, folder = '.')
targets::tar_destroy(ask = FALSE)
targets::tar_make()

# Check size folders
du -had1 renv/ | sort -h
du -had1 * | sort -h
```

6.2 Google Docs

We have a few Google Documents with information about available tasks, protocols, etc.

- All tasks
- List of protocols
- NEW tasks
- Checks specific tasks

6.3 Folders and how to work

We have two main locations, the Github jsPsychMaker project and the server.

Github jsPsychMaker project

- canonical_protocol:
  - machinery: last stable version
  - tasks: all available tasks
  - server: `protocols/999/`
- canonical_protocol_DEV
  - machinery: development version
  - tasks: all available tasks
  - server: `protocols/test/canonical_protocol_DEV/`
- canonical_protocol_clean
  - machinery: last stable version
  - tasks: Consent and Goodbye
  - server: `protocols/test/canonical_protocol_clean/`
- protocols_DEV
  - machinery: last stable version
  - should only contain tasks in development
  - server: `protocols/test/protocols_DEV/`

In `protocols_DEV` we prepare the new **protocolos**:

- Create a copy in `test/protocols_DEV` of `canonical_protocol_clean` and rename to the number of the new protocol, `test/protocols_DEV/NumberOfProtocol`
- Once the protocol is ready:
  - Copy protocol to root folder: `protocols/NumberOfProtocol`
  - ZIP subfolder and move zip to `protocols/test/protocols_DEV/OLD_TESTS/`
  - Delete folder `test/protocols_DEV/NumberOfProtocol`
  - If there are new tasks:
    - CHECK with: check_missing_prepare_TASK()
    - TEST with create_protocol_with_NEW_tasks.R
    - Copy tasks, images, videos, specific plugins, etc. to `protocols/999/`
    - TEST in canonical protocol `protocols/999/` just in case there is a weird interaction

6.4 Helper functions

There are a number of helper functions to make some of the jsPsychR admins tasks easier.

6.4.1 Check all protocols

For example, we can use `check_missing_prepare_TASK()` to:

- Download all the protocols (without data) to a local folder (`sync_protocols = TRUE`)
- Check the trialid's of all the tests are OK (`check_trialids = TRUE`)
- Check there are no duplicate short_name of tasks in the tareas jsPsychR and NUEVAS tareas
- Check which tasks do not have a prepare_TASK.R script
- Check tasks with no info on the tareas jsPsychR Google doc
- Check tasks with missing info on NUEVAS tareas

```
# Open jsPsychHelpeR RStudio project

  # Load check_missing_prepare_TASK() function
  # cli::cli_alert_info(getwd())

  WD = getwd()
  setwd("../../jsPsychHelpeR/")
  source("R/check_missing_prepare_TASK.R")
  # source("../../jsPsychHelpeR/R/check_missing_prepare_TASK.R")
  setwd(WD)

  # If sync_protocols = TRUE, will download to ../CSCN-server/protocols all the protocols fr
  DF_missing = check_missing_prepare_TASK(sync_protocols = FALSE,
                                          check_trialids = TRUE,
                                          delete_nonexistent = TRUE,
                                          check_new_task_tabs = TRUE,
                                          helper_folder = "../../jsPsychHelpeR",
                                          CSCN_server_folder = "../../CSCN-server/")

  # - Tasks with no prepare_TASK() script!
  # - Tasks NOT in Google Doc
  # - Check trialid's are OK
  DF_missing
```

```r
DF_missing$DF_FINAL %>% tidyr::replace_na(list(missing_script = "",
                                               missing_googledoc = "",
                                               missing_task = ""))


# Tasks ready to create prepare_*.R script
DF_missing$DF_FINAL %>% filter(!is.na(missing_script) & is.na(missing_gdoc))
DF_missing$DF_FINAL %>% filter(!is.na(missing_script) | !is.na(missing_gdoc)) %>%
  filter(!task %in% c("DEMOGR24", "DEMOGRfondecyt2022E1", "ITC", "fauxPasEv")) %>%  # "MDDI
  select(-matches("missing"), -Nombre, -Descripcion)
```

6.4.2 Create protocol with NEW tasks

With `create_protocol_with_NEW_tasks.R` we can create a protocol with the tasks for which we do not yet have a control snapshot (no .csv's in the 999.zip data).

This is a necessary step before the task can be moved to the canonical protocol.

The function `find_missing_tasks_in_999()` will read all the csv in `jsPsychHelpeR/data/999/999.zip` and depending on the value of the parameter `search_where` ("prepare_TASK" or "js"):

- all .js tasks in `jsPsychMaker/protocols_DEV/`
- all prepare_TASK.R in `jsPSychHelpeR/R_tasks/`

Comparing both sources, will look for tasks for which we do not have a .csv in the 999 protocol yet (remember the 999 protocol is the canonical_protocol in the server).

Then, it prepares a NEW_TASKS protocol using the tasks.js files found in the server (after downloading all the server protocols to `jsPsychR/CSCN-server/`). A couple important points:

- This is a bit tricky, as it will use all the tasks in the server that it can found, across all the protocols, and will select the newest one.
- Sometimes there are multiple copies, with different dates and sizes…
- It is important that the server is as clean as possible. With all the OLD non-updated protocols zipped.

To make sure the Github `jsPsychMaker/protocols_DEV/NEW_TASKS/` is up to date, `create_protocol_with_NEW_tasks.R` will UPLOAD `CSCN-server/.../NEW_TASKS` to the server, and then DOWNLOAD NEW_TASKS to `../jsPsychMaker/protocols_DEV/NEW_TASKS/`

6.4.3 Check canonical protocol DEV

With `000_CHECK_CANONICAL.R` we can check that the canonical protocol in development works and expected.

In the script you can:

- sync `canonical_protocol_DEV/` folder in jsPsychMaker to `999/` in the server
- launch 5 monkeys
- rename the csv files to a fixed date, etc.
- prepare data
- compare with snaphot (WIP)

7 New protocols and tasks

There are a number of elements the tasks need to work well with jsPsychR, so we recommend to use one of the systems we have developed.

For example, with `jsPsychMaker::create_protocol()`, you can use tasks we already developed, and/or create new tasks defining their parameters in csv/excel files. The tasks will be part of a fully working protocol. You will need R 4.2 or higher to use it.

`create_protocol()` can:

- Loop through the subfolders in `folder_tasks` to create one task per subfolder
- Copy `canonical_protocol_clean` to `folder_output`
- Include in the protocol any tasks in `canonical_tasks`
- Modify `config.js` to add all the tasks created and selected
- Modify `index.html` to inlude only the plugins those tasks will use
- Modify `index.html` to add the media those task will use
- Check task names are OK: no spaces, -, _, do not start by a number, …
- Check names of all trialid's are OK (NAMETASK_NUMBER, e.g. MYTASK_001)
- Check only one csv or xls/xlsx file per folder
- Check plugins used exist in jsPsych-6/plugins
- Check we have the necessary parameters (WIP)
- Delete files of plugins not used

`create_protocol()` cannot yet:

- Modify config.js to adapt `all_conditions` to the experimental tasks added
- Use shiny app to edit the local config.js

7.1 New protocols

You can create a new protocol in seconds, choosing from the tasks we already have available.

Make sure you have the last version of jsPsychMaker, installing from Github:

```
if (!require('pak')) install.packages('pak'); pak::pkg_install("gorkang/jsPsychMaker")
```

Check if there are new tasks available in a new version of the Github package:

```
jsPsychMaker::check_NEW_tasks_Github()
```

7.1.1 List available tasks

You can list available tasks to choose from. You have more details in the section available-tasks. You can choose between available tasks for jsPsych 6.3 and jsPsych 7.3 with `jsPsych_version`.

