



## RAISE: Rasing environmental knowledge & awareness through an innovative virtual environment



### System Documentation

|                                                                                                                                                       |                                                                                                                                                                                                                                                                           |
|-------------------------------------------------------------------------------------------------------------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Intellectual Output or Activity Number                                                                                                                | PR4                                                                                                                                                                                                                                                                       |
| Short Description                                                                                                                                     | The system documentation supports the long-term maintenance of the 3D virtual world following the conclusion of the RAISE project. It provides educators and professionals technical details on utilising open simulator features for adding new scenarios and languages. |
| Authors (per partner, if more than one partner provide it together)                                                                                   | Dimosthenis Minas (UPatras)<br>Xenos Michalis (UPatras)<br>Panagiotis Vrettis (UPatras)<br>Marianna Stamati (UPatras) Reviewer 1<br>Manolis Petrakis (Arsakeio) Reviewer 2                                                                                                |
| Status<br>(D: Draft<br>RD: Revised Draft<br>F: Final)                                                                                                 | F                                                                                                                                                                                                                                                                         |
| Dissemination Level<br>(P: Public<br>C: Confidential, only for members of the consortium, National Agency, Commission services and project reviewers) | P                                                                                                                                                                                                                                                                         |





|                  |                                                        |            |                                               |
|------------------|--------------------------------------------------------|------------|-----------------------------------------------|
| <b>File Name</b> | <b>RAISE_Del-01_PR4_System_Documentation_v1.0.docx</b> |            |                                               |
| <b>Date</b>      | PR4 ver0.1                                             | 02/09/2024 | Initial draft version                         |
|                  | PR4 ver0.1 Peer review                                 | 06/09/2024 | Reviewed by Marianna Stamati                  |
|                  | PR4 ver 0.2                                            | 13/09/2024 | Revised upon Marianna Stamati recommendation  |
|                  | PR4 ver 0.2 Peer review                                | 19/09/2024 | Reviewed by Manolis Petrakis                  |
|                  | PR4 ver 0.3                                            | 11/11/2024 | Revised upon Manolis Petrakis recommendations |
|                  | PR4 ver 1.0                                            | 28/11/2024 | Final version                                 |





## Contents

|                                                                     |    |
|---------------------------------------------------------------------|----|
| Executive Summary .....                                             | 5  |
| List of abbreviations .....                                         | 6  |
| List of Figures.....                                                | 7  |
| List of Tables.....                                                 | 8  |
| Introduction.....                                                   | 9  |
| 2. OpenSimulator Components .....                                   | 11 |
| 2.1 Server.....                                                     | 11 |
| 2.2 Clients .....                                                   | 11 |
| 2.3 Services.....                                                   | 11 |
| 2.4 Avatars.....                                                    | 12 |
| 2.5 Server Technical Requirements.....                              | 13 |
| 2.6 Viewer Technical Requirements.....                              | 14 |
| 2.7 Expandability .....                                             | 14 |
| 3. Instructions for using the environment.....                      | 16 |
| 3.1 Access the environment .....                                    | 16 |
| 3.2 Language selection .....                                        | 17 |
| 3.3 Tutorial .....                                                  | 17 |
| 3.4 Scenarios .....                                                 | 19 |
| 3.5 Interaction Between Non-Player Characters and Players .....     | 22 |
| 3.6 NPCs Responses .....                                            | 22 |
| 3.7 Limitations of NPCs Responses.....                              | 22 |
| 3.8 Interaction Between the Player and the Virtual Environment..... | 23 |
| 3.9 Environment Responses .....                                     | 24 |
| 3.10 Avatar Movement.....                                           | 24 |
| 3.11 Enable WASD Movement .....                                     | 25 |
| 3.12 Camera Controls .....                                          | 25 |
| 4. Scenarios.....                                                   | 27 |
| 4.1 Dialogue scripts .....                                          | 27 |
| 4.2 NPCs.....                                                       | 28 |





|                                                                              |    |
|------------------------------------------------------------------------------|----|
| 4.3 Scenario occupation sensors .....                                        | 28 |
| 4.4 Player idle sensors .....                                                | 29 |
| 4.5 Interactive objects .....                                                | 31 |
| 4.6 Scenario isolation AABBs .....                                           | 33 |
| 4.7 Scenario enter and exit buttons .....                                    | 34 |
| 5. Adding Language .....                                                     | 36 |
| 5.1 Languages .....                                                          | 38 |
| 5.2 Visual Change .....                                                      | 43 |
| 6. Adding Scenarios .....                                                    | 45 |
| 6.1 Game Mechanics Guidelines for New Scenarios .....                        | 47 |
| 6.2 General Guidelines when writing a scenario .....                         | 47 |
| 6.2.1 Defining Educational Objectives .....                                  | 47 |
| 6.2.2 Structuring the Scenario Flow .....                                    | 48 |
| 6.2.3 Engagement and Motivation .....                                        | 48 |
| 6.3 Ensuring Engagement .....                                                | 49 |
| 6.4 Assessing Learning Outcomes .....                                        | 49 |
| 6.5 Creating a Seamless Experience .....                                     | 50 |
| 7 Local Setup of RAISE virtual world .....                                   | 51 |
| 7.1 Repository Structure and Development Tools for RAISE Virtual World ..... | 55 |





### Executive Summary

The system documentation is designed to ensure the ongoing sustainability of the 3D virtual world learning environment after the RAISE project concludes. It offers technical instructions on utilizing open simulator features to add new scenarios and languages.

#### Document Versions and Change Explanations:

- Version 0.1: Initial version of the system documentation.
- Version 0.2: Revised upon Marianna Stamati recommendation.
- Version 0.3: Revised upon Manolis Petrakis recommendations
- Version 1.0:





### List of abbreviations

|         |                                                     |
|---------|-----------------------------------------------------|
| 3DVWLE  | 3-Dimensional Virtual World Learning Environment    |
| UPAT    | University of Patras                                |
| ISMAI   | University of Maia                                  |
| UPatras | University of Patras (Greece)                       |
| AEDAS   | Agrupamento de Escolas D. Afonso Sanches (Portugal) |
| Manin   | Liceo Gimnasio Statale Daniele Manin                |
| PR      | Project Results                                     |
| PM      | Project Management                                  |
| NPC     | Non-Playing Character                               |
| OAR     | OpenSimulator Archive                               |
| IAR     | Inventory Archive                                   |
| AABB    | Axis-aligned bounding box                           |





