EVE Tutorial 1 - The Viking Village Forelsket

Jascha Grübel, Raphael P. Weibel, and Victor R. Schinazi

6th January 2020
v1.3

Summary

This tutorial guides through the installation and usage of the *Experiment in Virtual Environments* (EVE) framework. In this tutorial, you will create a virtual reality (VR) experiment that showcases the core features of EVE. The focus of the experiment will be on spatial knowledge and navigation. Specifically, you will be taught how to setup different types of questionnaires, how to create navigable scenes and integrate them into EVE, and how to create different types of interactions for the participants. We will also walk you through the steps required to for testing and evaluating your experiment with the tools in EVE. The tutorial is designed for beginners using the Windows operating system.

Contents

1 Introduction

Thank you for choosing EVE, the framework to run *Experiments in Virtual Environments* (Grübel et al., 2016). This tutorial is primarily directed at beginners who wish to learn how to use the core components of EVE. In this tutorial, we will guide you through the different steps necessary for setting up EVE on your machine and preparing your first experiment. The tutorial are divided into two sections. The first section covers the downloading, installation and general setup of EVE and its different components. In the second section, we will use the Viking Village scene provided by Unity in order to showcase some of EVE's capabilities for developing, running and evaluating experiments in virtual reality (VR).

The experiment tutorial focuses on spatial cognition and navigation, and does not (by any means) cover all of the functionalities provided by the EVE framework. Beyond spatial cognition, other research (Hackman et al., 2019) has used EVE to study the impact neighborhood environments on emotion and physiological reactivity. Instead, we use this particular experiment as an example of how EVE can be used and adapted to fit the demands for a specific VR research question. In the end, it is up to you to customize EVE to fit the requirements of your own research question and experiment.

1.1 Requirements

The EVE framework works within the Unity game engine environment. Before starting the tutorial, researchers are strongly encouraged to familiarize themselves with the Unity engine. A set of step-based tutorials for learning Unity can be found here. The tutorial will present many features of the Unity 3D game engine but it is not meant to be a Unity 3D tutorial.

To manage your experiment's data effectively EVE uses a database. In this tutorial the MySQL database management system is used. Note that EVE does not work without the database setup, please follow the installation guide for the database carefully.

In order to use EVE's evaluation tools, you will also be required to download **R** and RStudio. We provide you with an **R**-package called evertools that allows you to access the database from **R** and provides you with helpful scripts for the data analysis. The installation of the package is described in the evaluation section 7.2.

This tutorial is specific for Windows. We have tested EVE with Windows 10. While Unity allows you to generally develop and export the experiment for different operating systems (e.g., Ubuntu, Mac OS, iOS, and Android) we have not adapted the various background libraries (e.g., database access) to be compatible with all operating systems.

EVE runs on all major operating systems (Windows, macOS, and Linux). If you are interested in running experiments on other platforms, please contact us for further support[1].

2 Installation

This is a quick guide for installing the different software required to run EVE 1.3. Feel free to skip the parts that you are familiar with. However, some items require a specific configuration in order to work with EVE. As such, we strongly recommend that first time users carefully follow these instructions (step by step).

2.1 Unity

We recommend downloading a recent long-term support (LTS) version of Unity. Currently, EVE has been tested with version 2018.4.6f1. You can access the LTS version via the Unity Hub.

1. Download Unity Hub from here[2].
2. Install Unity Hub according to the instructions.
3. Sign in with Unity ID as shown in Fig. 1.
   - You need a valid Unity ID to use Unity.
   - A free license is enough to use EVE.

   (a) Click the *Profile (1)*.
   (b) Click on *Sign in (2)*.
   (c) In the pop-up window enter your Unity ID credentials.

[1] We are actively developing EVE on Windows and Ubuntu, so all steps within Unity should also work on Ubuntu. However, you may be required to take some extra steps in order to the database work. Contact us for the necessary files.

[2] This link goes directly to the download as for now Unity Hub is not fully featured on the website.

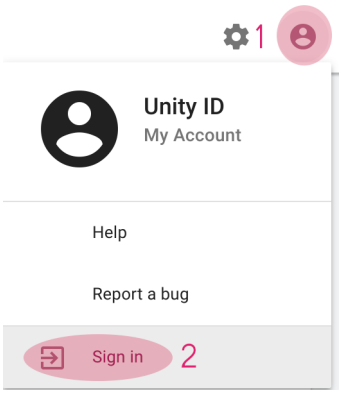


Figure 1: How to log into Unity Hub. The *Profile (1) is located in the top right corner. The* Sign in (2) at the bottom of the menu.

4. Activate a license as shown in Fig. 2.
    - Being logged in allows you to remotely activate the license
    - A pop-up with the license settings opens.

Figure 2: The license button turns blue when logged in.

5. In the license activation shown in Fig. 3, select the *Unity Personal (1)* license and state that you don't use it professionally (2).
6. Click done.

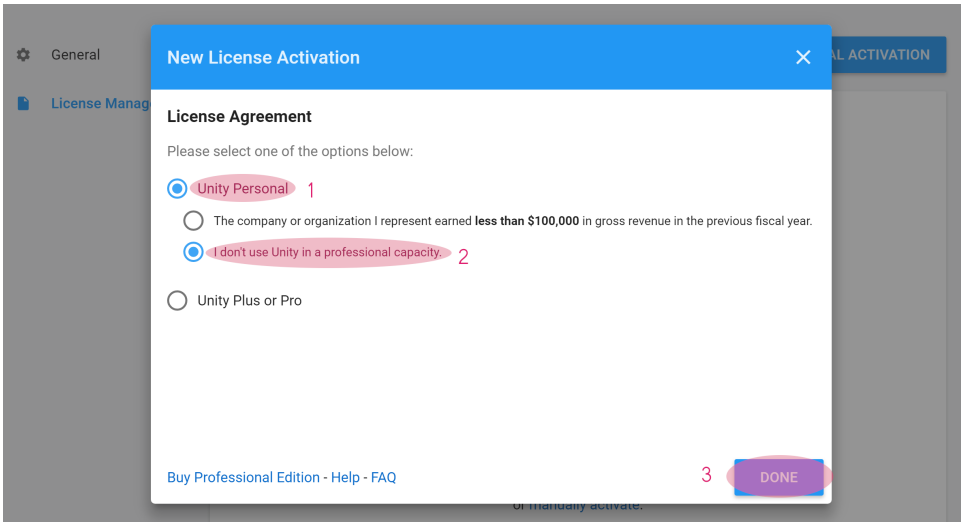


Figure 3: The activation windows allows you to select the *Unity Personal (1)* license with no professional use (2).

7. Return to the main page of Unity Hub with the little arrow in the top left corner.

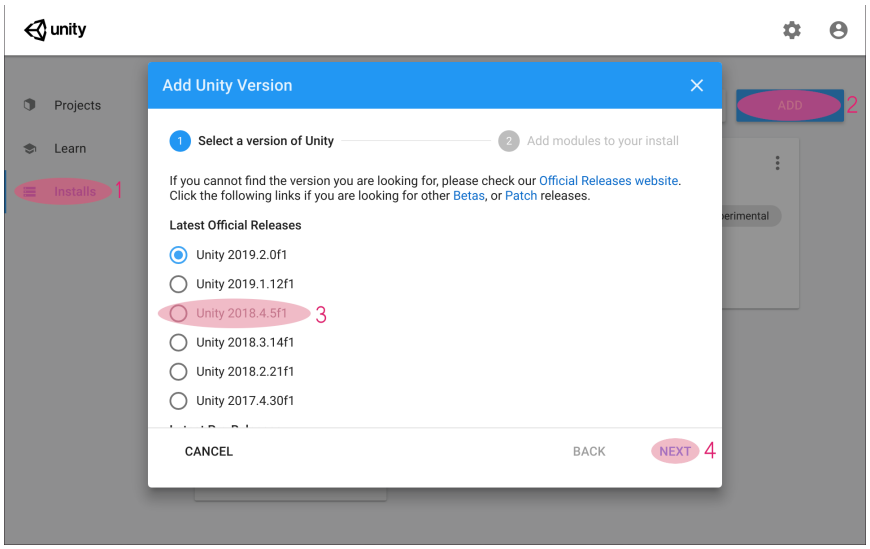


Figure 4: In the Unity Hub, *Installs (1)* allow you to see all versions of unity installed on your computer. You can *Add (2)* new versions of Unity. The long-term support (LTS) version of unity is always the last in a year or currently *2018.4.6f1 (3)*. A new version can be installed by clicking *Next (4)*.

8. In Unity Hub, shown in Fig. 4, click on *Installs (1)*.

9. Then click *Add (2)* to add a new unity version.

10. In the pop-up menu select version *Unity 2018.4.6f1 (3)* as it corresponds to the current LTS version.

11. Click *Next (4)* and follow the instructions using the default settings to complete the download and installation of Unity.

    - Note that if you don't have Visual Studio installed, it is highly recommended to take along the community version of Visual Studio so you can edit code easily with a good integration into Unity.

You may also try to use EVE with a newer unsupported version of Unity which should generally be possible. However, we cannot guarantee that new features of Unity won't break EVE. We will regularly update the EVE for compatibility with the newest long-term support version of Unity. The next update will come after Unity LTS 2019 is released in early 2020.

2.2 MySQL Database

Before you can install EVE, you are also required to install a database. EVE uses a database in the backend to manage the collected data. The database is what allows you to securely store and retrieve your data. Advanced users can change the type of database used (e.g., PostgresQL) although we currently only offer support for MySQL 8[3].

The installation type we recommend includes a program called "MySQL Workbench". This program makes interactions with the database simpler for beginners. Please note that these are just the minimum steps required to install MySQL on Windows. If you have previous knowledge with MySQL, you might want to change some of these settings. In case you run into to any problems during these steps, please check the Appendix A for a list of known issues. To download MySQL 8 follow these instruction:

1. Go to https://dev.mysql.com/downloads/mysql/.

2. Scroll down to the General Available (GA) releases and if necessary select "Microsoft Windows" (1) as in Fig. 5.

[3]We have implemented an agnostic `DatabaseConnector` for EVE with derived classes for specific databases. Currently, there is only a `MySqlConnector`. Switching the database requires the implementation of a new connector.

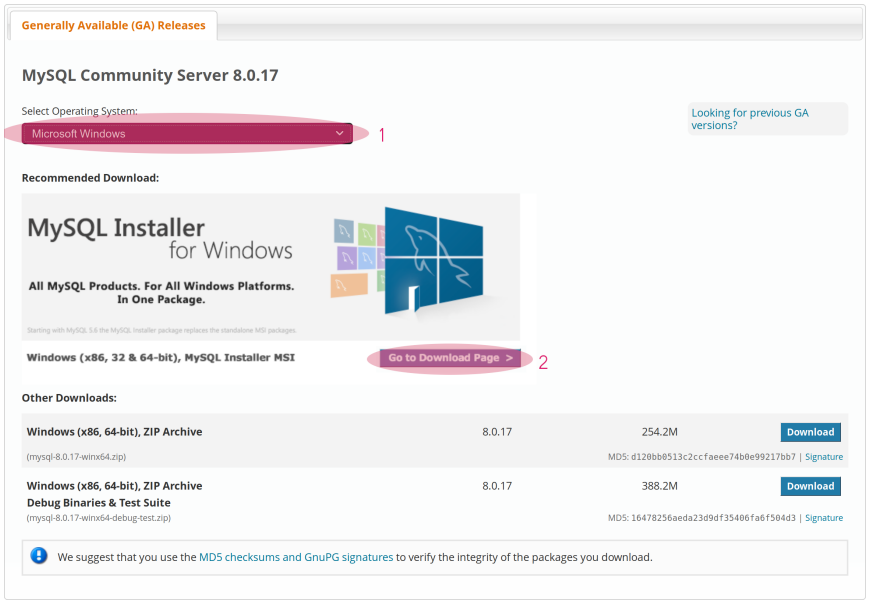


Figure 5: The MySQL homepage for windows.

.

3. Click "Go to Download Page" (2) as in Fig. 5.

    - Note that the downloads below are only for mysql and do not include the work bench.

4. Scroll down to the General Available (GA) releases and if necessary select "Microsoft Windows" as the platform.

5. From the list, select the mysql-installer-community installer (use the larger file size without web in file name) as shown in Fig. 6.

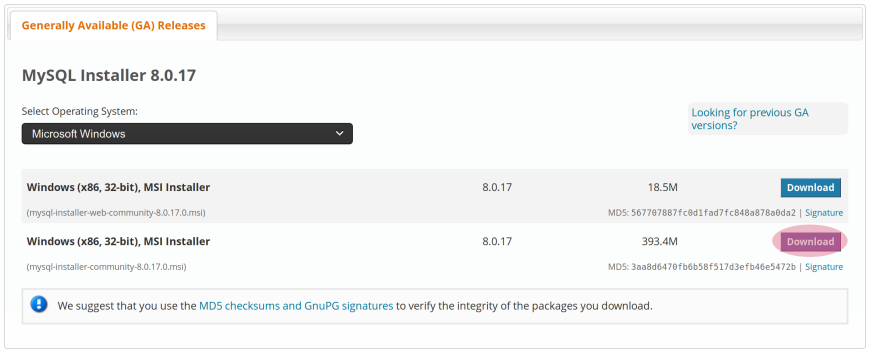


Figure 6: Download the community installer highlighted in red.

6. If you use MySQL only for EVE, choose the option “No thanks, just start my download.” as shown in Fig. 7

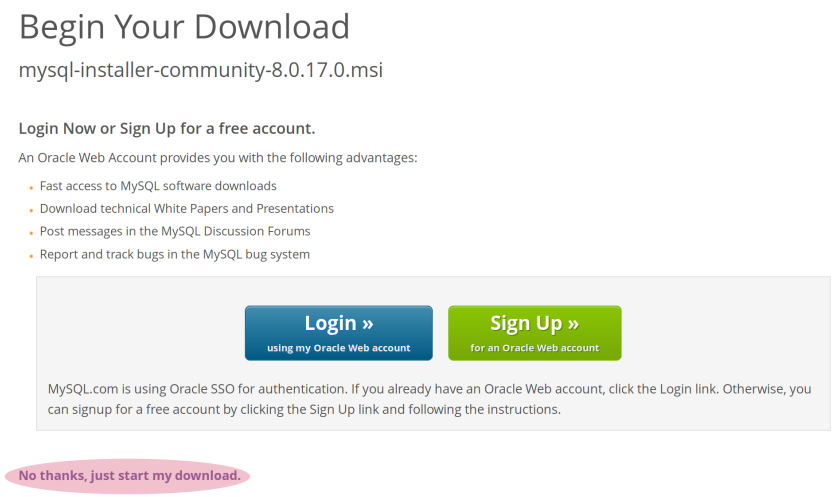


Figure 7: Skip the login to download without account.