```
  jsPsychMaker::list_available_tasks(jsPsych_version = 6)
```

```
$tasks
  [1] "ACE"                 "AIM"                 "AntiBots"
  [4] "APrioriDiagnostic"   "APrioriScreening"    "Bank"
  [7] "BART"                "BDI"                 "BNT"
 [10] "bRCOPE"              "CAMIR"               "CAS"
 [13] "CDRISC10"            "CEL"                 "CIT"
 [16] "CMApost"             "CMApre"              "Consent"
 [19] "ConsentAudio"        "ConsentHTML"         "Cov19Q"
 [22] "COVIDCONTROL"        "CRQ"                 "CRS"
 [25] "CRT7"                "CRTMCQ4"             "CRTv"
 [28] "CS"                  "CTS2"                "CTT"
 [31] "DASS21"              "DEBRIEF"             "DEMOGR"
```

```
 [34] "DES"                 "DGLS"                 "DMW"
 [37] "EAR"                 "ECRRS"                "EmpaTom"
 [40] "EQ"                  "ERQ"                  "ESM"
 [43] "ESV"                 "ESZ"                  "fauxPasEv"
 [46] "FDMQ"                "FKEA"                 "GASP"
 [49] "GBS"                 "GHQ12"                "Goodbye"
 [52] "HRPVB"               "HRPVBpost"            "IBT"
 [55] "ICvsID"              "IDQ"                  "IEC"
 [58] "INFCONS"             "IRI"                  "IRS"
 [61] "ITQ"                 "LoB"                  "LOT"
 [64] "LSNS"                "MAIA"                 "MAIAY"
 [67] "MCQ30"               "MDDF"                 "MDMQ"
 [70] "MIS"                 "NARS"                 "OBJNUM"
 [73] "OTRASRELIG"          "PBSr"                 "PERMA"
 [76] "PPD"                 "PRFBM"                "PRFBMpost"
 [79] "PSC"                 "PSETPP"               "PSPPC"
 [82] "PSS"                 "PVC"                  "PWb"
 [85] "REI40"               "Report"               "RMET"
 [88] "RobToM"              "RSS"                  "RTS"
 [91] "SASS"                "SBS"                  "SCGT"
 [94] "SCSORF"              "SDG"                  "SDQ20"
 [97] "SILS"                "SPM2"                 "sProQOL"
[100] "SRA"                 "SRBQP"                "SRSav"
[103] "STAI"                "STAIC"                "SWBQ"
[106] "UCLA"                "WaisMatrices"         "WaisMatricesES"
[109] "WaisWorkingMemory"   "WaisWorkingMemoryES"  "WEBEXEC"

$tasks_js
  [1] "ACE.js"                  "AIM.js"                  "AntiBots.js"
  [4] "APrioriDiagnostic.js"    "APrioriScreening.js"     "Bank.js"
  [7] "BART.js"                 "BDI.js"                  "BNT.js"
 [10] "bRCOPE.js"               "CAMIR.js"                "CAS.js"
 [13] "CDRISC10.js"             "CEL.js"                  "CIT.js"
 [16] "CMApost.js"              "CMApre.js"               "Consent.js"
 [19] "ConsentAudio.js"         "ConsentHTML.js"          "Cov19Q.js"
 [22] "COVIDCONTROL.js"         "CRQ.js"                  "CRS.js"
 [25] "CRT7.js"                 "CRTMCQ4.js"              "CRTv.js"
 [28] "CS.js"                   "CTS2.js"                 "CTT.js"
 [31] "DASS21.js"               "DEBRIEF.js"              "DEMOGR.js"
 [34] "DES.js"                  "DGLS.js"                 "DMW.js"
 [37] "EAR.js"                  "ECRRS.js"                "EmpaTom.js"
 [40] "EQ.js"                   "ERQ.js"                  "ESM.js"
 [43] "ESV.js"                  "ESZ.js"                  "fauxPasEv.js"
```

```
 [46] "FDMQ.js"                "FKEA.js"                "GASP.js"
 [49] "GBS.js"                 "GHQ12.js"               "Goodbye.js"
 [52] "HRPVB.js"               "HRPVBpost.js"           "IBT.js"
 [55] "ICvsID.js"              "IDQ.js"                 "IEC.js"
 [58] "INFCONS.js"             "IRI.js"                 "IRS.js"
 [61] "ITQ.js"                 "LoB.js"                 "LOT.js"
 [64] "LSNS.js"                "MAIA.js"                "MAIAY.js"
 [67] "MCQ30.js"               "MDDF.js"                "MDMQ.js"
 [70] "MIS.js"                 "NARS.js"                "OBJNUM.js"
 [73] "OTRASRELIG.js"          "PBSr.js"                "PERMA.js"
 [76] "PPD.js"                 "PRFBM.js"               "PRFBMpost.js"
 [79] "PSC.js"                 "PSETPP.js"              "PSPPC.js"
 [82] "PSS.js"                 "PVC.js"                 "PWb.js"
 [85] "REI40.js"               "Report.js"              "RMET.js"
 [88] "RobToM.js"              "RSS.js"                 "RTS.js"
 [91] "SASS.js"                "SBS.js"                 "SCGT.js"
 [94] "SCSORF.js"              "SDG.js"                 "SDQ20.js"
 [97] "SILS.js"                "SPM2.js"                "sProQOL.js"
[100] "SRA.js"                 "SRBQP.js"               "SRSav.js"
[103] "STAI.js"                "STAIC.js"               "SWBQ.js"
[106] "UCLA.js"                "WaisMatrices.js"        "WaisMatricesES.js"
[109] "WaisWorkingMemory.js"   "WaisWorkingMemoryES.js" "WEBEXEC.js"
```

7.1.2 Create a protocol

This will create a fully working protocol in `folder_output`. You can edit `config.js` to adapt the protocol to your needs. See experiment configuration for more details.

```
jsPsychMaker::create_protocol(canonical_tasks = c("AIM", "EAR", "IRI"),
                              folder_output = "~/Downloads/protocol999",
                              launch_browser = TRUE,
                              jsPsych_version = 6)
```

7.2 New tasks

7.2.1 Create tasks

You can create new tasks with `create_task()` using `csv` or `xls/xlxs` files for the items, and `html` files for the instructions. But we recommend you use `create_protocol()` instead, so the tasks will be part of a fully working protocol, and testing them will be a breeze.

There are some things to take into account:

- **folder_tasks** expects a folder with sub-folders with the ShortName of tasks (ShortName is an example of the ShortName of a task, for example, MyTask). Inside, they need to have one `ShortName.csv` or `ShortName.xls/xlxs` file and `*_instructions.html` files. Use `jsPsychMaker::copy_example_tasks(destination_folder = "~/Downloads/TEST")` to see a working example
- The **csv or xls/xlsx file** (ShortName.csv or Shortname.xls/xlsx) needs to have an `ID` and `plugin` columns, and then columns by the name of parameters used in the plugin (e.g. if using the `survey-text` plugin, you will need the `prompt` parameter). If you need **help with the plugins parameters**, see the jsPsych 6.3 list of plugins
- For each **html file** ending with `_instructions.html` or `instructions#.html` (# is a number), an instructions page will be created (e.g. ShortName_*instructions*.html, ShortName_*instructions2*.html, etc). If there is no html, a default page will be used.
- For **key questions** (e.g. present this question only if participants responded "3"), you need to create a column named `if_question` and include a logical condition. For example:
  - `1 != 25`: Response to item 1 is NOT 25
  - `3 == 20`: Response to item 3 is 20
  - `15 == yes`: Response to item 3 is yes
- If you use tasks with images, video or audio, make sure to include the files in a `media/` folder, inside a subfolder with the name of the task. So, if your task name is `ALL`:
  - Images: `media/images/ALL`
  - Videos: `media/videos/ALL`
  - Audio: `media/audios/ALL`
- If you use a plugin with different options or alternatives (e.g. `survey-multi-choice-vertical`), the different response `options` will be the words or sentences separated by semi-colons (e.g. Yes, I am; No, I am not). If your options have semi-colons, you can to use the `options_separator` parameter to change the default.

You can run the fully reproducible example included in jsPsychMaker:

1) Install jsPsychMaker from Github and load library

```
if (!require('pak')) install.packages('pak'); pak::pkg_install("gorkang/jsPsychMaker")
```

2) Copy example tasks

This will copy a few example tasks that you use to adapt your tasks. For example, `MultiChoice` and `Slider` tasks, a key questions mini-task (`IfQuestion`), and an `ImageButtonResponse` task.