## List of Figures

|                                                                                                  |    |
|--------------------------------------------------------------------------------------------------|----|
| Figure 1: OpenSimulator features .....                                                           | 10 |
| Figure 2: Choose a language by click on the corresponding country's monument .....               | 17 |
| Figure 3: Notecard upon entry to the tutorial room .....                                         | 18 |
| Figure 4: Last step of the tutorial, interact with Jane NPC.....                                 | 19 |
| Figure 5: Overview of the scenario selection room .....                                          | 20 |
| Figure 6: Click Play button to enter a scenario.....                                             | 21 |
| Figure 7: The player initiates a dialogue with an NPC, on the top right are the dialogue options | 22 |
| Figure 8: The player is reading a notecard with educational information .....                    | 23 |
| Figure 9: Movement controls .....                                                                | 25 |
| Figure 10: Camera Controls .....                                                                 | 26 |
| Figure 11: Example of dialogue with an NPC .....                                                 | 27 |
| Figure 12: Message informing the player that the scenario is occupied .....                      | 28 |
| Figure 13: Sensor detecting and checking player's position .....                                 | 29 |
| Figure 14: Sensor sending message to enter button .....                                          | 29 |
| Figure 15: Idle sensor .....                                                                     | 30 |
| Figure 16: Idle sensor tracking the player's position.....                                       | 31 |
| Figure 17: Interactive soda can, not clicked .....                                               | 32 |
| Figure 18: Interactive soda can, clicked .....                                                   | 33 |
| Figure 19: AABB (in red) shown around a scenario.....                                            | 34 |
| Figure 20: Enter button of a scenario.....                                                       | 35 |
| Figure 21: Exit button for a scenario .....                                                      | 36 |
| Figure 22: Images in a localised format.....                                                     | 37 |
| Figure 23: MySQL tables of divaopensim database .....                                            | 40 |
| Figure 24: Attributes of updated UserAccounts table .....                                        | 41 |
| Figure 25: Set English language as user language .....                                           | 44 |
| Figure 26: Right Click at created object and select edit button .....                            | 44 |
| Figure 27: Click at Add New Script button.....                                                   | 45 |
| Figure 28: Selecting an object to edit it.....                                                   | 46 |
| Figure 29: Copying an object and pasting it nearby .....                                         | 46 |
| Figure 30: Screenshot from GitHub repository .....                                               | 56 |
| Figure 31: Screenshot from GitHub repository .....                                               | 57 |





## List of Tables

|                                               |    |
|-----------------------------------------------|----|
| Table 1: Server's technical requirements..... | 13 |
| Table 2: Viewer's technical requirements..... | 14 |





## Introduction

The RAISE project created a 3D virtual world learning environment (3D VWLE) that leverages a gamification-based approach. In this environment, students can participate in several educational scenarios that aim to raise their environmental awareness.

This deliverable presents the system documentation aimed at ensuring the sustainability of the 3D virtual world learning environment after the RAISE project concludes. It provides educators and professionals technical instructions on using open simulator features to add new scenarios and languages.

The RAISE virtual world is developed upon OpenSimulator platform. OpenSimulator (<http://www.OpenSimulator.org>), commonly known as Opensim, is a free, open-source 3D application server that enables the creation of 3D virtual worlds where multiple users can interact simultaneously. These virtual worlds can be accessed through various open-source clients and can be kept private behind firewalls or made public. Written in C#, OpenSimulator's framework is designed to be easily extendable via external modules. The OpenSimulator project began in early 2007 as an open-source server counterpart to Linden Lab's Second Life open-source client. As a result, OpenSimulator's architecture closely mirrors that of Second Life, allowing users to create similarly detailed 3D graphical environments at little to no cost, assuming that the necessary hardware, software, and technical skills are available for free.

Avatars in OpenSimulator are fully customizable and resemble those in Second Life. In-world communication primarily relies on text-based tools such as chat and IM. Currently, a reliable free voice service with lip sync is available from Vivox Inc., upon request. A standout feature of OpenSimulator is the Hypergrid protocol, which allows hyperlinking between different Opensim worlds and supports seamless avatar transfers across these worlds. These capabilities, along with the freedom to own, build, and configure virtual worlds, have made OpenSimulator particularly popular within the educational and scientific communities.

OpenSimulator also supports the use of "OAR" (OpenSimulator Archive) files, which enable the saving and loading of entire regions, including objects, terrain, textures, and scripts, making it easier for educators and professionals to share and replicate educational scenarios. Another useful feature is "IAR" (Inventory Archive), allowing users to back up and restore avatar inventories, which is particularly beneficial for managing educational content.

Additionally, OpenSimulator provides extensive support for non-player characters (NPCs) enabling the creation of automated avatars that can be scripted to interact with users, thus adding more dynamic interactions to educational scenarios. These NPCs can act as guides, provide information, or simulate real-world situations in a controlled environment.

The following figure outlines some of the basic features of OpenSimulator:





| OpenSimulator                                                                                                                                                                                                                       |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Architecture                                                                                                                                                                                                                        | Indicative Costs                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         | Key Characteristics                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
| <p><b>Client - Server</b></p> <p><u>Client (or 'viewer'):</u><br/>open-source,<br/>multiplatform, free<br/>downloadable</p> <p><u>Opensim Server<br/>(v.0.7.6.1):</u><br/>open-source,<br/>multiplatform, free<br/>downloadable</p> | <ul style="list-style-type: none"> <li>▪ <u>'Region' (256x256m2)</u><br/>(private, flexible, allows up to<br/>15,000 prims): Free</li> </ul> <p><u>Options:</u></p> <ol style="list-style-type: none"> <li>1. <u>Self-hosted virtual world on own<br/>server(s):</u> Free<br/>(possible staff costs for system<br/>administration and configuration)</li> <li>2. <u>Hosting on vendor server(s)</u><br/>(dozens of vendors can offer<br/>hosting, consulting, design,<br/>development):<br/>Prices range from 15USD/month to<br/>100USD/month</li> </ol> | <ul style="list-style-type: none"> <li>▪ Physics engine:<br/>OpenDynamics (ODE)</li> <li>▪ Text and IM<br/>communication in-world.<br/>Voice not built-in.<br/>(voice service through<br/>external free modules:<br/><i>Vivox, Freeswitch,<br/>Mumble</i>)</li> <li>▪ In-world scripting: Linden<br/>Scripting Language (LSL),<br/>OpenSim Scripting<br/>Language (OSSL), C#</li> <li>▪ Built-in 3D building tool</li> <li>▪ Hypergrid protocol</li> <li>▪ High quality fully<br/>customizable avatars</li> <li>▪ Image, video, audio file<br/>upload (free)</li> <li>▪ 3D COLLADA format</li> </ul> |

Figure 1: OpenSimulator features





## 2. OpenSimulator Components

### 2.1 Server

The RAISE server is responsible for maintaining and updating the virtual world, managing every change made by users or objects within the environment. Because of this role, it is also known as the “simulator”. These changes can originate from various sources, such as user actions (e.g., an avatar pushing an object), scripts interacting with objects, or environmental effects like simulated gravity. The simulator handles all processes related to the physics of the environment and script execution.

Additionally, it manages client requests and can simulate one or more fixed areas of the virtual world, known as “Regions”. Currently, the RAISE server is hosted by University of Patras, but its open architecture allows for the expansion to additional “Regions”.