7. Open the downloaded file “`mysql-installer-community-...`”
8. Go through the installation process:
   (a) Choose “Custom” as the “Setup Type”.
   (b) Make sure that the list on the right is empty before you start adding items. The list can be cleared using the double arrow from right to left that is located between the two lists.
   (c) Select the following items for installation as shown in Fig. 8. Use the arrows between both lists to move items from left to right.
      i. MySQLServers > MySQLServer > MySQLServer 8.0 > `MySQLServer 8.0.17 - X64`
      ii. Applications > MySQL Workbench > MySQL Workbench 8.0 > `MySQL Workbench 8.0.17 - X64`

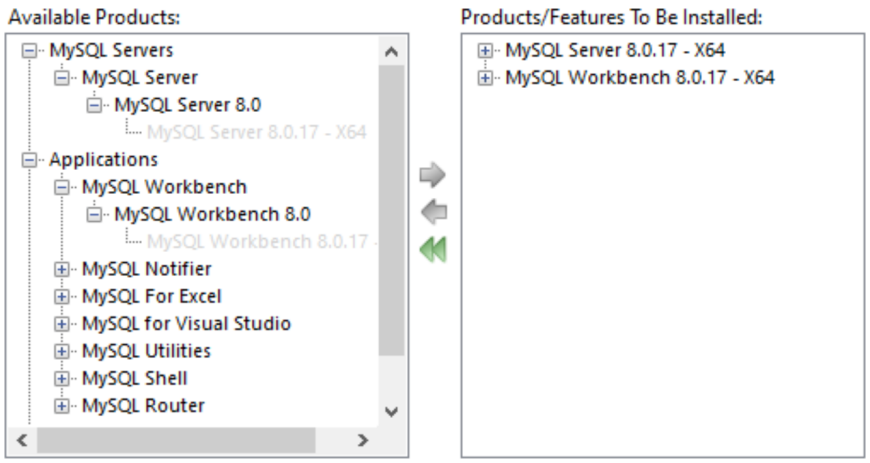


Figure 8: The recommended selection for the MySQL installer.

(d) The installer might show some missing requirements.
- These should be fixed by the installer.
- If asked, click "Yes" to continue.

(e) The installer will execute.
- After the installation, MySQL needs to be configured. We will only choose standard settings.

(f) Leave the "High Availability" settings on default and click "Next >".

(g) Leave the "Type and Networking" settings on default and click "Next >".
- Note that these settings might open up access to the database through your local network.

(h) Leave the "Authentication Method" settings on default and click "Next >".

(i) In the "Account and Roles" settings you can select any secure Root Password you like. Write it down for later use.

(j) On the same page select to "Add user" (1) as shown in Fig. 9.
   i. Choose a "User Name" (2) you will later use in EVE.
   ii. Leave the Role as "DB Admin".
   iii. Select a secure password (3) and write it down for later use in EVE.

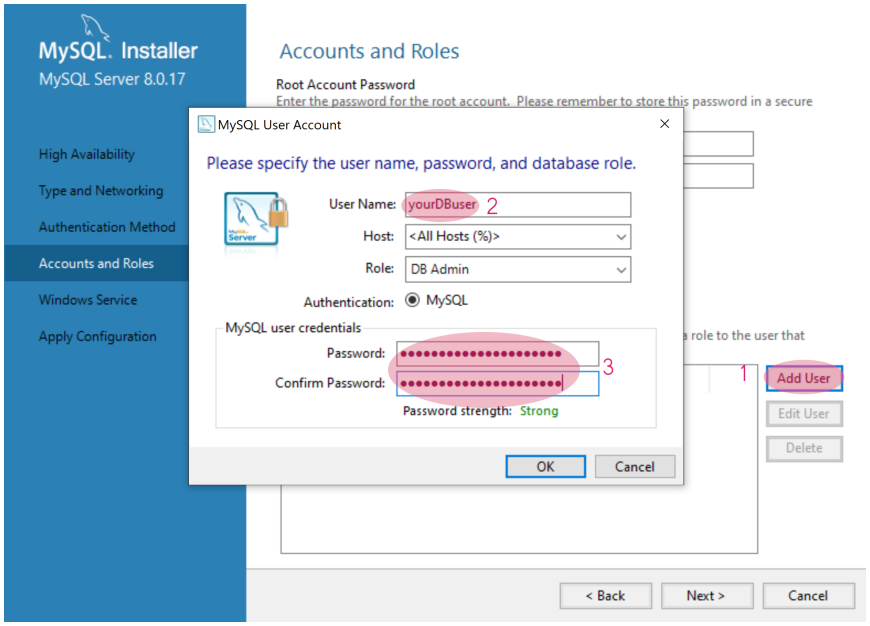


Figure 9: The *Add (1)* button opens the dialog to add a user to the database. The user will later access the database from EVE.

(k) Leave the "Windows Service" settings on default and click "Next >".

(l) On the "Apply Configuration" page, click "Execute" to finish the setup of MySQL.

(m) If all configuration steps completed successfully, click finish.

9. The remaining installation steps should be self-explanatory. Continue until the end of the setup. When the setup is complete, it will automatically start a program called "MySQL Workbench".

10. In MySQL Workbench you should see one connection as shown in Fig. 10

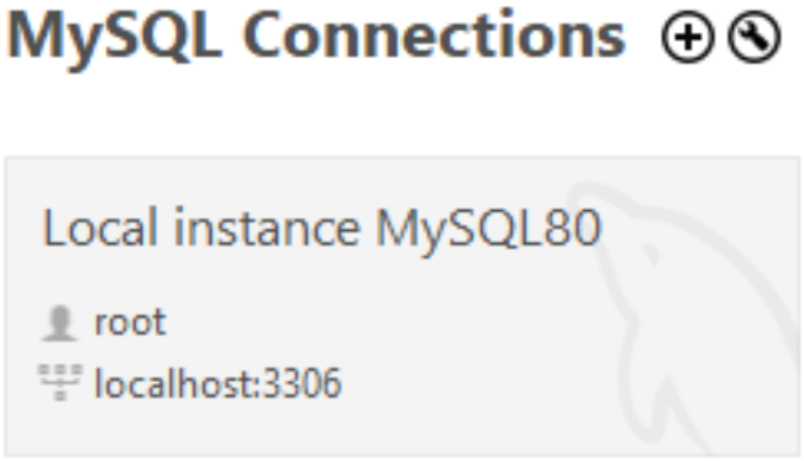


Figure 10: Home screen in Workbench with the local database.

(a) Click on the connection.
(b) Click on "Server status" on the left side as shown in Fig. 11

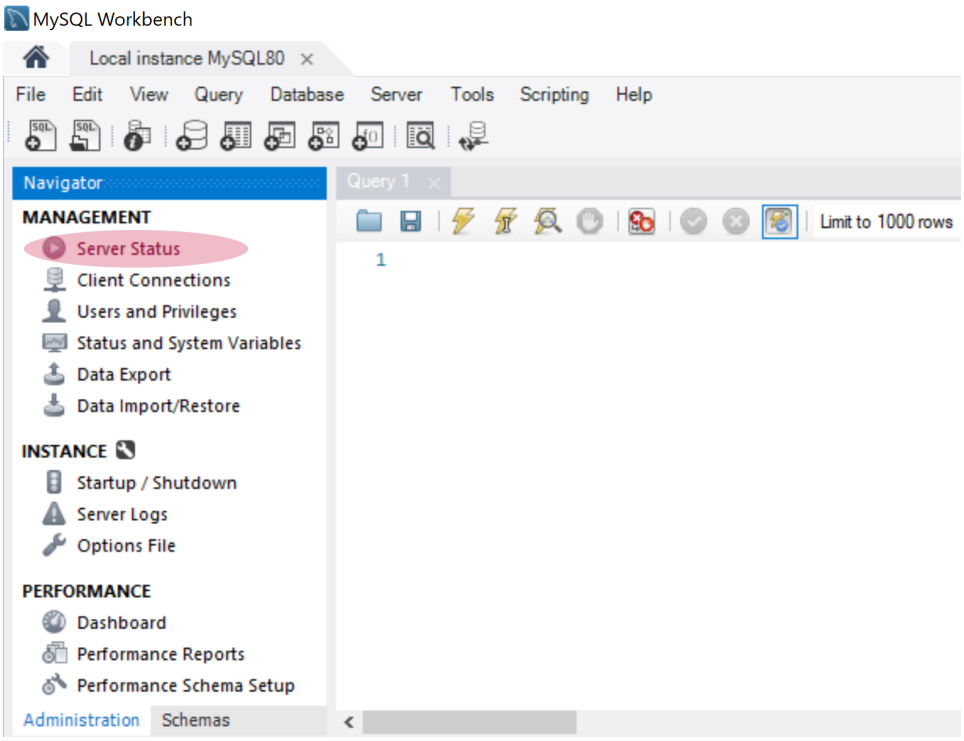


Figure 11: Status screen in Workbench with the local database.

(c) Check the server status on the right side. It should be on "Run-

ning" as shown in Fig. 12.

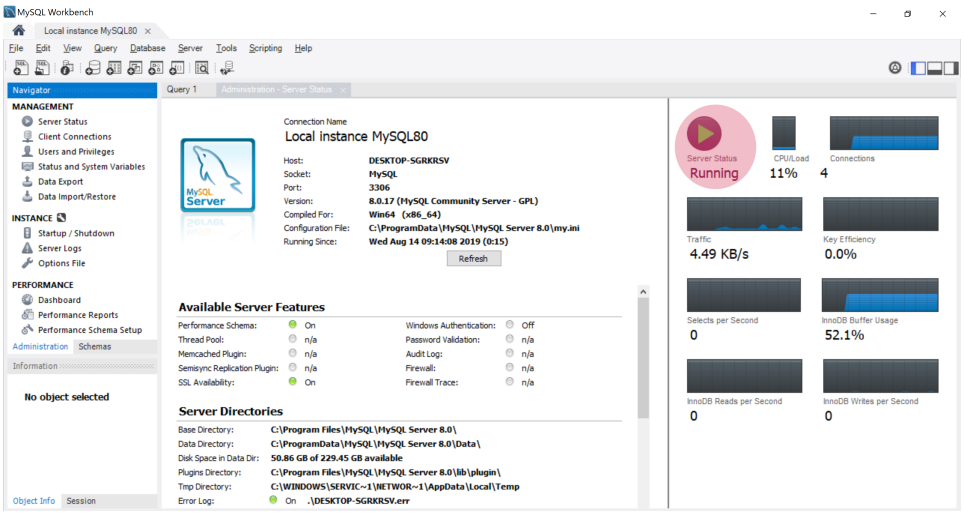


Figure 12: The status overview of the MySQL server in Workbench.

i. If the server is not running, click on "Startup/Shutdown" on the left side and wait for the server to shut down.

ii. Then click on the "Start Server" button on the right side and wait for the server to start up.

(d) Close "MySQL Workbench".

2.3 R and RStudio

EVE comes with an accompanying **R**-package called "evertools". Evertools allows you access information in the database without having to write SQL commands. To use evertools you have to first install **R**, followed by RStudio, and lastly if you are on Windows RTools.

2.4 EVE

Once you installed Unity and MySQL, there are two ways of downloading EVE. You can either download the entire EVE project and use it locally or you can synchronise with our public EVE git repository to contribute and improve EVE. For individuals with a computer science background, we recommend the second option (git synchronization) because this option will allow you to create and save backups of your project on github. If you do not have a computer science background or are trying out EVE for the first time, we suggest using the local install option. In later tutorials, we will introduce you to git.

2.4.1 Installing stand-alone version of EVE

Installing EVE as a stand-alone will provide you with all the functionalities of EVE. However, you will not be able to easily update EVE as new features are developed. The following steps will guide you through setting EVE up as a standalone.

1. Click the "Download ZIP File" button on the EVE website.

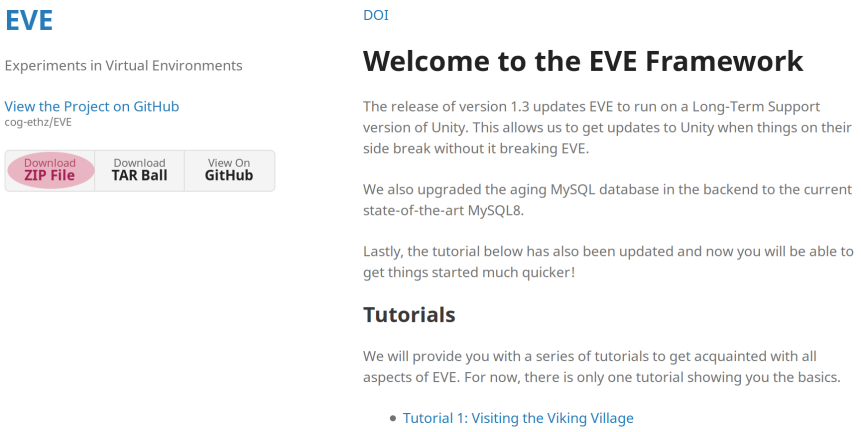


Figure 13: Download section of the EVE website.

2. Unpack the .zip into your preferred location. Note that this is already the Unity project and you will always work within this folder as long as you use EVE.

3. Download the large files. Note that it is important to do this *before* you open EVE in Unity[4].

   (a) Download the LargeFiles.zip.

   (b) Unpack the LargeFiles.zip into a temporary destination.

   (c) Copy the `Assets` folder you unpacked into the root of the Unity project where you already see a folder named `Assets`.

4. You are ready to configure EVE.

2.4.2 Getting to know the Unity Editor layout

The Unity layout can be adapted in different ways. As such, the layout described here might differ from the layout in your machine but the contents of the views will be the same. You can use our example image shown in Figure

[4]If you opened the project in Unity before, you need delete the project and unpack it again to have a clean slate.

14 as a guide to find the different views. We numbered different elements in the Figure and we will refer to these numbers throughout the instructions for this tutorial. For more information about each window, we encourage you to check out the official Unity manual describing the different windows. Please also look at the official unity manual for the console, since this is where errors and information are displayed in Unity.