```
jsPsychMaker::copy_example_tasks(destination_folder = "~/Downloads/ExampleTasks")
```

3) Create your protocol

```
# Create protocol
jsPsychMaker::create_protocol(folder_tasks = "~/Downloads/ExampleTasks/",
                              folder_output = "~/Downloads/protocol999",
                              launch_browser = TRUE)
```

7.3 HELP with new tasks

If you need help developing new tasks, you can open a new Issue in the jsPsychMaker Github.

We will ask you to add the details about the task in the NEW tasks document.

Once the task is implemented, our goal is to always end up having a sister task preparation script in jsPsychHelpeR. You can try to create the preparation script and do a Pull request, or ask for help opening a new Issue in the jsPsychHelpeR Github.

7.3.1 How to fill the NEW tasks document

NEW tasks document

First of all, you will need the original paper where the task was validated/translated to have all the details at hand. Please, send us a link to the paper.

The best way to fill the NEW tasks document is:

1. Find a task similar to yours in the document Tareas jsPsychR where we have information about all the available tasks.
2. Copy/paste the information from all the tabs to the NEW tasks document and adapt it.

Try to be as consistent as possible. For example, when entering the information about numeric conversion in the Puntajes_items tab:

```
All the cells must be:
1 = Mucho
2 = Poco
...
```

DO NOT do things like:

```
1: Mucho
1 Mucho
1 pto = Mucho
Mucho 1
```

Please, make sure you fill out all the details in all the tabs.

8 CreateSimulatePrepare

Here you can see the full process of creating a protocol with jsPsychMaker, simulating participants with jsPsychMonkeys and preparing the data with jsPsychHelpeR:

8.1 Create protocol

Create a protocol with `jsPsychMaker::create_protocol()`:

```
# 1) Install jsPsychMaker
if (!require('remotes')) install.packages('remotes'); remotes::install_github("gorkang/jsPsy

# 2) Check available tasks
jsPsychMaker::list_available_tasks()$tasks

# 3) Create protocol
jsPsychMaker::create_protocol(canonical_tasks = c("AIM", "EAR", "IRI"),
                              folder_output = "~/Downloads/protocol999",
                              launch_browser = FALSE)
```

You can now edit the configuration file (`~/Downloads/protocol999/config.js`) to adjust the project's parameters.

8.2 Simulate participants

Simulate participants with {jsPsychMonkeys}. Make sure your system has a functioning docker installation, see jsPsychMonkey's setup.

If you are on Windows, make sure Docker Desktop is open and running before releasing the monkeys.

```
# 1) Install jsPsychMonkeys
if (!require('remotes')) utils::install.packages('remotes'); remotes::install_github('gorkang

# 2) Run monkeys
  # Go to _targets.R: Change parameter `local_folder_tasks` to your folder_output above. For
  # - On Ubuntu, `local_folder_tasks = "~/Downloads/protocol999"`
  # - On Windows, `local_folder_tasks = "C:/Users/myusername/Downloads/protocol999"`

jsPsychMonkeys::release_the_monkeys(uid = 1:10,
                                    local_folder_tasks = "~/Downloads/protocol999/")
```

The monkeys responses csv's should be initially downloaded in your `Downloads` folder, and automatically moved to a `.data/` folder inside the protocol folder. For example, `~/Downloads/protocol999/.data`

8.3 Prepare data

Create a data preparation project with `jsPsychHelpeR::run_initial_setup()`:

```
# 1) Install
if (!require('remotes')) install.packages('remotes'); remotes::install_github("gorkang/jsPsyc

# 2) Create project
jsPsychHelpeR::run_initial_setup(pid = "999", data_location = "~/Downloads/protocol999/.data"

# 3) Restore all the necessary packages using {renv}
renv::restore(prompt = FALSE)

# 4) Run data preparation
targets::tar_make()
```

If you don't give a value to the `folder` parameter in `jsPsychHelpeR::run_initial_setup()`, the new project will be created in `~/Downloads/jsPsychHelpeRtest/`. After step 4), the prepared data can be found in the `outputs/data` folder of the new project, reports in `outputs/reports`, etc.

9 Common Tasks

9.1 Install dependencies

Install the last versions of all the packages:

```
if (!require('pak')) install.packages('pak'); pak::pkg_install("gorkang/jsPsychAdmin")
if (!require('pak')) install.packages('pak'); pak::pkg_install("gorkang/jsPsychMaker")
if (!require('pak')) install.packages('pak'); pak::pkg_install("gorkang/jsPsychMonkeys")
if (!require('pak')) install.packages('pak'); pak::pkg_install("gorkang/jsPsychHelpeR")
```

9.2 Developing a protocol

9.2.1 Adding new tasks

Copy example tasks to a local folder, and adapt with the information of the new task. For example:

```
jsPsychMaker::copy_example_tasks(
  destination_folder = "~/Downloads/ExampleTasks",
  which_tasks = "MultiChoice")
```

9.2.2 Creating a protocol

You can choose any of the canonical tasks (`jsPsychMaker::list_available_tasks()$tasks`), and/or a folder with the new tasks (`~/Downloads/ExampleTasks`).

```
jsPsychMaker::create_protocol(
  canonical_tasks = c("BNT"), # Berlin Numeracy Test
  folder_tasks = "~/Downloads/ExampleTasks/",
  folder_output = "~/Downloads/protocol9996",
  launch_browser = TRUE
)
```

9.2.3 Piloting the protocol

Use `local_folder_tasks` to test local protocols.

```
jsPsychMonkeys::release_the_monkeys(
  uid = 1:10,
  sequential_parallel = "parallel",
  number_of_cores = 10,
  local_folder_tasks = "~/Downloads/protocol9996/",
  open_VNC = FALSE
)
```

If your protocol is on a server, use `server_folder_tasks`. You will need a `credentials_folder` with the server access credentials.

```
jsPsychMonkeys::release_the_monkeys(
  uid = "1:10",
  sequential_parallel = "parallel",
  number_of_cores = 10,
  server_folder_tasks = "test/protocols_DEV/999",
  DEBUG = FALSE,
  credentials_folder = "~/my_location/.vault/",
  open_VNC = FALSE
  )
```

9.2.4 Creating helper project

If it is a local protocol:

```
jsPsychHelpeR::run_initial_setup(
  pid = 9996,
  data_location = "~/Downloads/protocol9996/.data/",
  folder = "~/Downloads/jsPsychHelpeR9996",
  dont_ask = TRUE
  )
```

If it is an online protocol:

```
jsPsychHelpeR::run_initial_setup(
  pid = 9996,
  download_files = TRUE,
  credentials_file = "/path_to_credentials/.credentials",
  folder = "~/Downloads/jsPsychHelpeR9996",
  dont_ask = TRUE
  )
```

9.2.5 Deleting pilot data

```
    # Delete the XYZ protocol rows in all the MYSQL tables
    source("admin/mysql_helper_functions.R")
    # list_credentials = decrypt_data(key_public = readLines(".vault/data_public_key.txt"), c
    delete_MySQL_tables_pid(pid)


  # 3) Limpiar los archivos de resultados de Monkeys -----------------------

    # Delete csv files in .data/
    # rstudioapi::navigateToFile(".vault/.credentials")
    DELETE_data_server(pid = PROTOCOLID)
```

9.2.6 Preparing Helper project

9.2.6.1 Get data

Will download a zip file with the data.