### 2.2 Clients

The client, or “viewer”, is the software responsible for rendering the 3D graphics of the avatars and the virtual world. Except during login, the client communicates exclusively with the simulator(s) of the virtual world. When connected to one Region of the virtual world, the client is informed of simulation events in neighboring Regions, enabling it to display a seamless 3D environment that extends beyond the capabilities of a single simulator.

The viewer primarily serves as an interface between the user and the simulator(s), providing tools for setting up the client, modifying landscapes, customizing avatars, facilitating communication and movement, in-world 3D building, editing and scripting, file uploads, and social networking. The quality of the graphics can vary significantly depending on the graphics adapter used in different systems.

Since OpenSimulator does not include a viewer in its distribution, users need to download and install many compatible open-source viewers locally to connect to the RAISE virtual world. These viewers differ in certain features, allowing users to choose the one that best suits their preferences.

### 2.3 Services

The backend of the system is composed of Services that provide the necessary resources to the virtual world simulator(s). All resource data elements are permanently stored in more or more databases, each identified by a unique ID known as a Universally Unique identifier (UUID). In the traditional architecture, access to these services is always mediated by the simulator using well-defined Uniform Resource Locators (URLs).

OpenSimulator offers many services, but the most critical for client-server communication include:





- **User Service:** This service persistently stores user data such as names, passwords, and biographies, and provides authentication mechanisms during the initial login to the virtual world.
- **Grid Service:** The grid service maintains information about the coordinates and positions of every region in the virtual world. Simulators from other regions use this information to locate their neighboring regions.
- **Asset Service:** This service is responsible for persistently storing all media data, referred to as “assets” in OS terminology. Assets can include textures, scripts, sounds, and object representations.
- **Inventory Service:** Each avatar has an inventory in the virtual world, functioning like a personal warehouse containing items (objects, scripts, textures, etc.) from the virtual world. This service persistently stores information about these items, with each item referring to a specific asset.

## 2.4 Avatars

The client program allows users to interact with one another through Avatars, which are human-shaped digital figures. Each user account can have only one avatar at a time, though the appearance of this avatar can be altered into as many different forms as the user desires. Avatar forms, like most other elements, can be created by the user through a user-friendly interface. The avatar can be customized with various outfits, interact with other objects and avatars, and even perform selected physical animations.





## 2.5 Server Technical Requirements

Table 1: Server's technical requirements

|                        | Minimum                                                                           | Recommended                                                                       |
|------------------------|-----------------------------------------------------------------------------------|-----------------------------------------------------------------------------------|
| Operating System       | Windows over the .NET Framework and on Unix-like machines over the Mono framework | Windows over the .NET Framework and on Unix-like machines over the Mono framework |
| .NET Framework version | 4.6                                                                               | >5.12                                                                             |
| Internet Connection    | Low latency 5000 Kbps up and down                                                 | Low latency 20000 Kbps up and down                                                |
| Memory                 | 1 GB                                                                              | 3 GB                                                                              |
| Disk Space             | 2 GB                                                                              | 5 GB                                                                              |
| Processor              | Dual Core                                                                         | Quad Core                                                                         |

To run OpenSimulator without any problems please ensure that your system meets at least the minimum server requirements.





## 2.6 Viewer Technical Requirements

Table 2: Viewer's technical requirements

|                     | Minimum                                            | Recommended                                         |
|---------------------|----------------------------------------------------|-----------------------------------------------------|
| Operating system    | Windows 8.1, 32-bit                                | Windows 10, 64-bit                                  |
|                     | Linux 64-bit kernel 4.4                            | Linux 64-bit kernel 4.15 or newer (1)               |
|                     | Mac OS 10.11 (El Capitan)                          | Mac OS 10.15 (Catalina)                             |
| Internet Connection | 750 Kbps down, 100 Kbps up                         | 10 Mbps down, 1 Mbps up                             |
| Display             | 1024×768 pixels                                    | 1920×1080 pixels                                    |
| Memory              | 8 GB for 64-bit systems<br>4 GB for 32-bit systems | 16 GB                                               |
| Disk Space          | 1 GB application, 1 GB cache                       | 1 GB application, 10 GB cache                       |
| Processor           | Dual Core @ 2.70 GHz (3)                           | Quad Core @ 3.30 GHz (3)(4)                         |
| Graphics Card       | Nvidia GTX 750Ti                                   | NVidia GeForce GTX 980                              |
|                     |                                                    | AMD Radeon R9 290X<br>1.12 GHz with 4 GB Memory (2) |

To run Firestorm Viewer without any problems please ensure that your system meets at least the minimum system requirements.

## 2.7 Expandability

To ensure the system's flexibility and adaptability across different contexts, transferability was established as a key objective. For this reason, the development team opted for an open platform, specifically OpenSimulator, instead of choosing one of the numerous commercial platforms available on the market. Open Simulator's open-source nature was particularly appealing because it provides the freedom to modify and extend the platform as needed, making it a more versatile and cost-effective solution. This decision was driven not only by the need for openness but also by the potential to customize and enhance the platform to meet evolving project requirements without the restrictions often associated with proprietary platforms.

Additionally, the system was purposefully designed to be scalable and open, allowing it to grow and evolve as new requirements emerge. The architecture supports the expansion of the platform to accommodate multiple languages and the creation of new scenarios, which ensures that it can be tailored to diverse educational or experiential settings. This forward-thinking





design was implemented with the aim of maximising the system's usability and exploitability, ensuring that it remains a valuable tool even after the conclusion of the EU funding period.

A critical component of this approach is ensuring the long-term sustainability of the virtual environment. By building on an open platform and emphasising scalability, the system is well-positioned to continue evolving and serving its users long after the initial project phase has ended. This allows educators, developers, and other stakeholders to continually add value to the environment, expanding its use cases and functionality over time. The following sections provide detailed documentation of the scripts and procedures necessary to achieve this functionality. These scripts can be used in their current form or modified to better suit specific needs, providing a high degree of flexibility to users. This comprehensive approach empowers developers and administrator to maintain, customize, and extend the platform according to these evolving objectives.





### 3. Instructions for using the environment

#### 3.1 Access the environment

The fundamental steps for accessing the environment are thoroughly detailed in the user manual. The essential steps to enter the virtual world are as follows:

**Step 1:**

Download the installation [here](#). The manual contains detailed instructions, but you can also see the basic installation instructions from the [Setup video](#).

**Step 2:**

To enter a 3D virtual world, the first step is to download a viewer. The recommended viewer is the Firestorm Viewer, which can be downloaded [here](#).

**Step 3:**

Set up the Firestorm Viewer by following the instructions provided in the video or manual, and by copying and pasting the following link into Firestorm:

<http://opensim.ceid.upatras.gr:9003/>

**Step 4:**

To log in to the virtual world, you must first create an account. Since this is a virtual environment designed for students aged 10-15 years old, user accounts are moderated, and you will need to contact the administrators informing them about your registration before accessing the virtual world. Please contact through the email [raiseprojecteurope@gmail.com](mailto:raiseprojecteurope@gmail.com).