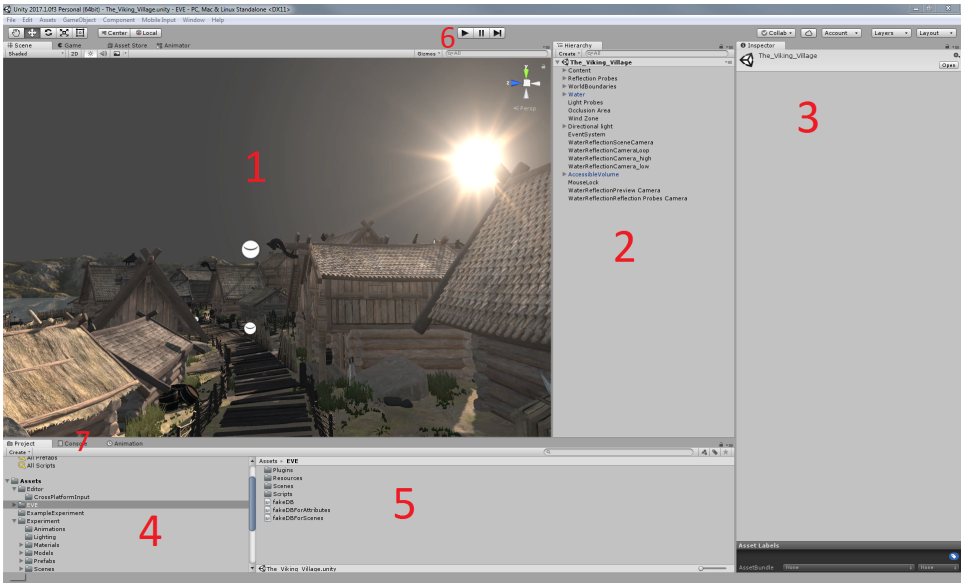


Figure 14: Unity editor layout. (1) Scene view; (2) Hierarchy view; (3) Inspector view; (4) Project view (folders); (5) Project view (files); (6) Play button; (7) Tabs to switch between the Project view and console.

2.4.3 Configuring EVE

Before configuring EVE, please make sure that you have properly installed all of the required software. You need Unity, MySQL and a copy of EVE on your local machine. To configure EVE:

1. Open Unity Hub again.
2. On the left side, click on *Projects (1)*.
3. Add EVE from your computer as shown in Fig. 15.

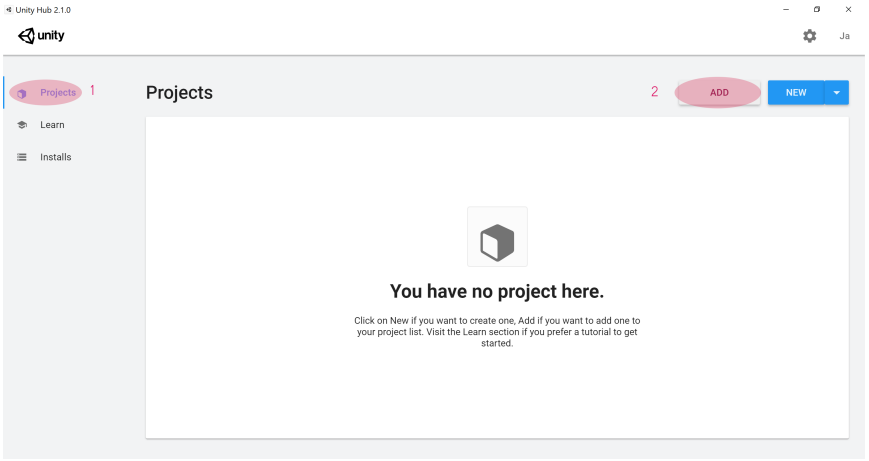


Figure 15: *Projects (1)* gives you an overview of all active unity projects. You can *Add (2)* existing projects or create new projects.

(a) At the top right, click on *Add (2)*.
(b) Find the folder where you stored the local copy of EVE.
(c) Click on the EVE folder.
(d) Click the "Select folder" button.
(e) Accept any possible warnings regarding upgrades to files.

- If you are using a version of Unity that is different from the one we recommend, you might get a warning. The warning states that your version of Unity and the project's version of Unity differ. Ignoring the warning may cause errors. In order to guarantee all functionalities of EVE, we recommend changing (upgrading or downgrading) your Unity installation to the recommendation.
- You might see a warning indicating that MiddleVR was not found but since we are not using MiddleVR in this tutorial, press the "OK"" button to continue.

4. Wait for the Unity editor to fully load.
5. Find the Project view and go to `Assets\Experiment\Resources`
6. Double-click the file `experiment_settings.xml` to open.
7. The file will be opened in Visual Studio 2017[5]. If you are want to use another text editor:

   (a) Right click the file and choose "Show in Explorer".

[5]If you have trouble opening it in Visual Studio, the link between unity and visual studio may be incomplete. Refer to this guide for help.

(b) The folder containing the file opens.

(c) Choose your preferred text editor to edit the file.

8. The file contains basic information describing an experiment. At this point, we are only interested in changing the database settings. Find the part of the file that looks as follows:

```
<DatabaseSettings>
    <Server>127.0.0.1</Server>
    <Schema>eve</Schema>
    <User>yourDBuser</User>
    <Password>your.s4ve.DB.password</Password>
</DatabaseSettings>
```

9. Change the entries User and Password, to the user and password that you setup during the MySQL installation. Save the file and close Visual studio.

   (a) If you use linux, you need to name the scheme *EVE* in capital letters as mysql is case sentive on linux.

10. In the Project view (4 in Fig. 14), go to `Assets\ EVE\ Scenes` and then double click on Launcher. If you are unfamiliar with the Unity Editor Layout, please refer to the next section of this tutorial where we discuss the Unity layout (see Figure 14).

11. Press the Play button (6 in Fig. 14):

    (a) You should get an error message: "Error: Unable to connect to the database! Press ok to check the database status".

    (b) Press "OK".

    (c) If everything was setup correctly you should see the following:

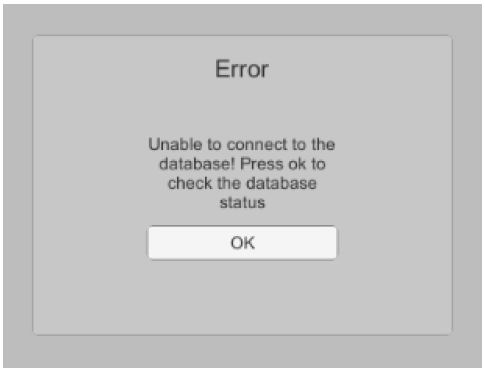


Figure 16: On the first start of EVE, the database needs to be initalized and this waring pops up to start the procedure.

12. Press the Play button (6) in the Unity editor again to stop the execution of EVE.

3 Exploring EVE

This section gives you a quick inside into the core structure of the setup of EVE in Unity. It should help you to navigate the folders and how to find things that are imporant.

EVE's main task is to manage your data and to provide you with building blocks to quickly create experiments. When you open the Project view (4), you can see the two most important folders: `EVE` and `Experiment`. The former contains the code to run EVE and many prepared objects that you can use in your experiment. The latter is a placeholder for you to store your new experiment specific code and data. Both follow a similar structure where there is a folder for `Resources`, `Scenes` and `Scripts`.

The `Resources`-folders have a special function in unity and you can read more about it here. Via the resources API in Unity, you can load objects from the `Resources`-folder and therefore images, prefabs, videos and other resources to be loaded during run-time should be stored in `Experiment \Resources`. Standard resources, that we believe should be available for every experiment can be found in `EVE \Resources`.

The `Scenes`-folders help you to keep track of the scenes in your experiment. If the scene is provided by EVE, you should look for it in `EVE \Scenes`. If you created the scene yourself, you should store it in `Experiment \Scenes`.

The `Scripts`-folders contain all `C#` code. For the scope of this tutorial, we will not write code ourselves. If you write components yourself, we recommend you store them in `Resources \Scripts`. The framework core is located in `EVE \Scripts` and editing this may cause EVE to stop function-

ing. We only recommend editing here if you know what you are doing. If you want to contribute new core features we strongly encourage you to engage us on the issue tracker on github. There, we can discuss how to integrate your work into the framework and how to make it available for other scientists to use.

4 Setting up a basic questionnaire only experiment

Now that EVE is configured, we are redy to set up our first experiment. We will start by setting up a questionnaire only experiment, and later extend this experiment to include virtual environments. In future tutorials, you will learn how to expand questionnaires on your own.

4.1 Preparing the questionnaires

We will use two different questionnaires: A basic demographic questionnaire and the Santa Barbara Sense of Direction questionnaire (Hegarty, Richardson, Montello, Lovelace & Subbiah, 2002).

- The demographics questionnaire is designed to collect basic information about the participants conducting the experiment. Demographics questionnaires vary in depth and scope but typically include items such as age, gender and handedness. It is entirely up to the researcher to decide what items to include in the demographics questionnaire. For the purpose of this tutorial, we included a question about gender, age and two questions about occupation.

- The Santa Barbara Sense of Direction Questionnaire (SBSOD) is a is a self-report questionnaire composed of 15 items used to measure environmental spatial ability. Each item is rated on a seven-point Likert scale (Likert, 1932) from "strongly disagree" to "strongly agree". Some items are reverse-scored in order to account for response bias. We provide the SBSOD as a "ready-to-use" questionnaire combined with the demographic questions.

In EVE, questionnaires are based on question sets. Question sets are a "meaningful" collection of questions. The idea is that question sets can be reused in different questionnaires without having to be re-created. We will use two question sets, one containing all the SBSOD questions and one containing our 4 demographic questions.

In this tutorial, we provide both question sets and questionnaires. Typically, before we can add the questionnaires in EVE, we have to create an xml file that tells EVE which question sets are part of which questionnaire. Creating these questionnaire and question sets will be covered in section 5.8.

4.2 Creating a questionnaire experiment in EVE

In this section, we will show you how to add a questionnaire scene to a experiment, save the settings, add all the questions to the database, and test the experiment.

1. Open Launcher Scene
   (a) In the Project view (4), go to `Assets\EVE\Scenes`
   (b) Double click on Launcher in the Project view for files (5).
      - Note: any time you open up the Launcher scene and switch to a different scene afterwards (or close Unity), Unity will ask you if you want to save the changes (even if you did not change anything). Unless you actually changed anything, you do not have to save the file.
2. Press the Play button (6).
   - This will once again open up the EVE main menu.
3. In the main menu click on "Configuration".
   - This will open the configuration menu of EVE.
4. Click on "Scene Setup".
   - The "Scene Setup", consists of two lists. On the left side are scenes that have been added to the experiment in execution order, and on the right side are scenes than can be added to experiment.
5. Remove the ExampleQuestionnaire from the experiment.
   - The two buttons next to a scene entry are for deleting the scene from the left list (red "X" button) or moving the scene up in the hierarchy ("arrow up" button).
   (a) Click on the red "X" next to `ExampleQuestionnaire.xml` to delete the scene.
6. Add new scenes to the list of available scenes on the right.
   (a) Click on the asterisk symbol "*" button at the bottom right corner of the menu.
      - This will open a windows explorer window.
   (b) Navigate to your EVE folder[6] and then go to `Assets\ EVE\ Resources\ Questionnaires`.

[6] folder where the EVE project is located on your computer.

(c) Press the open button on the windows explorer window.

- The `sbsod_lite.xml` will now appear in your right list.

7. Add the questionnaire scene to the experiment on the left.

   (a) Press the little green "+" button next to `sbsod_lite.xml`.

   - You will now see the `sbsod_lite.xml` entry in the right list.
   - Note that we added both question sets to this questionnaire for your convenience.

8. Save setting to store new scenes in experiment.

   (a) Press "Return to Config Menu" Button.

   (b) In the configuration menu, press the "Save settings"

   - A Windows explorer window will open.

   (c) Navigate to `Assets\ Experiment\ Resources`.

   (d) Save the file as `experiment_settings.xml` and replace the previous file.

9. Add the questions to the database.

   (a) Restart EVE[7].

   - Stop the execution by clicking the Play button (6).
   - Wait until you are back in the scene view.
   - Start the execution of the Launcher again by clicking the Play button (6) again.

   (b) In the main menu, select the "Database connection" button.

   (c) Press the "Add questions" button.

   - Wait for the green text "Questionnaires found (see log" to appear next to the text "Questionnaire exists:".

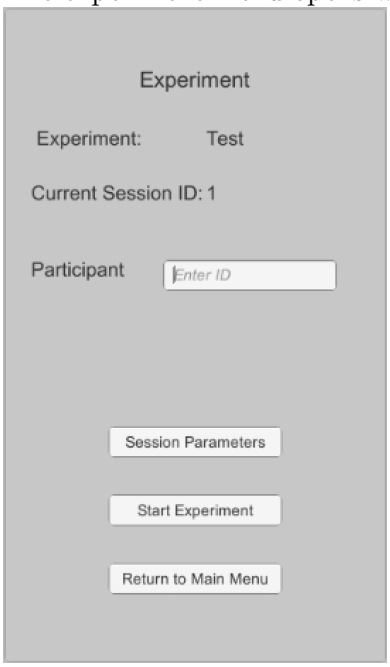
   (d) Go back to the main menu.

10. Test the experiment.

    (a) At the top of the menu, press the "Experiment".

[7]Unity can only read files as they were before unity started execution. We need to restart in order for the new settings to be readable. Otherwise we cannot find the questionnaires.

- The experiment menu opens with the following options:



  - The name of the experiment (which can be set in the xml file used in the database setup).
  - The current session id (which will be increased by one every time you start a new experiment session)
  - The participant identifier allows you to uniquely mark each participant.
  - A "Session Parameters" button where you can set the parameters of the experiment.
    - For this experiment no parameters have been added yet, so this list is empty.
  - A "Start Experiment" button which will start the experiment.
  - A "Return to Main Menu" button.

(b) Type in an identifier into the subject ID field.

(c) Press the "Start Experiment" button.

(d) Wait for another "Start" button to appear.
  - This button will start the actual experiment.
  - The extra step is used to keep the setup information hidden from the participants.
  - Bonus information: It is best to prepare everything up to this point before the participant arrives.

11. Press "Start" to begin the experiment with the questionnaire.

12. Fill out the questionnaire.
13. After you finished, a small "end of experiment" message appears.
14. Press the "OK" button to return to the main menu.
15. Stop the execution by clicking the Play button (6).

In this section you learned how to add scenes to an experiment, save an experiment, load questions from the database and start an experiment. In short, you have successfully setup and tested a small questionnaire experiment with EVE. Let's continue with an actual VR experiment.

5 Setting up a full VR experiment

So far you have learned how to setup a basic experiment using only questionnaires. In this section, we will guide you through the setup of an full experiment using a virtual tourist Viking village. The Viking Village can be downloaded as a free asset in the Unity Asset store. Since the village only exists as a sample project, whose setup interferes with the EVE project, we provide the extracted necessary assets (objects) from the project and provide it as a separate download. In the following paragraphs, we present the design of the experiment followed by the individual stages needed for for setting it up within EVE.