```
# if (!require('renv')) utils::install.packages('renv'); renv::install('gorkang/jsPsychHelpeR

# Developing protocol
jsPsychHelpeR::get_zip(pid = "test/protocols_DEV/31",
                       what = "data",
                       where = "data/")

# Production protocol
jsPsychHelpeR::get_zip(pid = "999",
                       what = "data",
                       where = "data/")
```

9.3 Protocol to production

From `admin/000_PREPARE_protocol_for_production.R` in jsPsychAdmin

```
# Checklist para pasar protocolos de test/protocols_DEV/ a produccion

  # PARAMETERS -------------------------------

  PROTOCOLID = "test/protocols_DEV/31"
  number_of_monkeys = "1:100"

  # ------------------------------------------

  # Automatic parameters
  pid = gsub("test/protocols_DEV/", "", PROTOCOLID)


   cli::cli_h1("PROTOCOL {pid}")


  # 1) Pilotaje final on Monkeys! ------------------------------------------

  # Clean data and MySQL DB
  # rstudioapi::navigateToFile(".vault/.credentials")
  jsPsychAdmin::clean_up_dev_protocol(protocol_id = PROTOCOLID) # Will ask for server passwo


  # LAUNCH MONKEYS
  jsPsychMonkeys::release_the_monkeys(uid = number_of_monkeys,
                                      server_folder_tasks = PROTOCOLID,
                                      sequential_parallel = "parallel",
                                      number_of_cores = 10,
                                      big_container = TRUE,
                                      keep_alive = FALSE,
                                      open_VNC = FALSE,
                                      screenshot = FALSE,
                                      credentials_folder = here::here(".vault/"))


  # CHECK jsPsychHelpeR runs OK

  # || THIS WILL take a while, as all the renv packages need to update
```

```r
# Create NEW jsPsychHelpeR project, downloading the files from the server
jsPsychHelpeR::run_initial_setup(pid = PROTOCOLID,
                                 download_files = TRUE,
                                 folder = "~/Downloads/jsPsychR_TESTING_for_PRODUCTION")

  # REMEMBER TO DO targets::tar_make() in jsPsychHelpeR project!


# 2) Clean data and  MySQL DB ---------------------------------------------

  # rstudioapi::navigateToFile(".vault/.credentials")
  jsPsychAdmin::clean_up_dev_protocol(protocol_id = pid) # Will ask for server password