### 3.2 Language selection

Once you have finished the previous steps of this tutorial, you are ready to log into the virtual world. Upon your initial entry, you will need to select your preferred language from four options: English, Greek, Italian and Portuguese. This selection is made by clicking on the corresponding country's most known monument displayed in the entry room. After choosing a language, there is an exit box that will automatically transport you to the tutorial area.

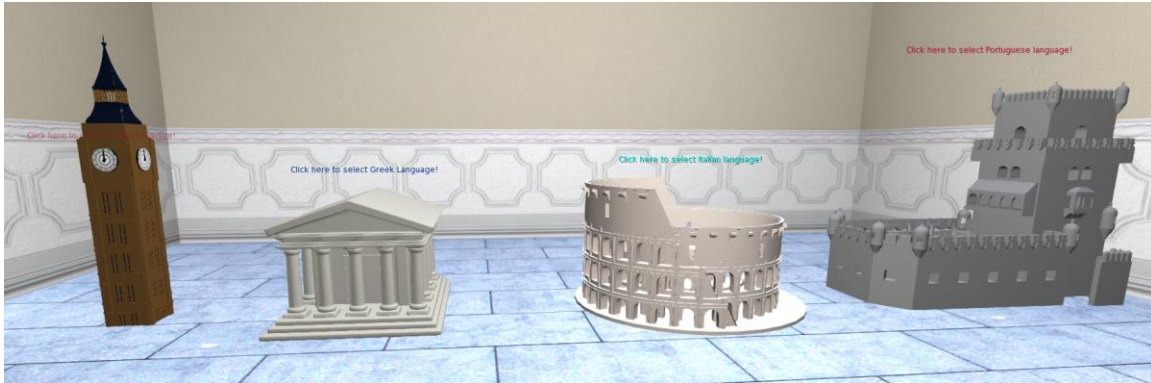


Figure 2: Choose a language by click on the corresponding country's monument

### 3.3 Tutorial

In the tutorial, you will learn the basics of navigating and interacting within the game. The tutorial is composed of 4 steps, each represented by a separate panel with an accompanying image (Figure 3). To begin, click on the first panel to receive instructions, then proceed to the next. Carefully read the instructions while observing the corresponding images. Some instructions are optional. To complete the tutorial, ensure that you click on all the panels, and finally, interact with the NPC inside the tutorial to confirm your understanding of the basic mechanics of the OpenSimulator platform.

The primary goals of the tutorial are:

- Become familiar with avatar navigation
- Learn how to interact with NPC's
- Learn how to enter an educational scenario.



Figure 3: Notecard upon entry to the tutorial room



Figure 4: Last step of the tutorial, interact with Jane NPC

### 3.4 Scenarios

A user has the option to participate in the available scenarios that are in the central scenario room. After completing the tutorial, you will be teleported to the central scenarios room (Figure 5), where you can choose to play one of the available scenarios. Each scenario is limited to one player at a time. To begin a scenario, click the “play” button located at the bottom right of the scenario’s image (Figure 6). Once you complete a scenario, you will be prompted to teleport back to the scenario selection room.



Figure 5: Overview of the scenario selection room



Figure 6: Click Play button to enter a scenario

### 3.5 Interaction Between Non-Player Characters and Players

Players can interact with non-player characters (NPCs) by clicking on them. This action will trigger a dialogue box to appear, presenting the player with various options or responses to choose from (Figure 7, top right corner). Selecting a response will advance the conversation.



Figure 7: The player initiates a dialogue with an NPC, on the top right are the dialogue options

### 3.6 NPCs Responses

NPCs can react to a click or a selected dialogue option by:

- Displaying a new dialogue,
- Playing a specific sound,
- Providing a notecard,
- Changing their position (with limitations noted in 3.7)
- Performing an animation (with limitations noted in 3.7)

Or by combining any of these actions.

### 3.7 Limitations of NPCs Responses

When creating a scenario, certain technical limitations should be considered:

- The text for each dialogue screen is limited to 256 characters (Figure 7, top right corner)
- Sounds can be no longer than 10 seconds. Sound is emitted from an object (such as the NPC), and the volume decreases as the player moves away from the object.

- Notecards function like physical letters that the player receives and stores in their inventory (Figure 8).
- NPCs can change their positions, but it is recommended to minimize movements due to technical constraints.
- NPCs can perform animations, but there is a limited selection available, and it is advisable to use as few animations as possible due to technical considerations.



Figure 8: The player is reading a notecard with educational information

### 3.8 Interaction Between the Player and the Virtual Environment

Players can initiate interactions with the environment by:

- Clicking on an object.
- Moving through a specific area (e.g. passing through a door)



### 3.9 Environment Responses

The environment can respond in the following ways:

- Displaying a dialogue.
- Playing a specific sound.
- Providing a notecard.

The same limitations that apply to NPC responses are also applicable here.

### 3.10 Avatar Movement

- Walk forward/backward: ↑/↓
- Turn left/right: ←/→
- Run: Double-tap Walk forward/backward key (some viewers: ctrl-R to turn run on/off)
- Jump or fly up: page up
- Crouch or fly down: page down
- Sit: Right-click on self/object/ground and select "Sit" (some viewers: ctrl-S to sit/stand)
- Stand: Right-click on self and select "Stand"

By selecting the movement icon (depicted as a person in motion, at the bottom of the screen, see Figure 9), a dialogue window will emerge displaying the movement controls (highlighted in red). Within this panel, you have the option to command your avatar to walk, run, or take flight, and to navigate freely in any direction.





Figure 9: Movement controls

### 3.11 Enable WASD Movement

To use WASD keys instead of the arrow keys for movement, you can enable this option by adjusting the following setting:

Go to Avatar -> Preferences -> Move & View -> Movement, and then check the “Pressing letter keys affects movement” option.

### 3.12 Camera Controls

To rotate the camera using the left and right arrow keys (←/→) on the keyboard. Scrolling the mouse wheel or sliding two fingers up or down on a trackpad, will adjust the zoom level. Zooming in fully will switch the camera to a first-person view. Alternatively, you can click the camera controls option in the horizontal panel located in the downside of Firestorm interface (highlighted with red)