5.1 Experiment design

This sample experiment investigates the effect of stress (i.e., time pressure) on the acquisition of spatial knowledge. We want to test the hypothesis that under stress the acquisition of spatial knowledge is limited. To that end, we can compare two groups, with and without time pressure, to discern the effect of stress on the acquisition process.

A basic spatial knowledge acquisition task is used. Participants should learn spatial relations within the environment by visiting landmarks. As participants explore the environment they slowly built up a spatial representation. According to our hypothesis, this process should be hindered by having to find locations under time pressure. Figure 17 presents a map of the Forelsket Viking Village with eight different locations or landmarks. The name of each location is indicated by a sign in the virtual world to draw the participants' attention.

The experiment consists of a learning phase and testing phase. In the learning phase, participants will use the mouse and keyboard to navigate the Viking Village and learn the position of eight different locations. During learning, participants will be allowed to freely explore the environment and will not be given a time limit. In the testing phase, participants will be asked

Figure 17: Map of the Forelsket Viking Village with eight locations. The map will be used as part of this tutorial in the design and implementation of navigation studies.

to retrieve these locations in a specific order. The name of the each location will be presented on the screen and participants will have to implement the shortest path to reach their destination. Participants will complete one such trial consisting of learning and testing[8]. At the end of each trial, participants will be asked to perform a series of judgements of relative direction (JRD; for a in-depth discussion see Kozlowski & Bryant, 1977; Bryant, 1984; Shelton & McNamara, 2001; Waller & Hodgson, 2006; Marchette, Vass, Ryan & Epstein, 2014; Vass & Epstein, 2017; Huffman & Ekstrom, 2019).

Previous research (Schinazi, Nardi, Newcombe, Shipley & Epstein, 2013; Weisberg, Schinazi, Newcombe, Shipley & Epstein, 2014) has shown a positive relationship between JRD performance and the acquisition of spatial knowledge. Since JRD tasks are notoriously difficult to implement in VR, we provide two variations of the task as alternatives to researchers. Selecting the most appropriate JRD will depend on the research goals.

For JRD1, participants will be teleported from the Viking Village to an empty scene with a circle drawn on the floor (see Figure 18). A set of instructions will then be presented on the screen. Participants will be asked to imagine a location and facing direction before using the mouse to point

[8]To look properly at the effect there should at least three trials. It is up to you to implement trial repetitions.

a crosshair to a goal location. Participants will not receive any feedback on their performance.

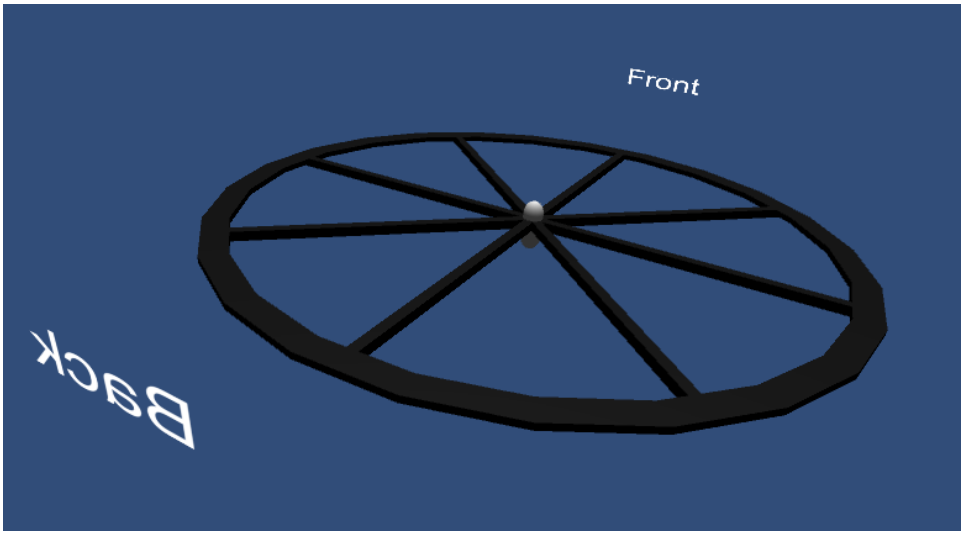


Figure 18: The 3D JRD1 task viewed from the side. The text "front" will be replaced with a location name. The text "back" helps participants to know when they turned 180 degrees.

In the second JRD test (JRD2), participants will be presented with a circle on the screen and an arrow that indicates their position and facing direction as shown in Figure 19. Participants are asked to imagine a location and facing direction and then drag and drop the image on the right side in one of the seven boxes around the circle. Participants are not be given any feedback regarding their performance.

Before starting the experiment, participants will be required to complete a demographics questionnaire and the Sanata Barbara Sense of Direction Scale. Participants will also have a chance learn to use the control interface (i.e., mouse and keyboard) and practice their skills in a maze environment.

Training with the control interface (e.g., joystick, mouse and keyboard) is an important part of VR experiments because it allows researchers to reduce possible confounds between navigation performance and individual differences in the ability to use a control interface (see Grübel, Thrash, Hölscher & Schinazi, 2017).

5.1.1 Experiment Protocol

Figure 20 below presents the different stages of the experiment. In the paragraphs that follow we will guide you through each of these stages. Each stage can be developed separately and later assembled by EVE in the desired order.

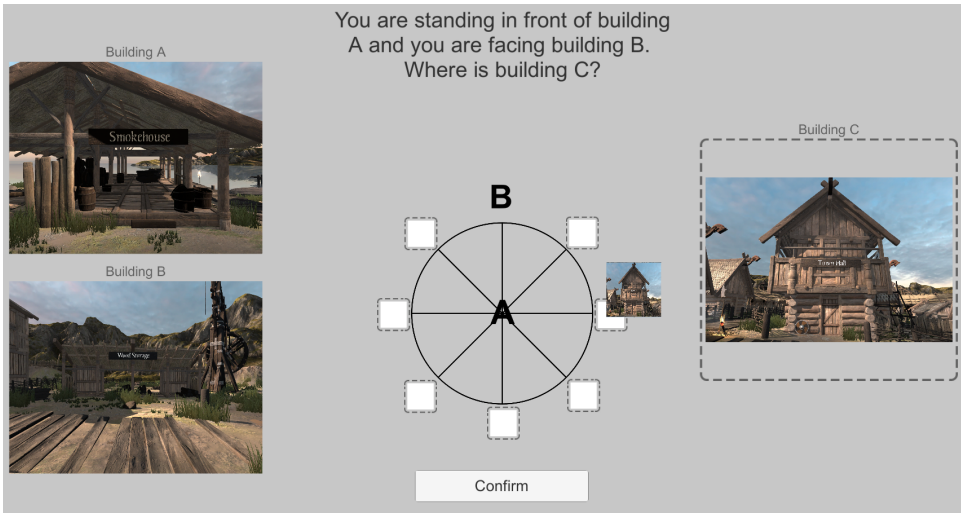


Figure 19: The JRD2 task. Participants are asked to imagine a location and facing direction and then drag and drop the image on the right side in one of the seven boxes around the circle. .

A proper experimental protocol is crucial for the proper execution of an experiment. A protocol ensures that all participants are guided that same way through the experiment stages. While Figure 20 gives you a highly abstracted idea of the experimental protocol, a detailed protocol for a study with physiological measures can be found in Weibel et al. (2018). *We strongly encourage you to produce a similarly detailed protocol for your experiment.*

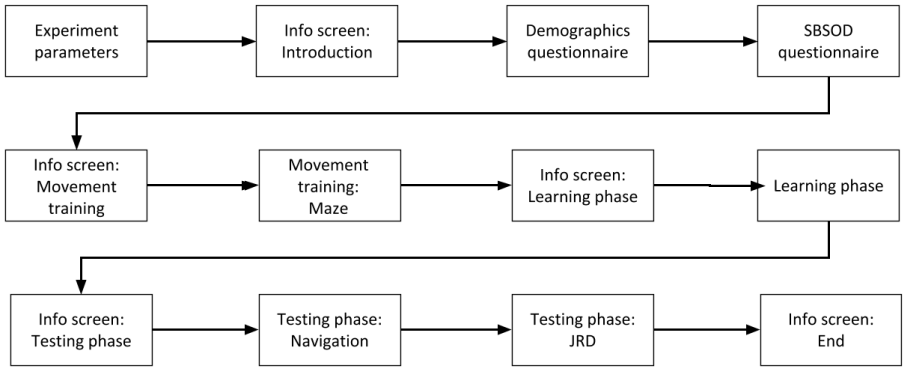


Figure 20: Experiment protocol depicting the different stages of the experiment.

5.2 Downloading experiment scene

In order to prepare the experiment scene, download the Viking village here. Extract the containing files of the zip-folder into the `Experiment` folder in EVE. You should have the following folders inside `Experiment` after you are done: `Lighting`, `Resources`, `Scenes`, `Scripts`, `Shaders`, `Standard Assets`, and `Textures`. If you choose to setup a different file hierarchy (e.g., by copying the zip folder `The_Viking_Village` into the experiment folder rather than its subfolders) you will have to adapt the path indication given in this tutorial accordingly.

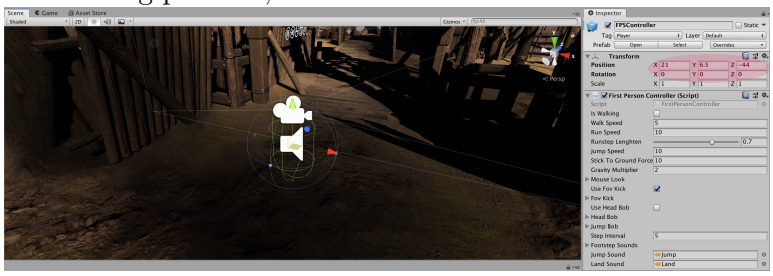
1. In Unity, wait until all files are imported.
2. Open the `The_Viking_Village`.
   (a) In the Project view (4), go to `Assets\Experiment\Scenes`
   (b) In the Project view (5), double click on the scene `The_ Viking_ Village.unity` (the file not the folder).

5.3 Setting up basic movement

In order to setup a virtual environment, it is helpful to have a way to "walk around" the environment. Therefore, the first step will be to add a character controller (from a first person perspective) that can be controlled by the user to navigate the virtual environment. EVE provides you with a character controller that can be dragged and dropped into the scene. As you will

learn later, the controller you place here will be removed once you are done setting up the virtual environment. This is because EVE uses a persistent controller which will placed in all the virtual environments during runtime (see section 5.4.3).

1. Go to the project window (4 in Figure 14)
2. Go to `Assets\EVE\Resources\Prefabs\Player\`.
3. Find the FPScontroller in the right side of the Project view (5).
4. Drag and drop the FPSController into the Hierarchy window (2).
5. Go to the Inspector window (3), and check the box that is next to the name "FPSController".
    - This will activate the controller in the scene.
6. Place the controller at the starting position of the experiment.
    - Find the FPSController in the hierarchy window (2) and double click it.
    - This will center the scene view (1) on the controller.
        - Once the view is centered on the controller, you can look around in the scene view (1) by holding down the right mouse button and moving the mouse in any direction. As long as you hold down the right mouse button, you can also use the the W A S D keys to move forward (W), backwards (S), left (A) and right (D). For more information please also check this official unity manual of the scene view.
    - There are two ways to change the position of the controller (or any object) in Unity:
        - Option 1: Set the absolute position using the Position and Rotation fields in the `Transform` component located at the top of the Inspector view (3).
        - Option 2: Manipulate the position inside the Scene view (1). Check the official unity manual on positioning objects.
    - Please use any of these ways to move the character controller to the starting position, which is shown below:



7. Press the Play button (6).

    - To maximize the Game view, you can press the "Maximize on Play" button in the top right of the Game view. The next time you press play it will automatically maximize the game view.
    - For more information about the game view please check the official unity manual on the game view.
    - Please note, that when you play any virtual environment with EVE objects inside there will be some error messages in the Console view (7). This is because without the launcher some parts of EVE are not properly configured. *This is okay for testing scenes, but when you want to test your experiment logic you have to use the launcher to get there.* You should only worry when you get error messages while running the virtual environment through the EVE launcher.

8. Walk around in the virtual environment to explore.

    - Once the game view has fully loaded, you can navigate the virtual environment using the W A S D keys (or the arrow keys) and your mouse to look around.
    - Take your time to explore the environment and once you are done you can continue with the tutorial.

9. Press the escape key on your keyboard to make your mouse visible again.

10. Stop the execution of the game view by pressing the Play button (6) again.

5.4 Setting up the learning phase

We will now setup the virtual environment for the learning phase of the experiment (section 5.1).

5.4.1 Add a non-diagetic user interface (UI)

First, we will add the different User Interface (UI) elements. To inform the participant of their progress in the task, we use non-diagetic elements (Fagerholt & Lorentzon, 2009) in the form of a Head Up Display (HUD). Non-diagetic elements are not part of the virtual environment but are situated in an overlay on top of the virtual environment (Fagerholt & Lorentzon, 2009). For the learning phase this includes displaying a list of locations that need to be found by the participants.

1. Go to `Assets\ EVE\ Resources \Prefabs`.

2. Drag and drop the `HUD_Canvas_4_panels` from the Project view (5) to the Hierarchy view (2).

3. In the Hierarchy view (2), double click the `HUD_Canvas_4_panels`.

   - This makes the scene view focus on the object.

4. In the Inspector view (3), press the little triangle to the left of the `HUD_Canvas_4_panels`

   - This reveals the children of this object. There are five children in this object.
   - Each child is a panel that allows you to display information at different locations of the screen: top, bottom, left, right or center of the screen.

5. Reveal the children of the `TopPanel`, by clicking on the little triangle to the left of it.

   - There are two items inside: `Timer` and `Money`.
   - Timer is used to show the amount of time left, whereas Money is used show the amount of money left.
   - In the learning phase we don't need the money counter or the timer.
   - The money counter is deactivated by default, that is why the entry in the hierarchy is greyed out.

6. Deactivate the `Timer` (which is activated by default).

   (a) Click on the object in the Hierarchy view (2)

   (b) Uncheck the box on the top left corner of the Inspector view (3).

7. The `BottomPanel` and `LeftPanel` are currently empty.

   - You can see that there is no triangle next to either panel.

8. Inside the `RightPanel` you find an object called `DestinationList`.

   - The Destination list manages which locations have been visited by the participant.
   - It also contains the logic that stops the experiment if the participant did not complete the experiment on time.