# 3) Revisar el config.js para pasar el experiment a produccion ----------

  # -[] online = true
  # -[] pid OK?
  # -[] debug_mode = false
  # - ETC...


# 4) Copiar protocolo ZIPeado a test/protocols_DEV/OLD_TESTS/ -------------

  # TODO: automatico!

# 5) Copiar protocolo a protocols/ ----------------------------------------

  # TODO: automatico!

# 6) BORRAR protocolo de test/protocols_DEV/OLD_TESTS/ --------------------

  # TODO: automatico!
```

References

jsPsychR could not be possible without the amazing jsPsych (de Leeuw 2015). We use {targets} to create well-structured and reproducible pipelines (Landau 2021b) for the jsPsychR tools.

jsPsychR implementation

The following R packages are used in jsPsychR: Navarrete and Valencia (2024), Navarrete (2025), Navarrete (2024b), Wickham, Hesselberth, et al. (2025), Wickham and Henry (2026), Landau (2021a), Csárdi (2026), Wickham, François, et al. (2026), Xie, Cheng, et al. (2025), Wickham (2025a), Müller (2025), Firke (2024), Pedersen (2025), William Revelle (2026), Wickham, Hester, and Bryan (2026), Ushey and Wickham (2026), Allaire et al. (2026), Ushey et al. (2026), Wickham (2025c), Müller and Wickham (2026), Wickham, Vaughan, et al. (2025), Almende B.V. and Contributors and Thieurmel (2025), Hester (2025), Wickham, Hester, Chang, et al. (2026), Navarrete (2024a), Navarrete (2025), Csárdi and Hester (2026), R Core Team (2026a), Navarrete and Valencia (2024), R Core Team (2026b), Csárdi and Chang (2024), Barrett et al. (2026), Hester and Bryan (2026), Bryan (2025), Warnes et al. (2023), Bache and Wickham (2026), Wickham and Bryan (2026), Henry and Wickham (2026), Waring et al. (2026), R Core Team (2026c), Ooms (2025), Wilke (2025), Chang et al. (2026), Xie (2025), Wickham, Bryan, et al. (2025), Wickham (2025b), Ooms (2026), Atkins et al. (2026), Cheng et al. (2025), Wickham (2026), Attali (2026), Perrier et al. (2026), Wickham, Girlich, et al. (2026), Iannone et al. (2026), Xie, Allaire, et al. (2025), Csárdi (2024b), Harrison et al. (2026), Meyer and Perrier (2024), Navarrete (2024b), Csárdi et al. (2024), Izrailev (2024), Csárdi (2024a), Hester et al. (2026), Vaughan et al. (2026), Hester et al. (2024), Temple Lang (2026)

Tasks implementation

The tasks have been implemented following: Antúnez and Vinet (2012), Bados et al. (n.d.), Baron-Cohen and Wheelwright (2004), Bezalel (2020), Boehm and Carter (2019), Bosc et al. (1997), Bradley and Lang (1994), Breinbauer K et al. (2009), Buchanan et al. (n.d.), Butler and Kern (2016), Cabello et al. (2013), Carreño-Moreno et al. (2022), Cohen et al. (1983), Cokely et al. (2012), Davis (n.d.), Daza et al. (2002), delValle et al. (2022), Díaz-Vilela

and Álvarez-González (n.d.), Díaz et al. (n.d.), Eckblad and Chapman (n.d.), Everett et al. (2018), Fonseca-Pedrero et al. (n.d.), Galiana et al. (2020), Garcia-Retamero (n.d.), García et al. (2019), Gomez and Fisher (2003), Gross and John (n.d.), Hays and DiMatteo (1987), Hetts et al. (2000), Hildebrandt et al. (2021), Hooker et al. (2000), Houran et al. (2003), Huber and Huber (2012), Jong et al. (2013), Kanske et al. (2015), Keaton (n.d.), Koenig et al. (1988), Lejuez et al. (2002), León et al. (2014), León et al. (2015), León Estrada et al. (2011), Levenson (n.d.), Lins et al. (n.d.), Lipkus et al. (2001), Lovibond and Lovibond (1995), Lyon et al. (2022), Mann et al. (1997), María et al. (n.d.), Mezzadra and Simkin (2017), Moodley et al. (2012), Morejón et al. (2004), Pargament et al. (1998), Pérez-Albéniz et al. (n.d.), Plante and Boccaccini (1997), Pommier et al. (2020), Pons (n.d.), Psoteg (n.d.), Reyna (2016), Rojas-Barahona et al. (2009), Ruiz et al. (2017), Ryff and Keyes (n.d.), Sala et al. (2012), Sánchez-López and Dresch (2008), Silva et al. (2020), Sirota and Juanchich (2018), Sirota et al. (2020), Ståhl et al. (2016), Stamm (n.d.), Stoll et al. (2016), Stone et al. (1998), Tobacyk (n.d.), Tobacyk and Milford (n.d.), Toplak et al. (2013), Vale et al. (2012), Weeks (2019), Weeks et al. (2020), Wu and Yao (2008), *Coeficiente de Empatía* (n.d.), *MANUAL PARA LA UTILIZACIÓN DEL CUESTIONARIO* (n.d.)

Bibliography

Allaire, JJ, Yihui Xie, Christophe Dervieux, et al. 2026. *rmarkdown: Dynamic Documents for r*. https://github.com/rstudio/rmarkdown.

Almende B.V. and Contributors, and Benoit Thieurmel. 2025. *visNetwork: Network Visualization Using "vis.js" Library*. https://datastorm-open.github.io/visNetwork/.

Antúnez, Zayra, and Eugenia V Vinet. 2012. "Escalas de depresión, ansiedad y estrés (DASS – 21): Validación de la Versión Abreviada en Estudiantes Universitarios Chilenos." *terapia psicolÓgica* 30: 9. https://doi.org/10.4067/S0718-48082012000300005.

Atkins, Aron, Toph Allen, Hadley Wickham, Jonathan McPherson, and JJ Allaire. 2026. *rsconnect: Deploy Docs, Apps, and APIs to "Posit Connect," "shinyapps.io," and "RPubs"*. https://rstudio.github.io/rsconnect/.

Attali, Dean. 2026. *shinyjs: Easily Improve the User Experience of Your Shiny Apps in Seconds*. https://deanattali.com/shinyjs/.

Bache, Stefan Milton, and Hadley Wickham. 2026. *magrittr: A Forward-Pipe Operator for r*. https://magrittr.tidyverse.org.

Bados, Arturo, Antonio Solanas, and Raquel Andrés. n.d. *Psychometric Properties of the Spanish Version of Depression, Anxiety and Stress Scales (DASS)*. 6.

Baron-Cohen, Simon, and Sally Wheelwright. 2004. "The Empathy Quotient: An Investigation of Adults with Asperger Syndrome or High Functioning Autism, and Normal Sex Differences." *Journal of Autism and Developmental Disorders* 34 (2): 163–75. https://doi.org/10.1023/B:JADD.0000022607.19833.00.

Barrett, Tyson, Matt Dowle, Arun Srinivasan, et al. 2026. *data.table: Extension of "data.frame"*. https://r-datatable.com.

Bezalel, Glenn Y. 2020. "'Moral Dumbfounding': Moral Foundations Theory for the Classroom." *Theory and Research in Education* 18 (2): 191–210. https://doi.org/10.1177/1477878520934014.

Boehm, Thomas L., and Erik W. Carter. 2019. "Facets of Faith: Spirituality, Religiosity, and Parents of Individuals With Intellectual Disability." *Intellectual and Developmental Disabilities* 57 (6): 512–26. https://doi.org/10.1352/1934-9556-57.6.512.

Bosc, M., A. Dubini, and V. Polin. 1997. "Development and Validation of a Social Functioning Scale, the Social Adaptation Self-evaluation Scale." *European Neuropsychopharmacology* 7 (1): S57–70. https://doi.org/10.1016/S0924-977X(97)00420-3.

Bradley, Margaret M., and Peter J. Lang. 1994. "Measuring Emotion: The Self-Assessment Manikin and the Semantic Differential." *Journal of Behavior Therapy and Experimental Psychiatry* 25 (1): 49–59. https://doi.org/10.1016/0005-7916(94)90063-9.

Breinbauer K, Hayo, Hugo Vásquez V, Sebastián Mayanz S, Claudia Guerra, and Teresa Millán K. 2009. "Validación en Chile de la Escala de Sobrecarga del Cuidador de Zarit en sus versiones original y abreviada." *Revista médica de Chile* 137 (5). https://doi.org/10.4067/S0034-98872009000500009.

Bryan, Jennifer. 2025. *Googlesheets4: Access Google Sheets Using the Sheets API V4*. https://googlesheets4.tidyverse.org.

Buchanan, Tom, Thomas M Heffernan, Andrew C Parrott, Jonathan Ling, Jacqui Rodgers, and Andrew B Scholey. n.d. *A Short Self-Report Measure of Problems with Executive Function Suitable for Administration via the Internet*. 6. https://doi.org/10.3758/BRM.42.3.709.

Butler, Julie, and Margaret L. Kern. 2016. "The PERMA-Profiler: A Brief Multidimensional Measure of Flourishing." *International Journal of Wellbeing* 6 (3): 1–48. https://doi.org/10.5502/ijw.v6i3.526.

Cabello, Rosario, José M. Salguero, Pablo Fernández-Berrocal, and James J. Gross. 2013. "A Spanish Adaptation of the Emotion Regulation Questionnaire." *European Journal of*