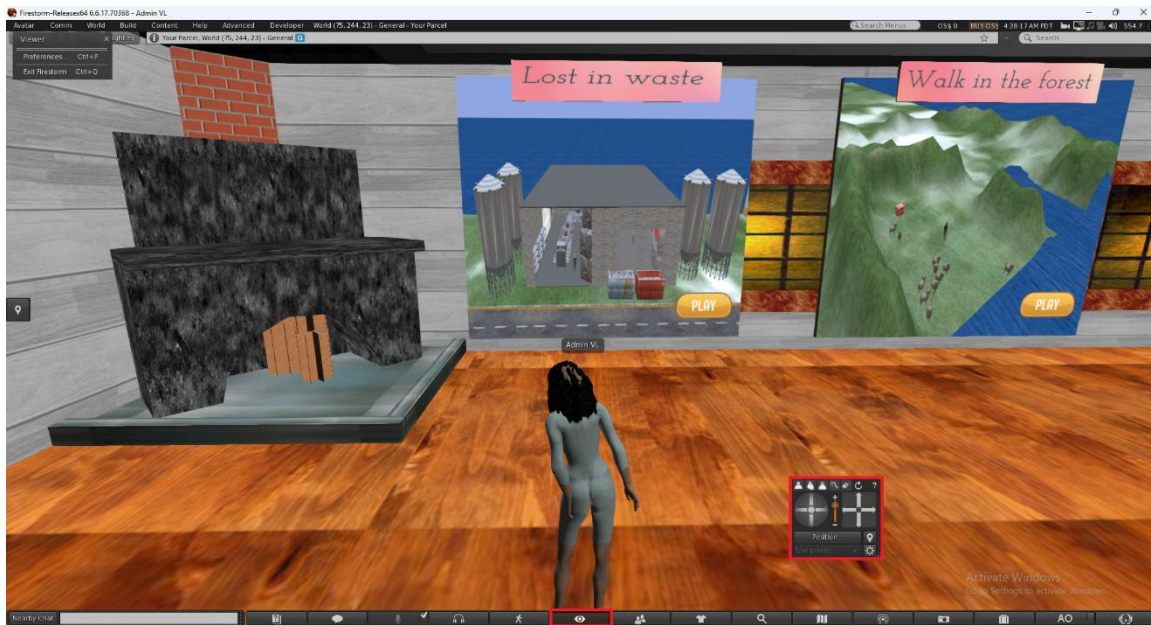


Figure 10: Camera Controls

## 4. Scenarios

All scenarios in the RAISE virtual world learning environment are comprised of the following specific elements:

- Dialogue scripts
- NPCs
- Scenario occupation sensors
- Player idle sensors
- Interactive objects
- Scenario isolation AABBs
- Scenario enter and exit buttons

Some of the aforementioned elements have multiple instances in some scenarios. At least one instance of each element type is required to make up the scenario.

### 4.1 Dialogue scripts

These scripts allow the player to converse with an NPC. The dialogue can follow a linear sequence, or a non-linear one with multiple choices by the player, to fit the scenario's needs.



Figure 11: Example of dialogue with an NPC

Choices are made with the player clicking on a button associated with a certain response. Because of the limitations of OpenSimulator (number of characters in dialog box limit), some pieces of dialogue need to be split between to dialog boxes. In such a case -and when no response is required- the player just clicks on the one button available to continue.

#### 4.2 NPCs

Non-player characters give life to scenarios and aim to increase player engagement. Dialogue and animations are used to represent conversations. Those are utilized to provide context and information to the player (e.g., narrative, educational material, explanations).

#### 4.3 Scenario occupation sensors

To ensure that only the expected number of players are inside a scenario at a time (usually just one), a sensor is used that checks whether the scenario is currently occupied, any time a player uses the enter button to access it. If the scenario is occupied, the player is informed and is not allowed to enter. If a player makes it inside a scenario that is currently occupied, the player is kicked back to the scenario selection room.

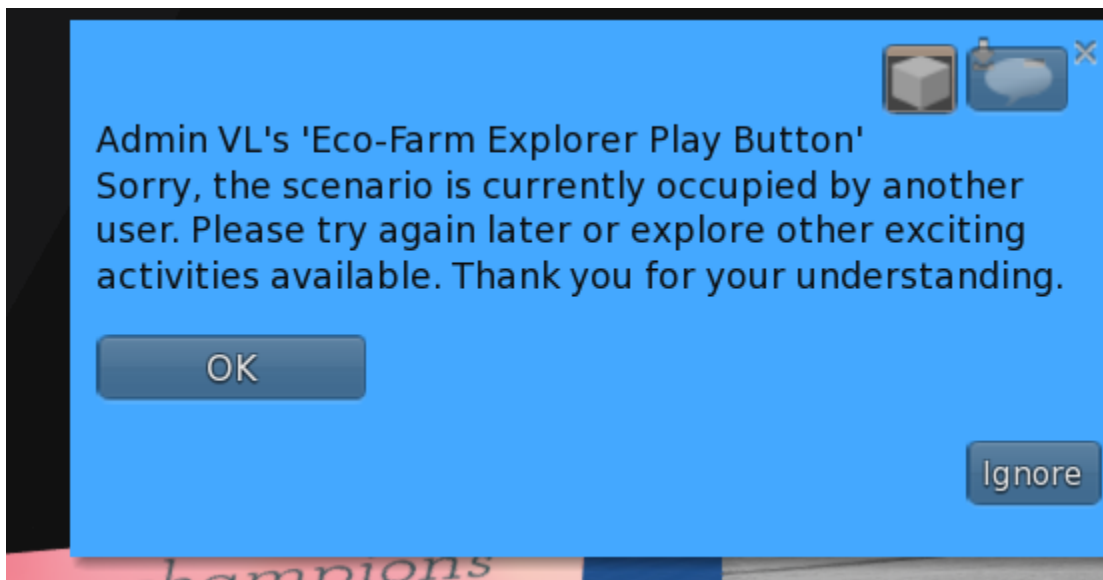


Figure 12: Message informing the player that the scenario is occupied

The first part of the sensor functionality checks whether a detected avatar is within the confines of the scenario (Figure 13). It also checks whether the avatar is one of the NPCs that are part of the scenario, as those would have to be excluded from detection (they are also seen as avatars in our implementation). This functionality is triggered by a message that is sent to the sensor by the enter button of the scenario (on demand).

```
sensor(integer num_detected)
{
    vector userPos;
    integer j;
    for (j=0; j < num_detected; j++)
    {
        key user_in = llDetectedKey(j);
        string detectedUser = llDetectedName(j);

        if (llListFindList(npUsers, [detectedUser]) == -1) // If detected user is not NPC
        {
            // Check if the avatar's position is within the AABB
            userPos = llList2Vector(llGetObjectDetails(user_in, [OBJECT_POS]), 0);
            // llOwnerSay("User's position is: " + (string)userPos);

            if (userPos.x >= 58 && userPos.x <= 110 && userPos.y >= 10 && userPos.y <= 29 && userPos.z >= 1 && userPos.z <= 32)
            {
                // Here are the real users in the scenario area
                realUsersInArea += [user_in];
                llOwnerSay(detectedUser);
                // getLang(user_in);
            }
        }
    }
}
```

Figure 13: Sensor detecting and checking player's position

The second part sends a message to the scenario enter button indicating whether it's occupied or free (Figure 14). The player is teleported inside the scenario accordingly.

```
// Scenario is occupied
if (llGetListLength(realUsersInArea) >= 1)
    llRegionSay(-900, "occupied");
// Scenario is free
else
    llRegionSay(-900, "free");

realUsersInArea = [];
}
```

Figure 14: Sensor sending message to enter button

#### 4.4 Player idle sensors

To handle the case where the player becomes inactive for too long (e.g., because they are away from their computers), we developed sensors that track the player's position in the scenario and check for idleness. If the player is determined to be idle for too long (most scenarios allow 7 to 10 minutes of lack of movement), the player is kicked back to the scenario selection room to free the scenario to other players.