9. Set the maximal duration participants have to find all destinations.

   (a) In the Hierarchy view (2), click the `DestinationList`.

(b) In the Inspector view (3), scroll down to the `CrossOfElement` script.

- There is an item in the script called `MaxDuration`. This items is used to set a maximum amount of time before the learning phase will be aborted, and the next scene is loaded (no matter the progress of the participant). The unit of `MaxDuration` is in minutes.

(c) Set `MaxDuration` to 10.

10. Reveal the children of the `CentralPanel`

    - There is an object called `Text and Image`, which will be used to display instructions in the center of the screen.

11. Reveal the children of the "Text and Image" object and then click on the `PopUpText` object.

12. In the Inspector view (3), scroll down to the script called `Timed Pop up Text`.

    - This Script is responsible for making an instruction text disappear after a certain time. You can set this delay in the field `Display reset`. By default it is set to 4 seconds but it can be changed to any positive value.

13. Set the `Timed Pop up Text` to 8.

14. In the Hierarchy view, hide the children of `HUD_Canvas_4_panels` by clicking again on the little triangle to its left.

5.4.2 Experiment destinations

Now that we have the UI display in our project, it is time to add the destinations markers inside the virtual environment (the Viking Village). Destination marker visually highlight the region that participants need to reach in a navigation task. In contrast to the classical UI, destination markers are diagetic (Fagerholt & Lorentzon, 2009). They are located in the same space as the virtual environment and can be understood as a filter to highlight possible interactions for the participant (Fagerholt & Lorentzon, 2009). Additionally, we will link the destinations to the user interface to enable interactions with the user.

1. Create a new `GameObject` inside the the scene.

    - This will contain all the destination markers that we will add. Although this step is not strictly necessary, but it helps you to keep things organized.

(a) Right click on an empty space in the Hierarchy view (2).

(b) select “Create empty”.

- You can see that there is now a new game object called `GameObject` in the hierarchy.

2. Change the name to `Destinations`.

(a) Click on the new game object.

(b) In the Inspector view (3), change its name by clicking on the name.

(c) Type the name “Destinations”.

(d) Make sure to apply the changes by pressing the enter key.

3. Link the `Destinatons` game object to the user interface, as shown in Fig. 21.

- There is a script in the user interface that performs all the actions on the destination markers. But the scripts needs to know where the markers are.

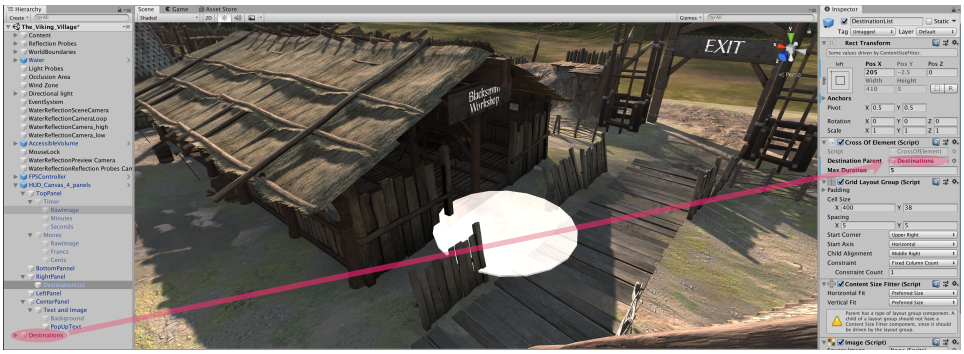


Figure 21: Drag and drop in Unity to move a hierarchy element into a script.

(a) In the Hierarchy view (2), click the `DestinationList`.

i. If you can’t see the `DestinationList`, you need reveal it inside its parents. To do this, press the little triangle next to the `HUD_Canvas_4_panels`.

ii. Reveal contents of `RightPanel`.

(b) Assign `Destinations` folder to `CrossOfElement` script

i. In the Inspector view (3), scroll down to the `CrossOfElement` script.

ii. Drag and drop the `Destinations` folder from the Hierarchy view (2) to the empty slot called `Destination Parent` in the `CrossOfElement` script in the Inspector view (3).

4. Add a single destination marker into the destinations list.

   (a) Go to `Assets\ EVE\Resources\ Prefabs\ Interactables\`.

   (b) Drag and drop the `DestinationMarker` object from the Project view (4) in the newly created "Destinations" object in the Hierarchy view (2).

   - You should now have the `DestinationMarker` as child of the `Destinations` object.

5. Rename the `DestinationMarker` to reflect the destination's name.

   (a) Click on the name in the Inspector View (3).

   (b) Type the name "Blacksmith Workshop".

   (c) Make sure to apply the changes by pressing the enter key.

6. Place the destination in the correct location.

   - As mentioned before, you can either move the object manually or you can set the coordinates in the inspector.
   - Note that the coordinates of children are always relative to the parent's object, therefore we also need to set the coordinates of the parent correctly.

   (a) The coordinates of the "Destinations" object (parent) are:

Position		
X	Y	Z
0	0	0

   (b) The coordinates of the "Blacksmith Workshop" object (child) are:

Position		
X	Y	Z
−39	6	−23

You can test if you have correctly placed the first destination marker (and referenced it in all the necessary places) by pressing the Play button (6) and walking to the destination. Once you are done testing, press the escape button to reveal the mouse, and then press the Play button (6) again to quit the Play mode.

We prepared 8 different locations for the complete experiment[9]. You can also find them in Figure 17. To add the additional destinations, please

[9]It is up to you whether you want to add all of the locations for this tutorial. However, if you want properly use the JRDs, you will have to add at least 3 destinations

repeat the steps outlines above. Table 1 presents the coordinates of the eight destination in the Viking Village.

	Position		
Name	X	Y	Z
Blacksmith Workshop	−39	6	−23
Gift Shop	28.5	6.5	−14.5
Smokehouse	−38	5.5	22
Cooking Shed	40.5	2.5	48.5
Town Hall	12.5	5.5	−5.5
Viking Boat	−67	4	31
Wood Storage	−67	5.5	0
Bakery	32.5	4.5	24

Table 1: The locations of all destinations

5.4.3 EVE Management of First Person Controller

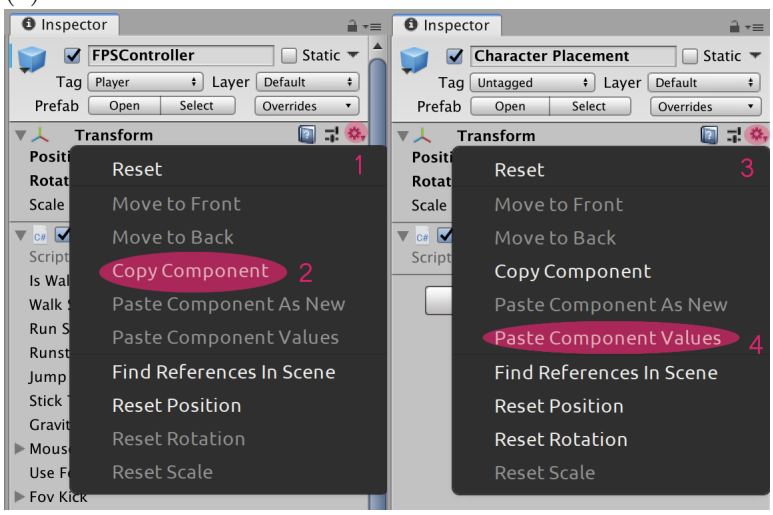
EVE uses a special persistent `FPSController` to keep track of the the actions of participants in the virtual environment. Instead of manually placing the controller in each environment, we denote locations of interest with markers that EVE will use to place the controller. EVE comes with three such placement markers:

- The `CharacterPlacement` object works as a beacon for EVE to know where to place the controller when a scene is loaded.
- The `RandomCharacterPlacement` object allows you to place multiple markers inside a scene. A random one will be used as the start location. The random choice is logged into the database.
  - *Note:* This script can be used as a starting point to create your own placement script where you can select between different starting locations based on an experiment parameter. Experiment parameters will be explained in a later section.
- The `NextLevelArea` triggers EVE to close the scene and move on to the next scene declared in the scene order.

To test your scene so far, you used the `FPSController` above (see section 5.3). However, EVE will later use the placement markers to place

the `FPSController` in the scene. Furthermore, this placement marker for the participants' controller only works, if a scene is started through the EVE Launcher. Once testing is complete, you need to remove your test `FPSController`. Leaving it in the scene would lead to two controllers in the scene when EVE adds the persistent controller. We will now setup our Viking Village so that EVE knows where to place the persistent `FPSController`:

1. Place the `CharacterPlacement` at the location of the `FPSController`.
   (a) In the Project view (4), go to `Assets\ EVE\ Resources\ Prefabs\ Player`.
   (b) Drag and drop the `CharacterPlacement` into the Hierarchy view (2).



   (c) In the Hierarchy view (2), click on the `FPSController`.
   (d) Copy its position and rotation to the new `CharacterPlacement`.
      - In the Inspector view (3), click on the little cog wheel next to the position information of the `FPSController` (labeled 1).
      - Click on "Copy component" (labeled 2).
      - In the Hierarchy view (2), click on the `CharacterPlacement`.
      - Again in the Inspector view (3), click on the little cog wheel next to the position information of the `CharacterPlacement` (labeled 3).
      - Click on "Paste Component Values" (labeled 4).

2. Disable `FPSController` used for testing.
   (a) Click on the object in the Hierarchy view (2)
   (b) In the Inspector view (3), uncheck the box on the top left corner next to the name.

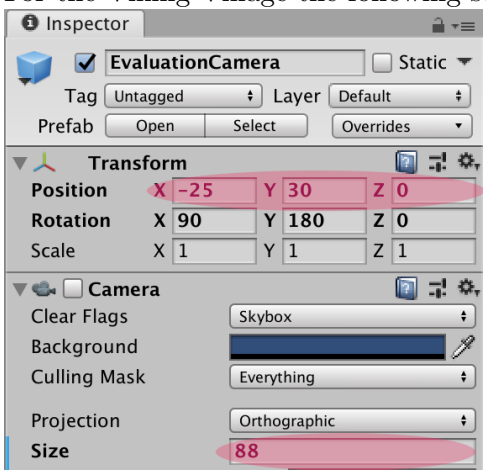
5.4.4 Create Evaluation Map

EVE allows you to create quick top-down maps that trace the path of participants as they navigate through the virtual environment. In order to create the evaluation map, you will need to place an "evaluation camera" in the scene that tells EVE the perspective from which to draw the map for you.

1. In the Project view (4), go to `Assets\ EVE\ Resources\ Prefabs`
2. Drag and drop the `EvaluationCamera` from the Project view (5) to the Hierarchy view (2).
3. Select the `EvaluationCamera` in the Hierarchy view (2).
   - You will see a small camera preview in the scene view.
4. Place the camera such that the every navigable part of the virtual environment is visible.
   - Option 1: Move the camera using the move tool of the scene view
   - Option 2: In the Inspector view (3), change the position fields of the `Transform` component.

Position		
X	Y	Z
−25	30	0

   - To zoom in or out, you have to use the `Size` field of the `Camera` component attached to the `EvaluationCamera`. We recommend a size of 88.
   - For the Viking Village the following settings work best.



5. Generate map and transformation information [10].

[10] Note that on Linux you need to keep the FPSController active or Unity may crash.

(a) Press the Play button (6).

(b) Press Escape to see the mouse again.

(c) Press again the Play button (6) to stop.

(d) Check the Console view (7) for an entry stating that the map has been created.

- In the console the location of the image is stated.
- You can ignore the errors shown at this time.
- The map for this scene has been created in the background.
- EVE now can generate paths on this map when you run a participant.
- The evaluation camera is no longer necessary.

(e) Deactivate the evaluation map generation.

- We are just deactivating the evaluation map generation because for the replay mode we want to use the evaluation camera for a bird's eye view.

i. In the Hierarchy view (2), click on `EvaluationCamera`.

ii. In the Inspector view (3), locate the `EvaluationMapGenerator` Script at the bottom.

iii. Uncheck the box on the top left corner next to the name `EvaluationMapGenerator`.

5.5 Setting up the testing phase

Setting up the testing phase follows some of the same the steps outlined above for preparing the learning phase. In order to to reduce the workload, we will start by duplicating the learning phase and adapting it to the requirements of the testing phase. Then, we will add a timer to testing scene and change the logic for testing.

5.5.1 Duplicate the scene

1. Open the `The_Viking_Village` scene (in case you are not there already).

   (a) Go to `Assets\ Experiment\ Scenes\` in the Project view (4).

   (b) Double click on the `The_Viking_Village`.

2. In the menu bar at the top of the Unity editor, go to "File" >"Save Scene as..."

3. Save the scene as `The_Viking_Village_Testing` in the folder `Assets\ Experiment\ Scenes`.

5.5.2 Adjust UI for testing phase

The UI for the testing phase differs slightly from the one used in learning phase described in the experiment design (section 5.1).

1. Reactivate the `Timer`.
   (a) In the Hierarchy view (2), go to `HUD_Canvas_4_panels\ TopPanel\ Timer`.
   (b) In the Inspector view (3), check the box on the top left corner of the `Transform` component of the `Timer`.

   - The time limit will be done through a *experiment parameter* which will be setup at a later stage.
   - Note: As long as you test the scene directly (without loading it through the main menu of EVE which will be covered later), the timer will not be functional since the connection to the database is not setup.

2. Deactivate the `DestinationList` in the UI.
   (a) In the Hierarchy view (2), go to `HUD_Canvas_4_panels\ RightPanel\ DestinationList`.
   (b) In the Inspector view (3), untick the box on the top left corner of the `Transform` component of the `DestinationList`.

3. Add the logic of testing to `PopUpText`.
   (a) In the Hierarchy view (2), go to `HUD_ Canvas_4_panels\ CenterPanel\ Text and Image\ PopUpText`
   (b) Press the "Add Component" at the bottom of the Inspector view (3).
      - Depending on you screen size you may have to scroll down.
   (c) Type in "GoToDestinations" into the search box.
   (d) Select the `GoToDestinations` script from the drop-down list.

4. In the `GoToDestinations` component, set the parameters for the testing phase.
   (a) Drag and drop the `Destinations` game object from the Hierarchy view (2) to the empty slot called `Destination Parent` in the `GoToDestinations` component in the Inspector view (3).
      - All children of the `Destination Parent` will have be visited by the participants in the order that they appear in the Hierarchy view (2).

(b) Set `Info Delay` to 4.

- This script first shows a message that the participants reached a goal, and then tells the participant about their next goal (destination).
- This delay defines how much time should pass before the message switches from "reached" to "go to".
- Note: The time here should always be smaller than the display reset time set in the `Timed Pop Up Text` set in the `PopUpText` component.