Psychological Assessment 29 (4): 234–40. https://doi.org/10.1027/1015-5759/a000150.

Carreño-Moreno, Sonia Patricia, Lorena Chaparro-Díaz, Nelly Cáliz Romero, and Nathaly Rivera Romero. 2022. "Validez de la escala de soledad UCLA en cuidadores de pacientes crónicos en Colombia." *Investigación en Enfermería: Imagen y Desarrollo*, ahead of print, July 13. https://doi.org/10.11144/Javeriana.ie24.vesu.

Chang, Winston, Joe Cheng, JJ Allaire, et al. 2026. *shiny: Web Application Framework for r*. https://shiny.posit.co/.

Cheng, Joe, Carson Sievert, Barret Schloerke, Winston Chang, Yihui Xie, and Jeff Allen. 2025. *htmltools: Tools for HTML*. https://github.com/rstudio/htmltools.

Coeficiente de Empatía. n.d.

Cohen, Sheldon, Tom Kamarck, and Robin Mermelstein. 1983. "A Global Measure of Perceived Stress." *Journal of Health and Social Behavior* 24 (4): 385–96. https://doi.org/10.2307/2136404.

Cokely, Edward T, Mirta Galesic, Eric Schulz, Rocio Garcia-Retamero, and Saima Ghazal. 2012. "Measuring Risk Literacy: The Berlin Numeracy Test." *Judgment and Decision Making* 7 (1): 23.

Csárdi, Gábor. 2024a. *crayon: Colored Terminal Output*. https://r-lib.github.io/crayon/.

Csárdi, Gábor. 2024b. *pingr: Check If a Remote Computer Is up*. https://r-lib.github.io/pingr/.

Csárdi, Gábor. 2026. *cli: Helpers for Developing Command Line Interfaces*. https://cli.r-lib.org.

Csárdi, Gábor, and Winston Chang. 2024. *callr: Call r from r*. https://callr.r-lib.org.

Csárdi, Gábor, and Jim Hester. 2026. *pak: Another Approach to Package Installation*. https://pak.r-lib.org/.

Csárdi, Gábor, Jim Hester, Hadley Wickham, Winston Chang, Martin Morgan, and Dan Tenenbaum. 2024. *remotes: R Package Installation from Remote Repositories, Including "GitHub"*. https://remotes.r-lib.org.

Davis, Mark H. n.d. *Measuring Individual Differences in Empathy: Evidence for a Multidimensional Approach*. 14. https://doi.org/10.1037/0022-3514.44.1.113.

Daza, Patricia, Diane M Novy, Melinda A Stanley, and Patricia Averill. 2002. "The Depression Anxiety Stress Scale-21: Spanish Translation and Validation with a Hispanic Sample." *Journal of Psychopathology and Behavioral Assessment*, 11. https://doi.org/10.1023/A:1016014818163.

delValle, Macarena V., María Laura Andrés, Sebastián Urquijo, Eliana V. Zamora, Ashish Mehta, and James J. Gross. 2022. "Argentinean Adaptation and Psychometric Properties of the Emotion Regulation Questionnaire (ERQ)." *Psychological Reports* 125 (5): 2733–59. https://doi.org/10.1177/00332941211021343.

Díaz, Darío, Raquel Rodríguez-Carvajal, Amalio Blanco, Bernardo Moreno-Jiménez, and Ismael Gallardo. n.d. *Adaptación española de las escalas de bienestar psicológico de Ryff*.

Díaz-Vilela, Luis, and Carlos J Álvarez-González. n.d. "DIFFERENCES IN PARANORMAL BELIEFS ACROSS FIELDS OF STUDY FROM A SPANISH ADAPTATION OF TOBACYK'S RPBS." *The Journal of Parapsychology*.

Eckblad, Mark, and Loren J Chapman. n.d. *Magical Ideation as an Indicator of Schizotypy*. 11. https://doi.org/10.1037/0022-006X.51.2.215.

Everett, Jim A. C., Nadira S. Faber, Julian Savulescu, and Molly J. Crockett. 2018. "The Costs of Being Consequentialist: Social Inference from Instrumental Harm and Impartial Beneficence." *Journal of Experimental Social Psychology* 79 (November): 200–216. https://doi.org/10.1016/j.jesp.2018.07.004.

Firke, Sam. 2024. *janitor: Simple Tools for Examining and Cleaning Dirty Data*. https://github.com/sfirke/janitor.

Fonseca-Pedrero, Eduardo, Mercedes Paino, Serafín Lemos-Giráldez, Eduardo García-Cueto, Úrsula Villazón-García, and José Muñiz. n.d. *Psychometric Properties of the Perceptual Aberration Scale and the Magical Ideation Scale in Spanish College Students*. 9.

Galiana, Laura, Amparo Oliver, Fernanda Arena, et al. 2020. "Development and Validation of the Short Professional Quality of Life Scale Based on Versions IV and V of the Professional Quality of Life Scale." *Health and Quality of Life Outcomes* 18 (1): 364. https://doi.org/10.1186/s12955-020-01618-3.

García, Jorge Acosta, Francisco Checa y Olmos, Manuel Lucas Matheu, and Tesifón Parrón Carreño. 2019. "Self Esteem Levels Vs Global Scores on the Rosenberg Self-Esteem Scale." *Heliyon* 5 (3): e01378. https://doi.org/10.1016/j.heliyon.2019.e01378.

Garcia-Retamero, Rocio. n.d. *Habilidades numéricas y salud: una revisión crítica*. 13.

Gomez, Rapson, and John W Fisher. 2003. "Domains of Spiritual Well-Being and Development and Validation of the Spiritual Well-Being Questionnaire." *Personality and Individual Differences* 35 (8): 1975–91. https://doi.org/10.1016/S0191-8869(03)00045-X.

Gross, James J, and Oliver P John. n.d. *Individual Differences in Two Emotion Regulation Processes: Implications for Affect, Relationships, and Well-Being*. 15. https://doi.org/10.1037/0022-3514.85.2.348.

Harrison, John, Ju Yeong Kim, and Jonathan Völkle. 2026. *RSelenium: R Bindings for "Selenium WebDriver"*. https://docs.ropensci.org/RSelenium/.

Hays, Ron, and M. Robin DiMatteo. 1987. "A Short-Form Measure of Loneliness." *Journal of Personality Assessment* 51 (1): 69–81. https://doi.org/10.1207/s15327752jpa5101_6.

Henry, Lionel, and Hadley Wickham. 2026. *rlang: Functions for Base Types and Core r and "Tidyverse" Features*. https://rlang.r-lib.org.

Hester, Jim. 2025. *covr: Test Coverage for Packages*. https://covr.r-lib.org.

Hester, Jim, and Jennifer Bryan. 2026. *glue: Interpreted String Literals*. https://glue.tidyverse.org/.

Hester, Jim, Lionel Henry, Kirill Müller, Kevin Ushey, Hadley Wickham, and Winston Chang. 2024. *withr: Run Code "With" Temporarily Modified Global State*. https://withr.r-lib.org.

Hester, Jim, Hadley Wickham, and Gábor Csárdi. 2026. *fs: Cross-Platform File System Operations Based on "libuv"*. https://fs.r-lib.org.

Hetts, John J., David S. Boninger, David A. Armor, Faith Gleicher, and Ariel Nathanson. 2000. "The Influence of Anticipated Counterfactual Regret on Behavior." *Psychology and Marketing* 17 (4): 345–68. https://doi.org/10.1002/(SICI)1520-6793(200004)17:4%3C345::AID-MAR5%3E3.0.CO;2-M.

Hildebrandt, Malin K., Emanuel Jauk, Konrad Lehmann, Lara Maliske, and Philipp Kanske. 2021. "Brain Activation During Social Cognition Predicts Everyday Perspective-Taking: A Combined fMRI and Ecological Momentary Assessment Study of the Social Brain." *NeuroImage* 227 (February): 117624. https://doi.org/10.1016/j.neuroimage.2020.117624.

Hooker, Christine, Neal J. Roese, and Sohee Park. 2000. "Impoverished Counterfactual Thinking Is Associated with Schizophrenia." *Psychiatry* 63 (4): 326–35. https://doi.org/10.1080/00332747.2000.11024925.

Houran, James, Michael A Thalbourne, and Rense Lange. 2003. "Methodological Note:

Erratum and Comment on the Use of the Revised Transliminality Scale." *Consciousness and Cognition* 12 (1): 140–44. https://doi.org/10.1016/S1053-8100(02)00025-9.

Huber, Stefan, and Odilo W Huber. 2012. *The Centrality of Religiosity Scale (CRS)*. 15. https://doi.org/10.3390/rel3030710.

Iannone, Richard, Joe Cheng, Barret Schloerke, et al. 2026. *gt: Easily Create Presentation-Ready Display Tables*. https://gt.rstudio.com.

Izrailev, Sergei. 2024. *tictoc: Functions for Timing r Scripts, as Well as Implementations of "Stack" and "StackList" Structures*. https://github.com/jabiru/tictoc.

Jong, Jonathan, Matthias Bluemke, and Jamin Halberstadt. 2013. "Fear of Death and Supernatural Beliefs: Developing A New Supernatural Belief Scale to Test the Relationship." *European Journal of Personality* 27 (5): 495–506. https://doi.org/10.1002/per.1898.

Kanske, Philipp, Anne Böckler, Fynn-Mathis Trautwein, and Tania Singer. 2015. "Dissecting the Social Brain: Introducing the EmpaToM to Reveal Distinct Neural Networks and Brain–Behavior Relations for Empathy and Theory of Mind." *NeuroImage* 122 (November): 6–19. https://doi.org/10.1016/j.neuroimage.2015.07.082.