The functionality is very similar to the scenario occupation sensors, with the distinction that we are now tracking the player's position upon detection (Figure 16).

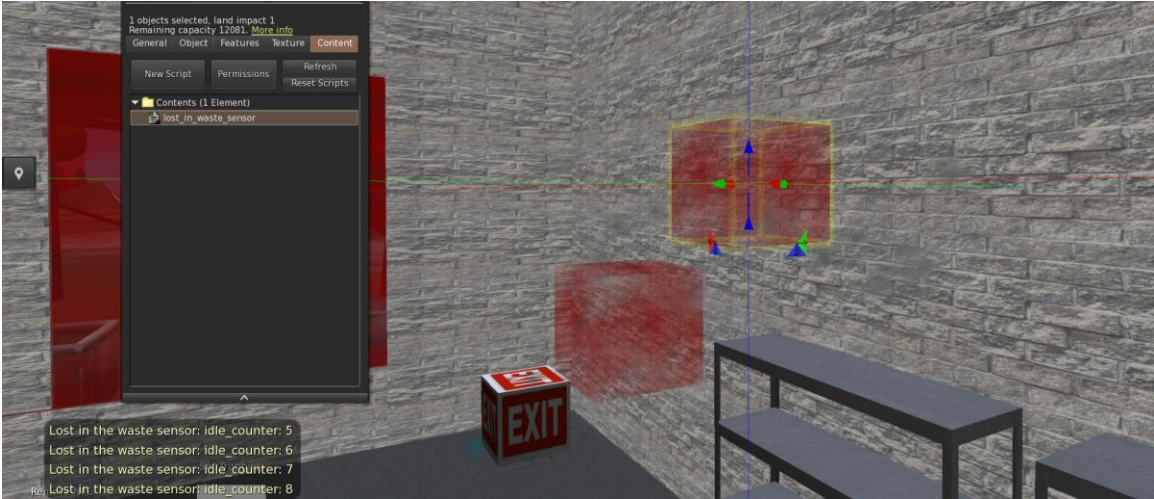


Figure 15: Idle sensor

```
realUsersInArea = [];  
  
// Check idle status  
if (userPos == user_prev_pos)  
{  
    user_idle_counter++;  
    llOwnerSay("idle_counter: "+user_idle_counter);  
  
    // Check whether to kick user from scenario  
    if (user_idle_counter >= idle_counter_limit)  
    {  
        llTeleportAgent(user, "", LandingPoint, LookAt);  
        llInstantMessage(user, llList2String(inform_user, Lang - 1));  
        user_idle_counter = 0;  
        passed = 0;  
        resetRoom(user);  
  
        // Reset Ranger NPC  
        llSay(-800, "dev_reset");  
    }  
}  
else  
{  
    llOwnerSay("idle_counter: reset");  
    user_prev_pos = userPos;  
    user_idle_counter = 0;  
}
```

Figure 16: Idle sensor tracking the player's position

## 4.5 Interactive objects

Many scenarios implement puzzles and minigames to increase interactivity and player experience. To do this, we use interactive objects that are scripted to follow specific behavior. Some objects can be selected and placed in other slots by clicking on them and then the appropriate placement. Others act as buttons that make changes in the scenario.



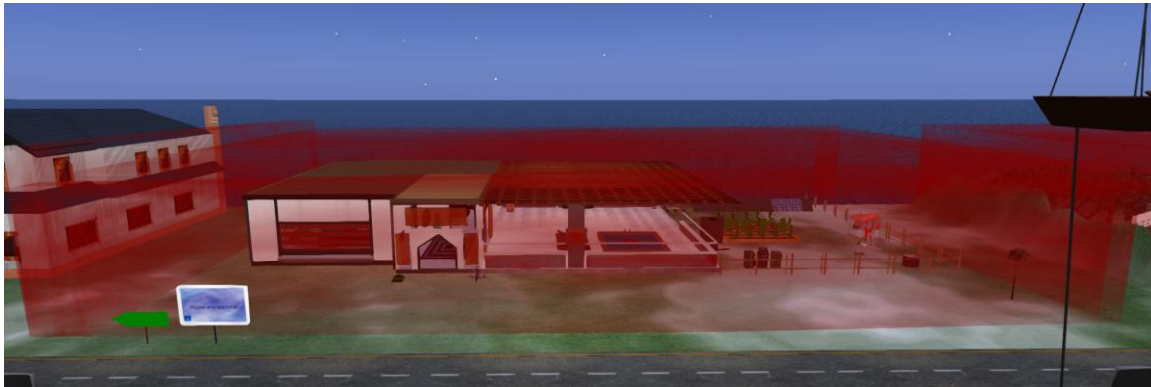
Figure 17: Interactive soda can, not clicked



Figure 18: Interactive soda can, clicked

#### 4.6 Scenario isolation AABBs

To guarantee that each scenario stays isolated from the wider open world, we need to keep players outside unless they follow the proper procedure to enter the scenario (i.e., using the appropriate button in the scenario selection room). To achieve that, we utilize an invisible AABB (Axis-aligned bounding box) around each scenario that prevents any player from walking inside it. If a player manages to “glitch” through (Open Simulator’s collision calculations are very lacklustre), the sensor will detect them and immediately kick them.



*Figure 19: AABB (in red) shown around a scenario*

#### 4.7 Scenario enter and exit buttons

The proper procedure to enter a scenario is to use the appropriate enter button found in the scenario selection room. Doing so will teleport the player to the scenario (if it's free from another player) and give them a short introduction dialog box for context.



Figure 20: Enter button of a scenario

To exit the scenario at any time, the player can click on the exit cubes found around the area.



Figure 21: Exit button for a scenario

## 5. Adding Language

The RAISE virtual world learning environment is designed to be localized to multiple languages. Since dialogue is the cornerstone of all scenarios, text needs to be translated to the language of the target audience. We already provide the scenarios in 4 languages (English, Italian, Portuguese, and Greek). The language selection room allows the player to select their desired language before entering the rest of the virtual environment. Languages are represented by a number from 1 to 4.

To add another language, you will have to first allow the players to select it in the language selection room. That should send a request to the database to set the player's language to your new desired index (e.g., 5). After that, you will need to translate the content of the scenarios to your language and add it to the scripts (primarily the dialogue scripts). All pieces of dialogue are given as lists, where the index represents the language index – 1. All you need to do, is add your text to those lists. For the notecards and images, they need to be provided in the content tab of the appropriate object with a file name that ends on an index that is related to the language (e.g., notecard\_forest4). The index of the currently selected language can then be appended in the script line that gives the notecard or image to the player. Finally, all buttons need to be provided with the translations as well.