5.5.3 Change the behavior of destination markers

Since we only want the current goal to be visible, we have to deactivate the visualization of the destinations markers. This step is necessary given that destination marker (represented by `Cylinder` objects in the scene) will be activated again to show the destination that needs to be visited.

1. Deactivate the `Cylinder` markers of the destinations.

   (a) Click the search bar at the top of the Hierarchy view (2).

   (b) Type in "Cylinder".

   - This will show all the objects in the scene called "Cylinder".

   (c) Mark all Cylinder objects in the Hierarchy view (2).

   - Since only the destination marker objects are called cylinder, we can edit all of them at once.

   i. Click on the `Cylinder` at the top of the Hierarchy view (2).

   ii. Hold down the shift button and click on the `Cylinder` at the bottom of the Hierarchy view (2).

   - All `Cylinder` objects should be selected.

   (d) Deactivate all `Cylinder` objects at once.

   - In the Inspector view (3), untick the box on the top left corner of the `Transform` component of the `Cylinder`.

   (e) Reset the search in the Hierarchy view (2).

   i. Press the little x on the right side of the search bar to return to the normal Hierarchy view (2).

5.5.4 Testing the "testing phase" and creating evaluation map

Now we can test our new scene and its experiment logic and then create an evaluation map.

1. Enable `FPSController`.

(a) In the Hierarchy view (2), select the `FPSController`.

(b) In the Inspector view (3), check the box on the top left corner of the `Transform` component of the `FPSController`.

2. Go through all tasks for testing.

(a) Press the Play button (6).

(b) Follow the instructions and visit all destinations.

- At the end, the screen fades to black.

(c) Press again the Play button (6) to end.

3. Disable `FPSController`.

- Don't forget to disable the FPSController like in Section 5.4.3 to allow EVE to manage the participants' controller.

(a) In the Hierarchy view (2), select the `FPSController`.

(b) In the Inspector view (3), uncheck the box on the top left corner of the `Transform` component of the `FPSController`.

4. Make an evaluation map[11].

- Note that to see the evaluation map you need to repeat the Section 5.4.4 for the testing phase.

(a) Enable the evaluation map generation.

i. In the Hierarchy view (2), click on `EvaluationCamera`.

ii. In the Inspector view (3), locate the `EvaluationMapGenerator` Script at the bottom.

iii. Check the box on the top left corner next to the name `EvaluationMapGenerator`.

(b) Press the Play button (6).

(c) Disable the evaluation map generation.

i. In the Hierarchy view (2), click on `EvaluationCamera`.

ii. In the Inspector view (3), locate the `EvaluationMapGenerator` Script at the bottom.

iii. Uncheck the box on the top left corner next to the name `EvaluationMapGenerator`.

[11] Note that on Linux you need to keep the FPSController active or Unity may crash.

5.6 Setting up a judgment of relative direction task (JRD)

As mentioned in experiment design section 5.1, the experiment will include a judgment of relative direction task (JRD). EVE offers two distinct possibilities to perform a JRD with different trade-offs. In the first subsection, we will setup a JRD that is a pointing task in an empty virtual environment. In the second subsection, we will setup a "drag and drop" style JRD.

5.6.1 JRD1: Pointing task

EVE contains a scene to setup JRD in 3D. The purpose of the JRD is to rotate (using the mouse) to the correct angle relative to the destination. A compass rose is often used as a backdrop for a JRD to give participants some orientation cue.

1. In the Project view (4), go to `EVE\ Scenes\ Tasks\`.
2. Open the scene `JRD_3d`.
    - The scene is already set up with some default values (3 locations unrelated to our Viking Village), which allows you to test instantly.
3. Test the pointing task JRD.
    (a) Press the Play button (6).
        - The scene consists of a generic environment where the camera is situated in the middle of a compass rose.
        - You can see a location name positioned at the front of the compass rose and the word "Back" at 180 degrees away from the location name.
        - You will also see an instruction text floating at the center of screen.
    (b) Use your mouse to rotate the camera left or right.
    (c) To lock in an angle, use the "Enter" key on your keyboard.
        - Once you lock an angle you are not able to use the mouse to change it.
        i. Option 1: Press the "Enter" key again to submit your selection and continue.
        ii. Option 2: Press the "Escape" (Esc) key to change you selected angle.
    (d) Complete all example trials.
        - After finishing all trials, the screen will fade to black and the main menu will be loaded (only when testing it directly by opening the scene from the Project view (4)).

(e) Press the Play button (6) to stop and to return to the Scene view (1).

4. Locate the `CheckPointAngle` component.

   - Option 1: Navigate through the Hierarchy (2) view to find it attached to the `First Person Controller`.
   - Option 2: Use the search bar at the top of the Hierarchy view (2) to search for it. Note: the result will only be shown once you type in the full name of the script.

5. Setup JRD pointing destinations in `CheckPointAngle`.

   (a) Setup `names` of locations.

   - This list contains the names of locations that will be shown in the instruction text, and the text positioned at the front of the compass rose, see Figure 18.

   i. Set the size of the list at the top to 8 for all locations.

Names	
Element	Name
0	Blacksmith Workshop
1	Gift Shop
2	Smokehouse
3	Cooking Shed
4	Town Hall
5	Viking Boat
6	Wood Storage
7	Bakery

   ii. Write the names of all locations into the the fields that appeared below.

   (b) Setup the `order` in which pointing tasks are shown.

   - This list contains the setup for each trial that you want to show.
   - You can add as many trials as you want to the order list.
   - A trial description consists of three digits.
     - The first digit is the location the participant is standing at.
     - The second digit is the location the participant is facing.
     - the third digit is the location the participant is asked to point to.
   - Each name entry that you set in the `names` list can be used by looking up the number which is written after "Element".

- In the default setting you can see in `names`:
  - "Element 0" is "Home".
  - "Element 1" is "Church".
  - "Element 2" is "Park".
- If we want to have a trial in which participants are told that they are at the Home (0) facing the Church (1), and then asked to point to the park (2), we would add "012" as an element to the `order` list.

i. Setup the trials for the JRD

- In the Inspector view (3), set the size of the `Order` to 12:

Trials	
Element	Trial
0	012
1	345
2	670
3	123
4	456
5	701
6	234
7	567
8	024
9	135
10	246
11	357

(c) Activate randomisation via `Randomize Order`.

- This check box can be used to specify the order as either in the list or randomised for each participant if ticked.

(d) Setup the maximum number of repetitions `MaxRepetitions` if you don't want to use all trials for each participant. -1 Indicates that each participant has to complete all trials.

i. Set `MaxRepetitions` to 5.

6. Test the JRD pointing task again with your setup.

You have setup a 3D JRD to analyse your participants' mental representation of landmark locations in your environment.

5.6.2 JRD2: "Drag and drop" style

EVE contains a scene to setup a "drag and drop" JRD.

1. In the project view (4), go to `EVE\ Scenes\ Tasks\`.

2. Open the scene `JRD_Drag_and_drop`.
    - This JRD presents the image of two buildings marked with "Building A" and "Building B". Participants are told that they are standing at a specific location, facing a building and asked to point to another building. For example, "you are standing in front of building A, facing building B, point to building C"
    - The participant then has to drag the image of building and drop it into one of seven boxes around the circle as shown in Figure 19.
    - This JRD has to be setup before using it (in contrast to the 3D JRD which can be tested right away).
3. Add images for Landmark A, Landmark B, and Landmark C to the JRD.
    (a) Use the search bar of the Hierarchy view to search for the `NextButton`.
    (b) In the Inspector view (3), find the `CheckJRD` component on the `NextButton`.
    (c) Add images to the `Landmarks` lists in `CheckJRD`.
        - Each trial will take three images from the list, and place Image A and B to the left and Image C at the drag and drop start point.
        - We provide you with eight images of all task locations from the viking village to setup this JRD.

        i. In the Inspector view (3), set the size of the `Landmarks` to 8.
            - If you can't see the size property, click the little triangle next to `Landmarks` to expand the view.

        ii. In the Project view (4), open the image location `Assets\ Experiment\ Resources\ Images\`.

        iii. In the Inspector view (3), have the lists for images visible.

        iv. Drag and drop images from the Project view to slots in the list in the following order:

Names	
Element	Name
0	Blacksmith Workshop
1	Gift Shop
2	Smokehouse
3	Cooking Shed
4	Town Hall
5	Viking Boat
6	Wood Storage
7	Bakery

- Make sure that you don't put the same image at two location (otherwise two images shown in one task will be the same).

(d) Add an order like in the previous JRD:

Trials	
Element	Trial
0	012
1	345
2	670
3	123
4	456
5	701
6	234
7	567
8	024
9	135
10	246
11	357

(e) Activate randomisation via `Randomize Order`.

- This check box can be used to specify the order as either in the list or randomised for each participant if ticked.

(f) Setup the maximum number of repetitions `MaxRepetitions` if you don't want to use all trials for each participant. -1 Indicates that each participant has to complete all trials.

i. Set `MaxRepetitions` to 5.

4. Test the JRD.

(a) Press the Play button (6) to start the JRD scene.

(b) The image marked as "Building C" can be dragged into any of the seven small boxes placed around the compass rose at the center of the screen.

(c) If you want to change your selection, pull the image marked as "Building C" onto another box.

- You must always start from the big image box on the right side, not the small image boxes around the compass rose.
- Your previous selection will automatically disappear.

5. Add your own images (optional).

(a) In the Project view (4), go to `Assets\ Experiment\ Images`.

(b) Find the image location in the Windows Explorer.

i. Right click on any of the images.
ii. Select “Show in explorer”.
iii. Wait for the Windows Explorer to open.

(c) Copy any other image files into the opened folder.

(d) Back in Unity, wait for the new assets to be recognized.

(e) Change the type of all images to `sprite`.

- The unity images used to display the image are of the type sprite.
- Since all new images are of the same kind, we can edit them in bulk

i. In the Project view (4), hold down the shift button and click each image.
ii. In the Inspector view (3), change the “Texture Type” entry of the import settings to “Sprite (2d and UI)”.
iii. Press the apply button on the bottom right of the Inspector view (3).

5.7 Control interface training: Maze training

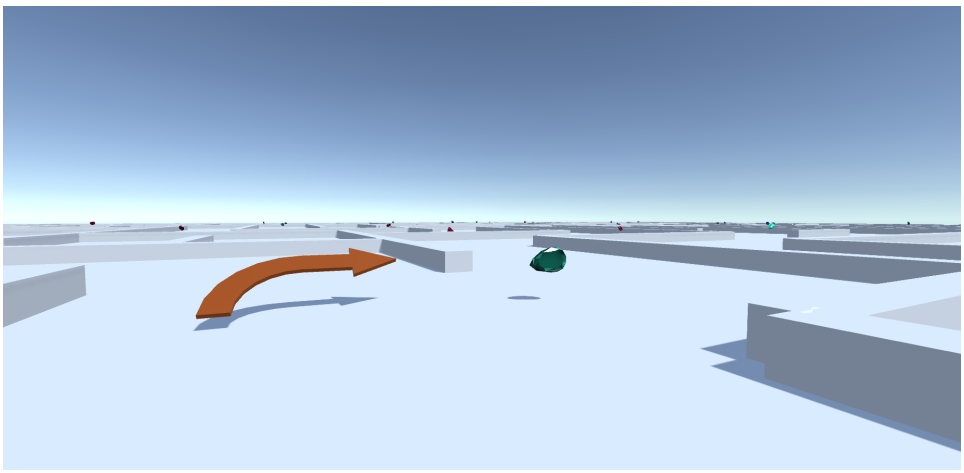


Figure 22: In this training maze, participants are made familiar with the controls to ensure that we measure their performance with regard to the experiment’s task rather than their ability to use the controller (Grübel, Thrash, Hölscher & Schinazi, 2017).

In the maze training scene, participants are placed at the start of a maze and asked to collect a variety of floating gems, see Fig. 22. Participants

are guided by a series of pop-up arrows that are activated as participants approach them. Experimenters have the choice of selecting how many gems need to be collected and the end location of the maze. Anecdotal research from our lab has shown that participants tend to become familiar with the control interface after 5 to 7 minutes of maze navigation.

1. In the Project view (4), go to `Assets\ EVE\ Scenes\ Training\`.
2. Open the `Maze` scene.
   - We recommend to leave the starting point where it is and only move the end point.
   - The starting point is marked with the object called `Character Placement`.
     - This is where our persistent first person controller will be placed (as discussed in Section 5.4.3).
   - If you want to test the maze from within the scene (without going through the experiment setup menu), you need to place the `FPSController` as discussed in section 5.3.
     - Make sure to remove the `FPSController` once your done testing. Leaving the `FPSController` in the scene will lead to conflicts.
3. Set the end point of the maze training according to your preference (Optional).
   (a) In the Hierarchy view (2), find the `nextLevelArea` object.
   - This object marks the end of the maze.

   (b) Move it using the move controls of the Scene view (1).
   - Make sure to move it along to path marked by the arrows and gems in the maze, otherwise you might set the endpoint at an unreachable position.
4. Make a map for participant evaluation like in Section 5.4.4.
   - The settings are already correct on the camera.

   (a) Press the Play button (6).
   (b) Press Escape to see the mouse again.
   (c) Press again the Play button (6) to stop.
   (d) Check the Console view (7) for an entry stating that the map has been created.

5.8 Info screens

Information screens help guide participants along the different phases of the experiment by presenting instructions about the task that will take place or the stimuli that will be presented. Information screens can prevent users from switching attention between tasks or moving around the lab during the experiment. We will include different information screens throughout the tutorial experiment. An information screen can be composed of a variety of sub-screens to accommodate for different amount of information while presenting it in a coherent fashion.

The introduction information screen presents participants with basic information about the experiment. The type of information will vary depending on the experiment but this screen is typically used to inform the participant about the different phases and the number of trials in the experiment. Detailed information about each phase can be presented with other information screens presented at different stages of the experiment.

For this first tutorial, we will use the introduction information screen to inform participants about the type of questionnaires they will be asked to complete (i.e., demographics and SBSOD), the control interface training scenes (i.e., maze training) and the different phases of the experiment (i.e., learning and testing). This screen will also provide some basic information regarding the JRD tests that will follow the navigation testing phase and the number of trials that participants will be asked to complete.

The first subsection will tell you how to setup an information screen, and the following subsections will tell you which information screens you should setup for this experiment. We leave out information screens for the introduction and the end screen to give you an opportunity to write them yourself.

5.8.1 Locate the templates

Information screens in EVE are provided as a type of question without answer[12]. To setup such a screen all we need to do is write our own `Questionnaire` and `QuestionSet`. So far, we only prepared questionnaires and did not have to look at them ourselves.