Keaton, Shaughan A. n.d. *Profile 53 Rational-Experiential Inventory–40 (REI-40)*. 8.

Koenig, Harold G., David O. Moberg, and James N. Kvale. 1988. "Religious Activities and Attitudes of Older Adults in a Geriatric Assessment Clinic." *Journal of the American Geriatrics Society* 36 (4): 362–74. https://doi.org/10.1111/j.1532-5415.1988.tb02365.x.

Landau, William Michael. 2021a. *tarchetypes: Archetypes for Targets*.

Landau, William Michael. 2021b. "The Targets r Package: A Dynamic Make-Like Function-Oriented Pipeline Toolkit for Reproducibility and High-Performance Computing." *Journal of Open Source Software* 6 (57): 2959. https://doi.org/10.21105/joss.02959.

Leeuw, Joshua R. de. 2015. "jsPsych: A JavaScript Library for Creating Behavioral Experiments in a Web Browser." *Behavior Research Methods* 47 (1): 1–12. https://doi.org/10.3758/s13428-014-0458-y.

Lejuez, C. W., Jennifer P. Read, Christopher W. Kahler, et al. 2002. "Evaluation of a Behavioral Measure of Risk Taking: The Balloon Analogue Risk Task (BART)." *Journal of Experimental Psychology: Applied* 8 (2): 75–84. https://doi.org/10.1037/1076-898X.8.2.75.

León Estrada, Irene, Juan García García, and Lola Roldán Tapia. 2011. "Construcción de la

escala de reserva cognitiva en población española: estudio piloto." *Revista de Neurología* 52 (11): 653. https://doi.org/10.33588/rn.5211.2010704.

León, Irene, Juan García-García, and Lola Roldán-Tapia. 2014. "Estimating Cognitive Reserve in Healthy Adults Using the Cognitive Reserve Scale." *PLoS ONE* 9 (7): e102632. https://doi.org/10.1371/journal.pone.0102632.

León, Irene, Juan García-García, and Lola Roldán-Tapia. 2015. "Escala de Reserva Cognitiva y Envejecimiento." *Anales de Psicología* 32 (1): 218. https://doi.org/10.6018/analesps.32.1.182331.

Levenson, Hanna. n.d. *Activism and Powerful Others: Distinctions Within the Concept of Internal-External Control*. 9. https://doi.org/10.1080/00223891.1974.10119988.

Lindsay, D. Stephen. 2023. "A Plea to Psychology Professional Societies That Publish Journals: Assess Computational Reproducibility." *Meta-Psychology* 7 (September). https://doi.org/10.15626/MP.2023.4020.

Lins, Samuel, Ezra Bottequin, Ádám Dóka, et al. n.d. *To Think, to Feel, to Have: The Effects of Need for Cognition, Hedonism and Materialism on Impulse Buying Tendencies in Adolescents*. https://doi.org/10.5334/jeps.bh.

Lipkus, Isaac M., Greg Samsa, and Barbara K. Rimer. 2001. "General Performance on a Numeracy Scale Among Highly Educated Samples." *Medical Decision Making* 21 (1): 37–44. https://doi.org/10.1177/0272989X0102100105.

Lovibond, P. F., and S. H. Lovibond. 1995. "The Structure of Negative Emotional States: Comparison of the Depression Anxiety Stress Scales (DASS) with the Beck Depression and Anxiety Inventories." *Behaviour Research and Therapy* 33 (3): 335–43. https://doi.org/10.1016/0005-7967(94)00075-U.

Lyon, Aaron R., Catherine M. Corbin, Eric C. Brown, et al. 2022. "Leading the Charge in the Education Sector: Development and Validation of the School Implementation Leadership Scale (SILS)." *Implementation Science* 17 (1): 48. https://doi.org/10.1186/s13012-022-01222-7.

Mann, Leon, Paul Burnett, Mark Radford, and Steve Ford. 1997. "The Melbourne Decision Making Questionnaire: An Instrument for Measuring Patterns for Coping with Decisional Conflict." *Journal of Behavioral Decision Making* 10 (1): 1–19. https://doi.org/10.1002/(SICI)1099-0771(199703)10:1%3C1::AID-BDM242%3E3.0.CO;2-X.

MANUAL PARA LA UTILIZACIÓN DEL CUESTIONARIO. n.d.

María, Ana, Celis Atenas, and Vera Villarroel. n.d. *Propiedades psicométricas de la Escala de Autoestima de Rosenberg en universitarios chilenos*.

Meyer, Fanny, and Victor Perrier. 2024. *shinybusy: Busy Indicators and Notifications for "Shiny" Applications*. https://github.com/dreamRs/shinybusy.

Mezzadra, Joaquín, and Hugo Simkin. 2017. *Validación de la Escala Abreviada de Afrontamiento Religioso Brief-RCOPE en el Contexto Argentino en estudiantes de confesión católica*. 11. https://doi.org/10.35670/1667-4545.v17.n1.17071.

Moodley, Trevor, Karel G. F. Esterhuyse, and Roelf B. I. Beukes. 2012. "Factor Analysis of the Spiritual Well-being Questionnaire Using a Sample of South African Adolescents." *Religion & Theology* 19 (1-2): 122–51. https://doi.org/10.1163/157430112X650339.

Morejón, Antonio J VÁZQUEZ, Rosa JIMÉNEZ García-Bóveda, and Raquel VÁZQUEZ-MOREJÓN Jiménez. 2004. *Escala de autoestima de Rosenberg: fiabilidad y validez en población clínica española*. 22.

Müller, Kirill. 2025. *here: A Simpler Way to Find Your Files*. https://here.r-lib.org/.

Müller, Kirill, and Hadley Wickham. 2026. *tibble: Simple Data Frames*. https://tibble.tidyverse.org/.

Navarrete, Gorka. 2024a. *jsPsychAdmin: Admin Tasks for jsPsychR Packages*. https://github.com/gorkang/jsPsychAdmin.

Navarrete, Gorka. 2024b. *jsPsychMonkeys: Release Monkeys to a jsPsych Experiment Using the r Package targets, Docker and RSelenium*. https://github.com/gorkang/jsPsychMonkeys.

Navarrete, Gorka. 2025. *jsPsychHelpeR: Standardize and Automatize Data Preparation and Analysis of jsPsych Experiments Created with jsPsychMaker*. https://github.com/gorkang/jsPsychHelpeR.

Navarrete, Gorka, and Herman Valencia. 2024. *jsPsychMaker: Create Behavioral Experiments and Surveys Using jsPsych and r*. https://github.com/gorkang/jsPsychMaker.

Obels, Pepijn, Daniël Lakens, Nicholas A Coles, Jaroslav Gottfried, and Seth A Green. 2020. "Analysis of Open Data and Computational Reproducibility in Registered Reports in Psychology." *Advances in Methods and Practices in Psychological Science* 3 (2). https://doi.org/10.1177/2515245920918872.

Ooms, Jeroen. 2025. *writexl: Export Data Frames to Excel "xlsx" Format*. https://ropensci.r-

universe.dev/writexl.

Ooms, Jeroen. 2026. *V8: Embedded JavaScript and WebAssembly Engine for r*. https://jeroen.r-universe.dev/V8.

Pargament, Kenneth I., Bruce W. Smith, Harold G. Koenig, and Lisa Perez. 1998. "Patterns of Positive and Negative Religious Coping with Major Life Stressors." *Journal for the Scientific Study of Religion* 37 (4): 710–24. https://doi.org/10.2307/1388152.

Pedersen, Thomas Lin. 2025. *patchwork: The Composer of Plots*. https://patchwork.data-imaginist.com.

Pérez-Albéniz, Alicia, Joaquín de Paúl, and Juan Etxeberría. n.d. *Adaptación de Interpersonal Reactivity Index (IRI) al español*.

Perrier, Victor, Fanny Meyer, and David Granjon. 2026. *shinyWidgets: Custom Inputs Widgets for Shiny*. https://github.com/dreamRs/shinyWidgets.

Plante, Thomas G., and Marcus T. Boccaccini. 1997. "The Santa Clara Strength of Religious Faith Questionnaire." *Pastoral Psychology* 45 (5): 375–87. https://doi.org/10.1007/BF02230993.

Pommier, Elizabeth, Kristin D. Neff, and István Tóth-Király. 2020. "The Development and Validation of the Compassion Scale." *Assessment* 27 (1): 21–39. https://doi.org/10.1177/1073191119874108.

Pons, Gonzalo Brito. n.d. *CULTIVATING HEALTHY MINDS AND OPEN HEARTS: A MIXED-METHOD CONTROLLED STUDY ON THE PSYCHOLOGICAL AND RELATIONAL EFFECTS OF COMPASSION CULTIVATION TRAINING IN CHILE*.

Psoteg, Coden. n.d. *Decision-Making Patterns, Conflict Sytles, and Self-Esteem*.

R Core Team. 2026a. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing. https://doi.org/10.32614/R.manuals.

R Core Team. 2026b. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing. https://doi.org/10.32614/R.manuals.

R Core Team. 2026c. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing. https://doi.org/10.32614/R.manuals.