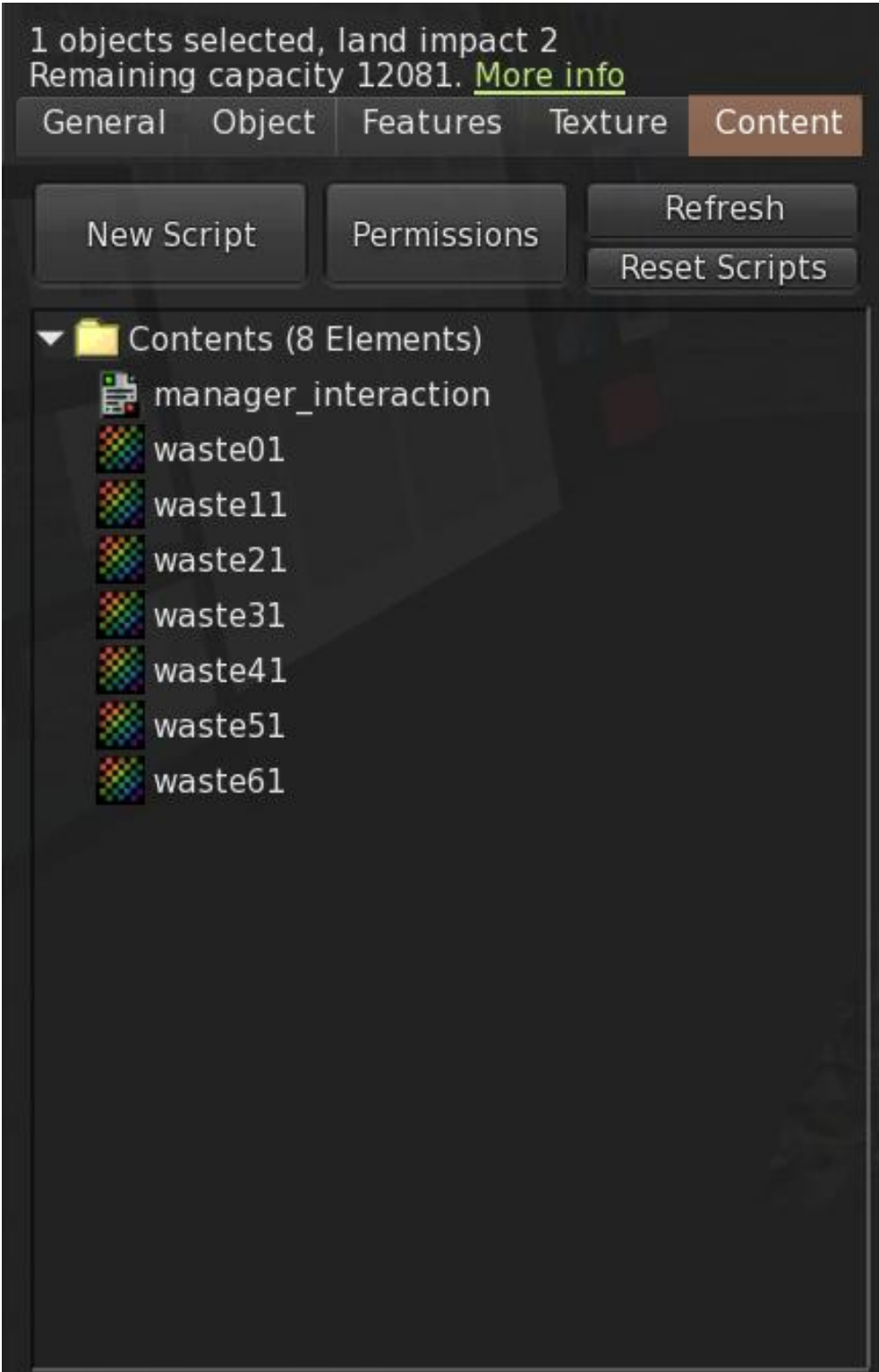


Figure 22: Images in a localised format

## 5.1 Languages

The OpenSimulator is built on MySQL and has been modified (as shown below) to meet the specific requirement of the web server. This server utilizes PHP scripts to handle three variables – lang, room, role- which are passed through an initiator script. These variables are used to store or update the user preferences or user progress within its scenario. These variables are defined as follows:

- Lang: user's language preference
- Role: used to the multiplayer scenario to define the role preference of the user
- Room: used to store the user progress within each scenario

Below is an example of the PHP script (osInitiate.php) that updates the database:

```
<?php
$servername = "mariadb";
$username = "root";
$password = "mariadb";
$dbname = "divaopensim";

// Create connection
$conn = mysqli_connect($servername, $username, $password, $dbname);
// Check connection
if (!$conn) {
    die("Connection failed: " . mysqli_connect_error());
}

$sql_lang = "ALTER TABLE UserAccounts ADD COLUMN IF NOT EXISTS lang
TINYINT(4) NOT NULL DEFAULT 0";
$sql_room = "ALTER TABLE UserAccounts ADD COLUMN IF NOT EXISTS room INT(11)
NOT NULL DEFAULT 0";
$sql_role = "ALTER TABLE UserAccounts ADD COLUMN IF NOT EXISTS role_id
TINYINT(4) NOT NULL DEFAULT 0";
$sql1 = "ALTER TABLE UserAccounts ADD Trophies VARCHAR(30) NOT NULL
DEFAULT '0:0:0:0:0:0:0:0:0:0'";
// $sql_selected_item = "ALTER TABLE prims ADD COLUMN IF NOT EXISTS
end_House_explorer TINYINT(4) NOT NULL DEFAULT 0";

if (mysqli_query($conn, $sql_lang)) {
    echo "Table UserAccounts created successfully the column lang";
} else {
    echo "Error creating table: " . mysqli_error($conn);
}
```

```
if (mysqli_query($conn, $sqlroom)) {
    echo "Table UserAccounts created successfully the column room";
} else {
    echo "Error creating table: " . mysqli_error($conn);
}

if (mysqli_query($conn, $sqlrole)) {
    echo "Table UserAccounts created successfully the column role_id";
} else {
    echo "Error creating table: " . mysqli_error($conn);
}

if (mysqli_query($conn, $sql1)) {
    echo "Field Trophies added successfully";
} else {
    echo "Error creating table: " . mysqli_error($conn);
}

// if (mysqli_query($conn, $sql_selected_item)) {
//     echo "Table prims created successfully the column
// selected_item_flag";
// } else {
//     echo "Error creating table: " . mysqli_error($conn);
// }

mysqli_close($conn);

?>
```