Questions are stored in EVE as an XML (eXtensible Markup Language) file. This allows EVE to accommodate a great variety of question types that that can be easiy read by human and still be understood by the computer as well. If you want to learn more about XML, please consider following this tutorial.

[12]This is based on the `TestSet.xml` located at `Assets\ Experiment\ Resources\ QuestionSets\` and `ExampleQuestionnaire.xml` located at `Assets\ Experiment\ Resources\ Questionnaires\`. `TestSet.xml` contains all currently supported question types of EVE. If you want to create your own questionnaires this is a great place to get started.

You can find the necessary files in `Assets\ Experiment\ Resources\ QuestionSets\` and `Assets\ Experiment\ Resources\ Questionnaires\`.

5.8.2 Understand question sets

We will start by understanding a simple question set that contains an information screen. In Figure 23, we can see the `InfoScreen` node. This is where we describe to EVE how to format the information screen. If we would add more than one question to the question set, we could add more nodes inside `<Questions >` below the `InfoScreen` node. The Name attribute of the `QuestionSet` node needs to be the same as the file name. It will be referenced later in a questionnaire, so be careful about typos.

```
<?xml version="1.0" encoding="utf-8"?>
<QuestionSet xmlns:xsd="http://www.w3.org/2001/
    XMLSchema" xmlns:xsi="http://www.w3.org/2001/
    XMLSchema-instance" Name="
    InformationQuestionSetExample">
  <Questions>
    <InfoScreen Name="example_confirm" NColumns="1"
        NRows="1">
      <Text><![CDATA[<b><size=48>Title</size></b>

This is where the instruction is written.]]></Text>
      <ConfirmationRequirement Required="true"
          ConfirmationDelay="5" />
    </InfoScreen>
  </Questions>
</QuestionSet>
```

Figure 23: The XML file for a `QuestionSet` consisting of a single `InfoScreen`.

Every question needs a unique name. Therefore we should always choose the `Name` of any question node as something meaningful. The unique name should always reflect the task of the question or the kind of information that is being conveyed. The name will be used in **R** to find the participants' answer to the question.

The important part is the `Text` element where we write our information that we want to display to the participant. It is very important that you always keep the `<![CDATA[` at the beginning of the `Text` element and the `]]>` at the end. It is called a data-section in XML and it allows us to use the html formatting that Unity offers us to resize and color the text. It is also very important to know that new lines in the data-section are part of

your content. That is why the empty line in the data-section looks strange and "This" starts without indentation.

Next, we have a look at `ConfirmationRequirement`. If `Required` is set to "true" the participants need to press next again to confirm that they are ready. The alternative is to set `Required` to "false" in which case the next button functions normally. The `ConfirmationDelay` is the time in seconds that the next button cannot be pressed. This stops participants from quickly clicking through instructions. The delay should be set such that the next button becomes clickable once the instruction is read at a moderate pace.

In future sections, you will be shown how to add the title and text to the information screens yourself.

5.8.3 Understand questionnaires

To embed the question set into EVE, it needs to be listed inside a questionnaire. Since our information screens always appear alone before each scene, the questionnaire is very simple, as show in Figure 24.

```
<?xml version="1.0" encoding="utf-8"?>
<Questionnaire xmlns:xsd="http://www.w3.org/2001/
    XMLSchema" xmlns:xsi="http://www.w3.org/2001/
    XMLSchema-instance" Name="
    InformationExampleQuestionnaire">
  <QuestionSets>
    <QuestionSet>InformationQuestionSetExample</
        QuestionSet>
  </QuestionSets>
</Questionnaire>
```

Figure 24: The XML file for a `Questionnaire` consisting of a single `QuestionSet`.

All the questionnaires are already prepared. However, you can have a look over them and see that the `QuestionSet` inside each `Questionnaire` has exactly the same name as the question sets you will be preparing.

Now, we have prepared all components of the experiment and it is time to gather everything.

6 Combining all stages in EVE

Now that we have setup all the necessary stages for the experiment, we can put them together using the EVE launcher. There are three steps required to finalize the experiment:

1. We need to add all the scenes we prepared to the Unity build settings.
2. We need to add and order all the scenes in the EVE menu.
3. We need to add the necessary experiment parameters.

6.1 Adding scenes to the Unity build settings

In Unity, every scene in the experiment has to be added to the build settings in order to be loaded.

1. At the top of the Unity window, click "File" in the menu bar.
2. In the "File" menu, click on "Build Settings..."
   - A new window called "Build Settings" will open.
3. Add all scenes to the build:
   (a) Add EVE scenes:
      i. In the Project view (3), navigate to the EVE scene folder `Assets\ EVE\ Scenes\`
      ii. In the Project view (4), drag and drop the three scenes `Training\ Maze .unity`, `Tasks\ JRD_ 3d .unity`, and `Tasks\ JRD_ Drag_ and_ drop .unity` into the "Build Settings" list called "Scenes In Build".
   (b) Add experiment-specific scenes.
      i. In the Project view (3), navigate to the experiment scene folder `Assets\ Experiment\ Scenes\`
      ii. In the Project view (4), drag and drop the two scenes `The_ Viking_ Village .unity` and `The_ Viking_ Village_ Testing .unity` into the "Build Settings" list called "Scenes In Build".
   - The final list should look like Fig. 25.

6.2 Preparing scene order

Similarly to the setup of the basic experiment we have to add the scenes in the EVE configuration menu. Start the launcher in Unity (by pressing the Play button (6)), then go to "Configuration" >"Scene Setup" and add the scenes in the following order (you can use the arrows next to the added scenes to move them up in the list):

1. Info screen introduction (optional)
2. Demographics questionnaire & SBSOSD (`sbsod_lite.xml`)

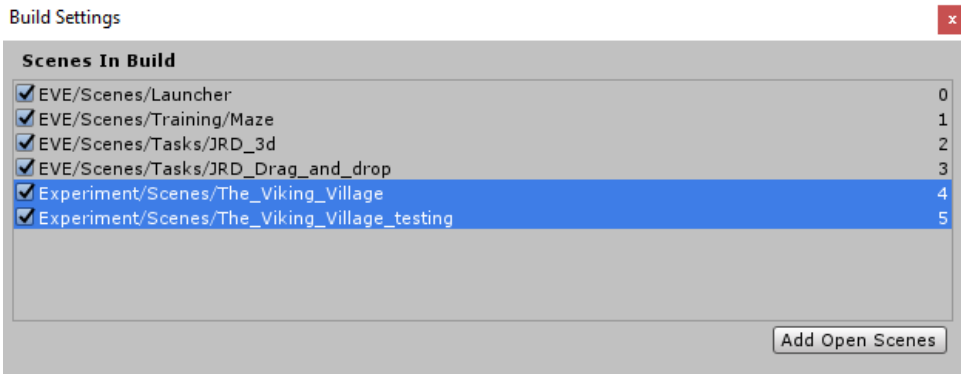


Figure 25: The build settings when you added all scenes should look like this.

3. Info screen: Movement training (`info_movement.xml`)
4. Maze (`Maze.unity`)
5. Info screen: Learning phase (`info_learning.xml`)
6. Learning phase (`The_Viking_Village.unity`)
7. Info screen: Testing phase (`info_testing.xml`)
8. Testing phase (`The_Viking_Village_Testing.unity`)
9. Info screen: JRD (pointing) (`info_jrd_pointing.xml`)
10. JRD (3D) (`JRD_3D.unity`)
11. Info screen: JRD (questionnaire) (`info_jrd_questionnaire.xml`)
12. JRD (Drag and Drop) (JRD_Drag_and_drop.unity)
13. Info screen: End (optional)

Once you have added and ordered all scenes, it should look like Figure 26. You can now go back to the config menu and save the settings like shown in Section 4.2. Remember that without saving, your scene order will not be stored when you start EVE the next time.

6.3 Experiment parameters

The experiment we prepared requires us to set two parameter called "maxDuration" and "timePressure". The former will be used to define to maximal number of seconds that the participant has spent in the testing phase (before it switches to the next stage of the experiment) and the later will define whether we apply the time pressure defined above by saying "yes" for time pressure and "no" for no time pressure.

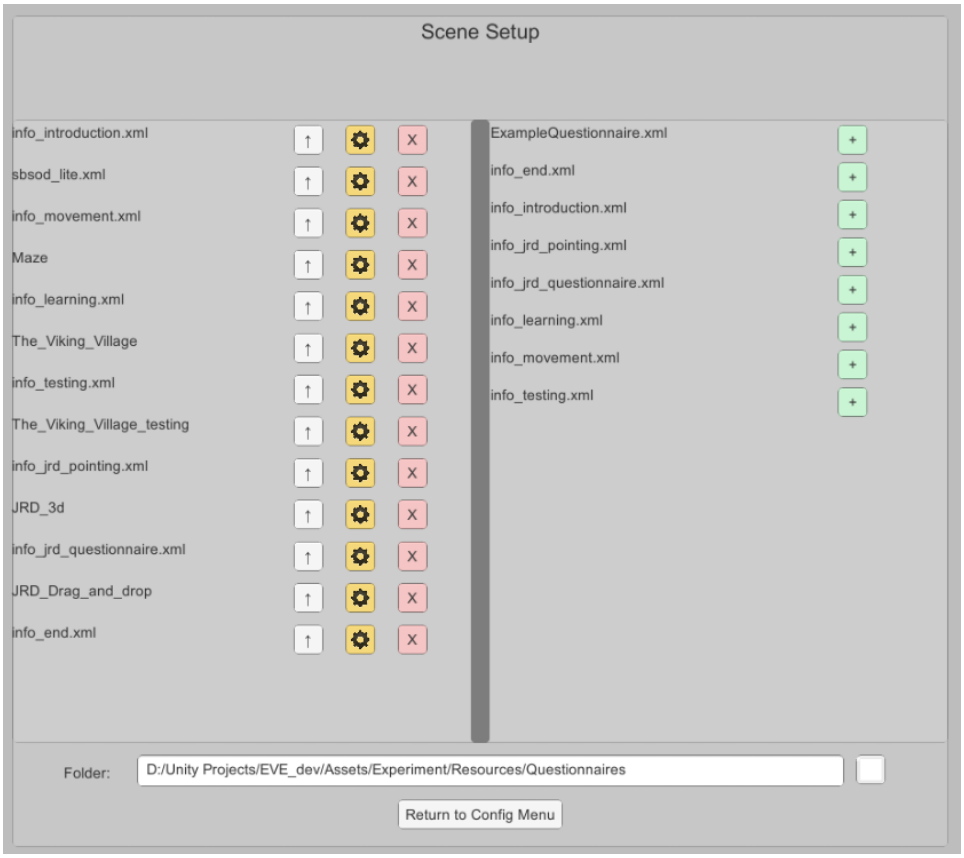


Figure 26: The scene order should look like this when you added all scenes in the proper order.

1. In the Project view (4), go to `Assets\ EVE\ Scenes` and then double click on Launcher.
2. Press the Play button (6).
3. Click on the "Configuration".
4. Click on the "Experiment Parameters" button.
5. Insert the experiment parameter `maxDuration`.
   - Press the green "+" button to add a new experiment parameter.
   - Enter the name of the new experiment parameter called `maxDuration`.
   - Press the "OK" button.
     - You should now see the parameter in the list similar to Fig. 27.
     - If you want to delete the parameter, press the red "x" button next to the parameter in the list.

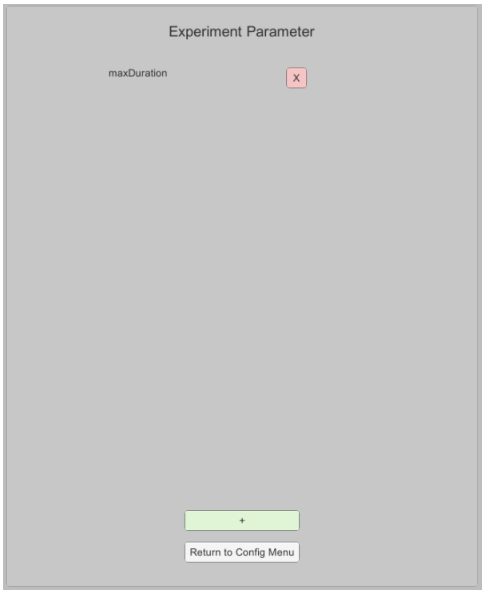


Figure 27: The experiment parameter menu with the parameter `maxDuration` inserted.

6. Insert the experiment parameter `timePressure` in the same way as the previous experiment parameter.

7. Return to the configuration menu and save the settings one more time.

The experiment parameters now appear in the list of session parameters when you want to start an experiment. You are required to set a value to the session parameters before you can start any experiment.

6.4 Adding new questionnaires to the database

Before we can use our new questionnaires, we need to make the database aware of these.

1. In the Project view (4), go to `Assets\ EVE\ Scenes` and then double click on Launcher.

2. Press the Play button (6).

3. Click on the "Database connection".

   - You see the database menu shown in Fig. 28a.
   - You can only "Reset tables" which includes deleting all current data. Be advised not to ever click this when you collected real data.

4. Click on “Reset tables”.
    - A pop-up window appears to ask you whether you really want to delete all data.
5. Confirm that you want to delete all data.
    - The database menu now looks like in Fig. 28b.
6. Press “Add questions” to add all questions again including your new info screens.
    - Note that to add questions the database, a reset is necessary because the new questions may conflict with the old when looking at the data without a reset.

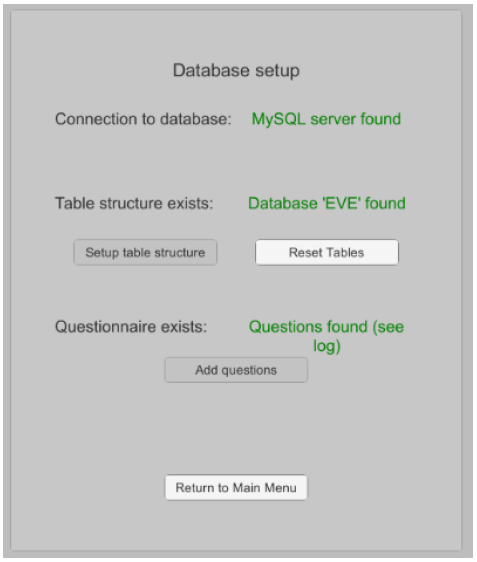


Figure 28a: The database connection menu allows you to reset all data tables and delete everything.