Reyna, Cecilia. 2016. "Psychometric Study of the Rational Experiential Inventory Among Undergraduate Argentinean Students." *Revista de Psicología* 34: 19. https://doi.org/10.

18800/psico.201602.005.

Rojas-Barahona, Cristian A, Beatriz Zegers P, and Carla E Förster M. 2009. "La escala de autoestima de Rosenberg: Validación para Chile en una muestra de jóvenes adultos, adultos y adultos mayores." *Revista médica de Chile* 137 (6). https://doi.org/10.4067/S0034-98872009000600009.

Ruiz, Francisco J., Diana M. García-Beltrán, and Juan C. Suárez-Falcón. 2017. "General Health Questionnaire-12 Validity in Colombia and Factorial Equivalence Between Clinical and Nonclinical Participants." *Psychiatry Research* 256 (October): 53–58. https://doi.org/10.1016/j.psychres.2017.06.020.

Ryff, Carol D, and Corey Lee M Keyes. n.d. *The Structure of Psychological Well-Being Revisited*. https://doi.org/10.1037/0022-3514.69.4.719.

Sala, Maria Nives, Paola Molina, Birgit Abler, Henrik Kessler, Leonard Vanbrabant, and Rens Van De Schoot. 2012. "Measurement Invariance of the Emotion Regulation Questionnaire (ERQ). A Cross-National Validity Study." *European Journal of Developmental Psychology* 9 (6): 751–57. https://doi.org/10.1080/17405629.2012.690604.

Sánchez-López, María del Pilar, and Virginia Dresch. 2008. "The 12-Item General Health Questionnaire (GHQ-12): Reliability, External Validity and Factor Structure in the Spanish Population." *Psicothema* 20 (4): 839–43.

Silva, Washington Allysson Dantas, Tátila Rayane de Sampaio Brito, and Cicero Roberto Pereira. 2020. "COVID-19 Anxiety Scale (CAS): Development and Psychometric Properties." *Current Psychology*, ahead of print, November 13. https://doi.org/10.1007/s12144-020-01195-0.

Sirota, Miroslav, Chris Dewberry, Marie Juanchich, Lenka Valuš, and Amanda C. Marshall. 2020. "Measuring Cognitive Reflection Without Maths: Development and Validation of the Verbal Cognitive Reflection Test." *Journal of Behavioral Decision Making*, ahead of print, October 28. https://doi.org/10.1002/bdm.2213.

Sirota, Miroslav, and Marie Juanchich. 2018. "Effect of Response Format on Cognitive Reflection: Validating a Two- and Four-Option Multiple Choice Question Version of the Cognitive Reflection Test." *Behavior Research Methods* 50 (6): 2511–22. https://doi.org/10.3758/s13428-018-1029-4.

Ståhl, Tomas, Maarten P. Zaal, and Linda J. Skitka. 2016. "Moralized Rationality: Relying on Logic and Evidence in the Formation and Evaluation of Belief Can Be Seen as a Moral Issue." *PLOS ONE* 11 (11): e0166332. https://doi.org/10.1371/journal.pone.0166332.

Stamm, B Hudnall. n.d. *Self-scoring directions Research Information on the ProQOL – CSF-vIV: Professional Quality of Life: Compassion Satisfaction and Fatigue Subscales.*

Stoll, Kathrin, Yvonne Hauck, Soo Downe, et al. 2016. "Cross-Cultural Development and Psychometric Evaluation of a Measure to Assess Fear of Childbirth Prior to Pregnancy." *Sexual & Reproductive Healthcare* 8 (June): 49–54. https://doi.org/10.1016/j.srhc.2016.02.004.

Stone, Valerie E., Simon Baron-Cohen, and Robert T. Knight. 1998. "Frontal Lobe Contributions to Theory of Mind." *Journal of Cognitive Neuroscience* 10 (5): 640–56. https://doi.org/10.1162/089892998562942.

Temple Lang, Duncan. 2026. *XML: Tools for Parsing and Generating XML Within r and s-Plus.*

Tobacyk, Jerome J. n.d. *A Revised Paranormal Belief Scale.* 6.

Tobacyk, Jerome, and Gary Milford. n.d. *Belief in Paranormal Phenomena: Assessment Instrument Development and Implications for Personality Functioning.* https://doi.org/10.1037/0022-3514.44.5.1029.

Toplak, Maggie E., Richard F. West, and Keith E. Stanovich. 2013. "Assessing Miserly Information Processing: An Expansion of the Cognitive Reflection Test." *Thinking & Reasoning* 20 (2): 147–68. https://doi.org/10.1080/13546783.2013.844729.

Ushey, Kevin, JJ Allaire, Hadley Wickham, and Gary Ritchie. 2026. *rstudioapi: Safely Access the RStudio API.* https://rstudio.github.io/rstudioapi/.

Ushey, Kevin, and Hadley Wickham. 2026. *renv: Project Environments.* https://rstudio.github.io/renv/.

Vale, Francisco A. C., Ari P. Balieiro-Jr, and José Humberto Silva-Filho. 2012. "Memory Complaint Scale (MCS): Proposed Tool for Active Systematic Search." *Dementia & Neuropsychologia* 6 (4): 212–18. https://doi.org/10.1590/S1980-57642012DN06040004.

Vaughan, Davis, Henrik Bengtsson, and Matt Dancho. 2026. *furrr: Apply Mapping Functions in Parallel Using Futures.* https://github.com/futureverse/furrr.

Waring, Elin, Michael Quinn, Amelia McNamara, Eduardo Arino de la Rubia, Hao Zhu, and Shannon Ellis. 2026. *skimr: Compact and Flexible Summaries of Data.* https://docs.ropensci.org/skimr/.

Warnes, Gregory R., Ben Bolker, Thomas Lumley, et al. 2023. *gtools: Various r Programming*

Tools. https://github.com/r-gregmisc/gtools.

Weeks, Fiona H. 2019. *Preference for Caesarean Attitudes Toward Birth in a Chilean Sample of Young Adults*. 7.

Weeks, Fiona H., Michelle Sadler, and Kathrin Stoll. 2020. "Preference for Caesarean and Attitudes Toward Birth in a Chilean Sample of Young Adults." *Women and Birth* 33 (2): e159–65. https://doi.org/10.1016/j.wombi.2019.03.012.

Wickham, Hadley. 2025a. *forcats: Tools for Working with Categorical Variables (Factors)*. https://forcats.tidyverse.org/.

Wickham, Hadley. 2025b. *rvest: Easily Harvest (Scrape) Web Pages*. https://rvest.tidyverse.org/.

Wickham, Hadley. 2025c. *stringr: Simple, Consistent Wrappers for Common String Operations*. https://stringr.tidyverse.org.

Wickham, Hadley. 2026. *httr: Tools for Working with URLs and HTTP*. https://httr.r-lib.org/.

Wickham, Hadley, and Jennifer Bryan. 2026. *readxl: Read Excel Files*. https://readxl.tidyverse.org.

Wickham, Hadley, Jennifer Bryan, Malcolm Barrett, and Andy Teucher. 2025. *usethis: Automate Package and Project Setup*. https://usethis.r-lib.org.

Wickham, Hadley, Romain François, Lionel Henry, Kirill Müller, and Davis Vaughan. 2026. *dplyr: A Grammar of Data Manipulation*. https://dplyr.tidyverse.org.

Wickham, Hadley, Maximilian Girlich, Mark Fairbanks, and Ryan Dickerson. 2026. *dtplyr: Data Table Back-End for "dplyr"*. https://dtplyr.tidyverse.org.

Wickham, Hadley, and Lionel Henry. 2026. *purrr: Functional Programming Tools*. https://purrr.tidyverse.org/.

Wickham, Hadley, Jay Hesselberth, Maëlle Salmon, Olivier Roy, and Salim Brüggemann. 2025. *pkgdown: Make Static HTML Documentation for a Package*. https://pkgdown.r-lib.org/.

Wickham, Hadley, Jim Hester, and Jennifer Bryan. 2026. *readr: Read Rectangular Text Data*. https://readr.tidyverse.org.

Wickham, Hadley, Jim Hester, Winston Chang, and Jennifer Bryan. 2026. *devtools: Tools to*

Make Developing r Packages Easier. https://doi.org/10.32614/CRAN.package.devtools.

Wickham, Hadley, Davis Vaughan, and Maximilian Girlich. 2025. *tidyr: Tidy Messy Data*. https://tidyr.tidyverse.org.

Wilke, Claus O. 2025. *ggridges: Ridgeline Plots in "ggplot2"*. https://wilkelab.org/ggridges/.

William Revelle. 2026. *psych: Procedures for Psychological, Psychometric, and Personality Research*. Northwestern University. https://CRAN.R-project.org/package=psych.

Wu, Chia-huei, and Grace Yao. 2008. "Psychometric Analysis of the Short-Form UCLA Loneliness Scale (ULS-8) in Taiwanese Undergraduate Students." *Personality and Individual Differences* 44 (8): 1762–71. https://doi.org/10.1016/j.paid.2008.02.003.

Xie, Yihui. 2025. *knitr: A General-Purpose Package for Dynamic Report Generation in R*. https://yihui.org/knitr/.

Xie, Yihui, JJ Allaire, and Jeffrey Horner. 2025. *markdown: Render Markdown with "commonmark"*. https://github.com/rstudio/markdown.

Xie, Yihui, Joe Cheng, Xianying Tan, and Garrick Aden-Buie. 2025. *DT: A Wrapper of the JavaScript Library "DataTables"*. https://github.com/rstudio/DT.