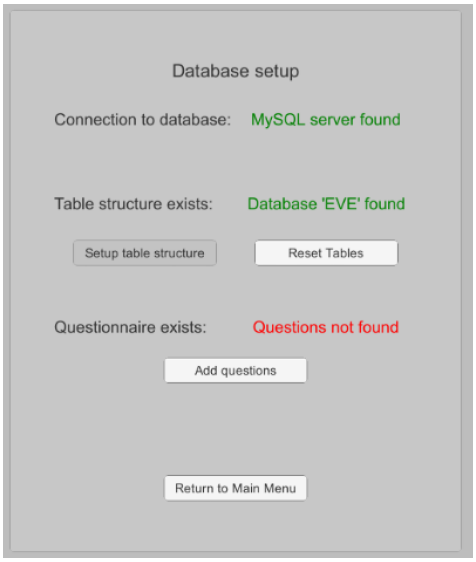


Figure 28b: If the database is empty, you are allowed to add questions again.

You have now successfully created your own information screens and stored them in questionnaires.

6.5 Test experiment

Now, the experiment is setup. Return to the main menu and then go to the experiment menu. In order to start the experiment, you have to first press the Parameters button and fill in the the “maxDuration” value (remember this value is defined in seconds) and the “timePressure” (remember this value

is defined as “yes” to add time pressure and “no” to leave time pressure out). Once you set the value, you can start the experiment as explained in the setup of the questionnare experiment in Section 4.2. In order to have an interesting evaluation, it would be good to perform the experiment at 3 and up to 10 times.

7 Evaluation

EVE offers you two ways to look at the created data: A quick overview inside the EVE launcher, and more in-depth analysis using **R**.

7.1 Evaluation in Unity

To get an overview of the participants' performance, you can look at some meta-data like time spend per scene and distance walked, look at the path participants took or replay the participants' actions in the virtual environment.

1. In the Project view (4), go to `Assets\ EVE\ Scenes` and then double click on Launcher.
2. Press the Play button (6).
3. Click the "Evaluation" button.
4. Click the "Inspect Participants" button.
    - You get an overview list of all participants.
    - You can see the session and the participant identifier.
    - The red "X" button allows you to delete a participant.
5. Click "Show details".
    - An overview of all scenes is shown as in Fig. 29a.
    - For each scene, you will get meta information: the scene identifier, scene name, the time the participant spend in there and if it was 3D the distance they walked.
    - The Replay button takes you into the 3D scene and allows you to look at the participants' performance during the scene. You can observe the replay both in the participants' view and from a bird's eye view.
    - The show map button generates an overview of the scene based on the `EvaluationCamera` in the scene and a red outline of the participants' paths .

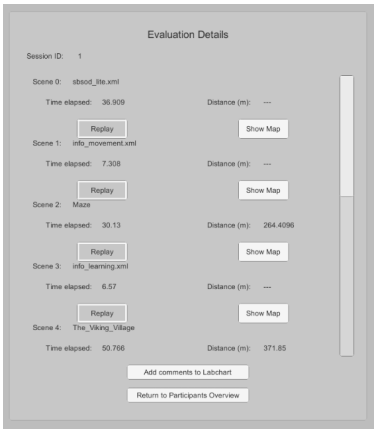


Figure 29a: The database connection menu allows you to reset all data tables and delete everything.

Figure 29b: If the database is empty, you are allowed to add questions again.

7.2 Evaluating with EVE R Tools

We offer you a variety of basic operations in **R** for collecting the questionnaire responses and looking at session parameters and participant movement data. This will allow you to prepare the data quickly for statistical analyses to follow. To get started, we provide you with an **R** notebook that covers the basics of evertools.

However, R still relies on an old type of connection to MySQL. Before starting with the evaluation, please follow these steps to generate a MySQL user in MySQL workbench that works in R:

1. Open Workbench.
2. In the administration view, click on "Users and Privileges".
3. At the bottom of the page, click "Add Account".
   (a) Give the account a "Login Name".
   (b) Set the password type to "Standard".
   (c) Add and repeat the password.
   (d) In the "Administrative Roles" tab, select DBA.
4. In the bottom right corner, press "Apply".

Use the newly created MySQL user and password in the R-File provided above[13].

[13] If you get an error that the authentication type is not allowed, please check Appendix B for possible solutions.

1. Unpack the file “evertools_example.zip” into a location of your choice.
2. Open Rstudio.
3. Open the file “evertools_example.Rmd”
4. Follow instructions in the file.

8 Exporting the program as a standalone

Creating a standalone program of a Unity program is straightforward. We therefore refer to the official Unity guidelines for publishing. The results of creating a build are a `.exe` file (depending on your windows settings you might not see the file endings) and a folder with the same name as the executable but with an extra `_data` at the end. For example, if you call your program `EVE.exe` then the folder will be called `EVE_data`.

8.1 Important: Adding missing library files

In order for EVE to run in the standalone version, you have to add two files whenever you create a build. Download the files from our website, extract the two “.dll” files and then copy them into the “Managed” folder that is located inside the “_data” folder.

This step is necessary as unity does not export some editor libraries (.dll-files) when creating a build. For your convenience we already found the files in the editor and provided them to you.

9 Conclusion

This is the first tutorial and gives an overview of the core features of EVE. We introduce how to setup the software and how to create a simple navigation experiment. With the experiment, we introduce how to move from an experiment design via an experiment protocol to an experiment implementation. Our experiment also explains how to create questionnaires, set markers in the world, store information in the database and retrieving the data from the database as well as publishing a build that can be used to run a study.

Future tutorials will focus on different aspects of EVE. Currently a tutorial on question creation is in the making. While not a tutorial per se, we published an article on using physiological sensors with EVE (Weibel et al., 2018) which you may also have a closer look at in case you are interested in collecting physiological data.

10 Reference Solution

We make the fully completed tutorial available as a download here.

References

Bryant, K. J. (1984). Methodological convergence as an issue within environmental cognition research. *Journal of Environmental Psychology*, *4*(1), 43–60.

Fagerholt, E. & Lorentzon, M. (2009). Beyond the hud-user interfaces for increased player immersion in fps games.

Grübel, J., Thrash, T., Hölscher, C. & Schinazi, V. R. (2017). Evaluation of a conceptual framework for predicting navigation performance in virtual reality. *PloS one*, *12*(9), e0184682.

Grübel, J., Weibel, R., Jiang, M. H., Hölscher, C., Hackman, D. A. & Schinazi, V. R. (2016). Eve: A framework for experiments in virtual environments. In *Spatial cognition x* (pp. 159–176). Springer.

Hackman, D. A., Robert, S. A., Grübel, J., Weibel, R. P., Anagnostou, E., Hölscher, C. & Schinazi, V. R. (2019). Neighborhood environments influence emotion and physiological reactivity. *Scientific reports*, *9*(1), 9498.

Hegarty, M., Richardson, A. E., Montello, D. R., Lovelace, K. & Subbiah, I. (2002). Development of a self-report measure of environmental spatial ability. *Intelligence*, *30*(5), 425–447.

Huffman, D. J. & Ekstrom, A. D. (2019). Which way is the bookstore? a closer look at the judgments of relative directions task. *Spatial Cognition & Computation*, *19*(2), 93–129.

Kozlowski, L. T. & Bryant, K. J. (1977). Sense of direction, spatial orientation, and cognitive maps. *Journal of Experimental Psychology: human perception and performance*, *3*(4), 590.

Likert, R. (1932). A technique for the measurement of attitudes. *Archives of psychology*.

Marchette, S. A., Vass, L. K., Ryan, J. & Epstein, R. A. (2014). Anchoring the neural compass: Coding of local spatial reference frames in human medial parietal lobe. *Nature neuroscience*, *17*(11), 1598.

Schinazi, V. R., Nardi, D., Newcombe, N. S., Shipley, T. F. & Epstein, R. A. (2013). Hippocampal size predicts rapid learning of a cognitive map in humans. *Hippocampus*, *23*(6), 515–528.

Shelton, A. L. & McNamara, T. P. (2001). Visual memories from nonvisual experiences. *Psychological Science*, *12*(4), 343–347.

Vass, L. K. & Epstein, R. A. (2017). Common neural representations for visually guided reorientation and spatial imagery. *Cerebral cortex*, *27*(2), 1457–1471.

Waller, D. & Hodgson, E. (2006). Transient and enduring spatial representations under disorientation and self-rotation. *Journal of Experimental Psychology: Learning, Memory, and Cognition*, *32*(4), 867.

Weibel, R. P., Grübel, J., Zhao, H., Thrash, T., Meloni, D., Hölscher, C. & Schinazi, V. R. (2018). Virtual reality experiments with physiological measures. *JoVE (Journal of Visualized Experiments)*, (138), e58318.

Weisberg, S. M., Schinazi, V. R., Newcombe, N. S., Shipley, T. F. & Epstein, R. A. (2014). Variations in cognitive maps: Understanding individual differences in navigation. *Journal of Experimental Psychology: Learning, Memory, and Cognition*, *40*(3), 669.

A Previously known issues with MySQL

In this section we list some known issues with the installation and offer possible solutions. If can you find other issues with MySQL (with respect to EVE), please inform us about the details and we will add it to this list.

A.1 MySQL 5.7

EVE previously used MySQL 5.7 and we keep the solutions around in case something similar may be a problem with MySQL 8.

- MySQL installer is unable to find/install the correct Visual Studio version: "Visual Studio x.x.x has Failing Requirements. Visual Studio version 2010, 2012, 2015 or 2017 must be installed."
  - Solution: Try installing an older version of Microsoft Visual Studio. Using Microsoft Visual Studio 2013 seems to work most of the time.

B R and Native MySQL Passwords

Sometimes, MySQL seems to be set to have SHA2 as the standard password. In that case, a user as generated in Section 7.2 will not work. A new user with a native (i.e. old) MySQL password must be generated in the command line. Below you find instructions how to do that.

1. Add MySQL to the Windows PATH.
   (a) Follow the instructions from this website.
   - You may have to find the path to the `bin` files of MySQL first, it is usually similar to this: `C:\Program Files\MySQL\MySQL Server 8.0\bin`.

2. Open PowerShell.
   (a) Press the Windows Button on your Keyboard.
   (b) Type "PowerShell".
   (c) Select the item in the search result called "PowerShell".
      - The PowerShell window pops up with a blue background.
   (d) In the PowerShell window, type the following commands:
      i. Type `mysql -r -p` and then press ENTER.
         - You will be asked to enter the root password for MySQL we defined in Section 2.2.
         - MySQL should be open and you can write the following command.
      ii. Type:

```
CREATE USER 'yourRuser'@'localhost'
    IDENTIFIED WITH mysql_native_password
    BY 'your.s4ve.DB.password';

GRANT ALL PRIVILEGES ON *.*
    TO 'yourRuser'@'localhost';
```

         - Please use a different password and write it down for later use
      - Now you can access MySQL from within R.

C Optional exercise: Prepare your own Information Screens

We have not added questionnaires and question sets for the introduction and the end of the experiment. You can create copies of existing questionnaires and question sets to create your own introduction information screen and end information screen. Here are a few helpful instructions on how to get you there.

1. In the Project view (4), go to `Assets\ Experiment\ Resources\ Questionnaires\`.
2. Copy `info_movement.xml` file 2 times and rename them.
   (a) In the Project View (5), right click any file and choose "Show in Explorer".
   (b) The folder containing the file opens.

(c) Select the file `info_movement.xml`.

(d) Copy file by pressing "Ctrl" and "C".

(e) Paste file by pressing "Ctrl" and "V".

(f) Right click new copy.

(g) Press "F2" to rename file.

(h) Name questionnaire "info_introduction.xml".

(i) Paste file by pressing "Ctrl" and "V".

(j) Right click new copy.

(k) Press "F2" to rename file.

(l) Name questionnaire "info_end.xml".

3. In Unity, correct the names in new questionnaire `info_introduction.xml`.

   (a) In the Project view (4), open the file `info_introduction.xml`.

   (b) Replace the `Name` of the attribute in the `Questionnaire` node (see Figure 24) with "info_introduction".

   (c) Replace the name in the `QuestionSet` (see Figure 24) with "InfoIntroductionSet".

4. Correct names in new questionnaire `info_end.xml`.

   (a) In the Project view, open the file `info_end.xml`.

   (b) Replace the `Name` of the attribute in the `Questionnaire` node (see Figure 24) with "info_end".

   (c) Replace the name in the `QuestionSet` (see Figure 24) with "InfoEndSet".

5. In the Project view (4), go to `Assets\ Experiment\ Resources\ QuestionSets\`.

6. Copy `InfoMovementSet.xml` file 2 times and rename them.

   (a) Right click any file and choose "Show in Explorer".

   (b) The folder containing the file opens.

   (c) Select the file `InfoMovementSet.xml`.

   (d) Copy file by pressing "Ctrl" and "C".

   (e) Paste file by pressing "Ctrl" and "V".

   (f) Right click new copy.

   (g) Press "F2" to rename file.

   (h) Name question set "InfoIntroductionSet.xml".

(i) Paste file by pressing “Ctrl” and “V”.

(j) Right click new copy.

(k) Press “F2” to rename file.

(l) Name question set “InfoEndSet.xml”.

7. In Unity, correct content of the new question set `InfoIntroductionSet.xml`.

   (a) In the Project view, open the file `InfoIntroductionSet.xml`.

   (b) Replace the `Name` of the attribute in the `QuestionSet` node (see Figure 23) with “InfoIntroductionSet”.

   (c) Replace the `Name` of the attribute in the `InfoScreen` node (see Figure 23) with a meaningful unique name.

   (d) Write your own instruction to the participant in the `Text` node.

8. Correct content of the new question set `InfoEndSet.xml`.

   (a) In the Project view, open the file `InfoEndSet.xml`.

   (b) Replace the `Name` of the attribute in the `QuestionSet` node (see Figure 23) with “InfoEndSet”.

   (c) Replace the `Name` of the attribute in the `InfoScreen` node (see Figure 23) with a meaningful unique name.

   (d) Write your own instruction to the participant in the `Text` node.

D Document History

Ver.	Authors	Contents
1.3	Jascha Grübel Raphael P. Weibel Victor R. Schinazi	Fix errors in tutorial, update to new version of EVE, Mysql 8, LTS Unity and add *Exploring EVE* section for overview.
1.2	Jascha Grübel Raphael P. Weibel Victor R. Schinazi	Fix errors in tutorial, update to new version of EVE, Unity and move document to LaTeX.
1.1	Raphael P. Weibel Jascha Grübel Victor R. Schinazi	Fix errors in tutorial, react to changes in MySQL, update to new version of EVE, Unity and googledoc version.
1.0	Raphael P. Weibel Jascha Grübel Victor R. Schinazi	Design of tutorial, original draft and googledoc version.