To further explain, the `osInitiate.php` script is the script that updates and add attributes to the build-in MySQL table of OpenSim with the name `UserAccounts`. After that, we have created the `osLang.php` script that builds on the initialisation done by the previous `osInitiate.php` script. While `osInitiate.php` sets up the necessary database structure, including the `lang` column, the `osLang.php` script is responsible for handling the retrieval and updating of language preferences for individual users. It processes the avatar's ID and language preference passed through a URL, updating the database with the appropriate language setting or retrieving the stored value. This script is called through a function like `getLang()`, which ensures that language settings are dynamically managed for each user during interaction within the OpenSimulator environment.



```
MariaDB [divaopensim]> SHOW TABLES;
+-----+
| Tables_in_divaopensim |
+-----+
| Avatars                |
| Friends                |
| GridUser               |
| Presence               |
| UserAccounts           |
| assets                 |
| auth                   |
| bakedterrain           |
| classifieds            |
| estate_groups          |
| estate_managers        |
| estate_map             |
| estate_settings        |
| estate_users           |
| estateban              |
| hg_traveling_data      |
| im_offline             |
| inventoryfolders       |
| inventoryitems         |
| land                   |
| landaccesslist         |
| migrations             |
| os_groups_groups       |
| os_groups_invites      |
| os_groups_membership   |
| os_groups_notices      |
| os_groups_principals   |
| os_groups_rolmembership|
| os_groups_roles        |
| primitems              |
| prims                  |
| primshapes              |
| regionban              |
| regionenvironment      |
| regionextra            |
| regions                |
| regionsettings         |
| regionwindlight        |
| spawn_points           |
| terrain                |
| tokens                 |
| userdata               |
| usernotes              |
| userpicks              |
| userprofile            |
| usersettings           |
+-----+
46 rows in set (0.00 sec)
```

Figure 23: MySQL tables of divaopensim database



```
MariaDB [divaopensim]> DESCRIBE UserAccounts;
```

| Field       | Type        | Null | Key | Default             | Extra |
|-------------|-------------|------|-----|---------------------|-------|
| PrincipalID | char(36)    | NO   | PRI | NULL                |       |
| ScopeID     | char(36)    | NO   |     | NULL                |       |
| FirstName   | varchar(64) | NO   | MUL | NULL                |       |
| LastName    | varchar(64) | NO   | MUL | NULL                |       |
| Email       | varchar(64) | YES  | MUL | NULL                |       |
| ServiceURLs | text        | YES  |     | NULL                |       |
| Created     | int(11)     | YES  |     | NULL                |       |
| UserLevel   | int(11)     | NO   |     | 0                   |       |
| UserFlags   | int(11)     | NO   |     | 0                   |       |
| UserTitle   | varchar(64) | NO   |     |                     |       |
| active      | int(11)     | NO   |     | 1                   |       |
| lang        | tinyint(4)  | NO   |     | 0                   |       |
| room        | int(11)     | NO   |     | 0                   |       |
| Trophies    | varchar(30) | NO   |     | 0:0:0:0:0:0:0:0:0:0 |       |

14 rows in set (0.00 sec)

Figure 24: Attributes of updated UserAccounts table

Below is an example of the PHP script (osLang.php) that returns the user language or sets the user language:

```
<?php
$servername = "mariadb";
$username = "root";
$password = "mariadb";
$dbname = "divaopensim";

// Create connection
$conn = mysqli_connect($servername, $username, $password, $dbname);
// Check connection
if (!$conn) {
    die("Connection failed: " . mysqli_connect_error());
}

//stores url properties in variables
$idAvatar = $_GET["idAvatar"];
$lang = $_GET["lang"];

//$room = $_GET["room"];
// if id and lang are set find id
if(isset($idAvatar ) && isset($lang))
```



```
{
    // find id in db
    $myQuery = "SELECT * FROM UserAccounts WHERE
`PrincipalID`='$idAvatar'";
    $result = mysqli_query($conn,$myQuery) or die('Error searching the
avatar');

    if (mysqli_num_rows($result) == 1)
    {
        // Puts, updates the lang in the table
        $myQuery ="UPDATE UserAccounts SET lang ='$lang' WHERE PrincipalID
='$idAvatar'";
        if ($conn->query($myQuery) === false)
        {
            echo "Error updating record: " . $conn->error;
        } else if($conn->query($myQuery)=== true && $lang === "01"){
            echo "You have successfully chosen English as your language
Click!";
        } else if($conn->query($myQuery)=== true && $lang === "02"){
            echo "Έχετε επιλέξει τα Ελληνικά ως την γλώσσα επιλογή σας
Πατήστε!";
        } else if($conn->query($myQuery)=== true && $lang === "03"){
            echo "Avete scelto con successo l'italiano come lingua
Cliccare su";
        } else if($conn->query($myQuery)=== true && $lang === "04"){
            echo "Você escolheu com sucesso o português como seu idioma.
Clique";
        }
    }
}

/*
if (mysqli_num_rows($result) == 0)
{

    // Inserts id and lang in the table
    $myQuery = "INSERT INTO useraccounts (PrincipalID, lang)
VALUES ('$idAvatar', '$lang')";
    if (mysqli_query($conn, $myQuery))
    {

    }

}

else
```





```
{
    echo "Error: " . $myQuery . "<br>" . mysqli_error($conn);
}
}
*/

}

// Returns the lang value stored in the table
if(isset($idAvatar ) && !isset($lang))
{
    $myQuery = "SELECT * FROM UserAccounts WHERE
`PrincipalID`=' $idAvatar'";
    $result = mysqli_query($conn,$myQuery) or die('Error getting the
language for this avatar');
    $row = mysqli_fetch_assoc($result);
    echo $row["lang"];
}

mysqli_close($conn);
?>
```

Similarly, different script save the user's progress and retrieve mechanisms.

## 5.2 Visual Change

To add a new language, you first need to provide the player/user with an option to select it. Begin creating a new test object to simulate the addition of a new language and alter its script to meet the new language needs. To do that right-click to the new created object, go to content tab and click the button add script to create a new script.

In the next image, I have provided the actual code from the English language. This code shows how, when the user clicks the corresponding document, the English language is set.



```
Set_Language_Room > set_greek_language.lsl
1  list MENU = ["OK"];
2  integer dialogChannel;
3  key user;
4
5  default
6  {
7      state_entry()
8      {
9          llSetText("Click here to select Greek Language!", <0.0, 0.180, 0.518>, 1); // <0, 0.07, 0.54>
10         dialogChannel = -1 -(integer)("0x" + llGetSubString( (string)llGetKey(), -7, -1));
11     }
12
13     touch_start(integer total_number)
14     {
15         user = llDetectedKey(0);
16         llHTTPRequest("http://lang:80/osLang.php?idAvatar="+ (string)llDetectedKey(0) + "&lang=02", [HTTP_METHOD, "POST"], "");
17     }
18
19     http_response(key request_id, integer status, list metadata, string body)
20     {
21         llDialog(user, llStringTrim(body, STRING_TRIM), MENU, dialogChannel);
22     }
23 }
24
25 //language table
26 //English = 01
27 //Greek = 02
28 //Italian = 03
29 //Portuguese = 04
```

Figure 25: Set English language as user language

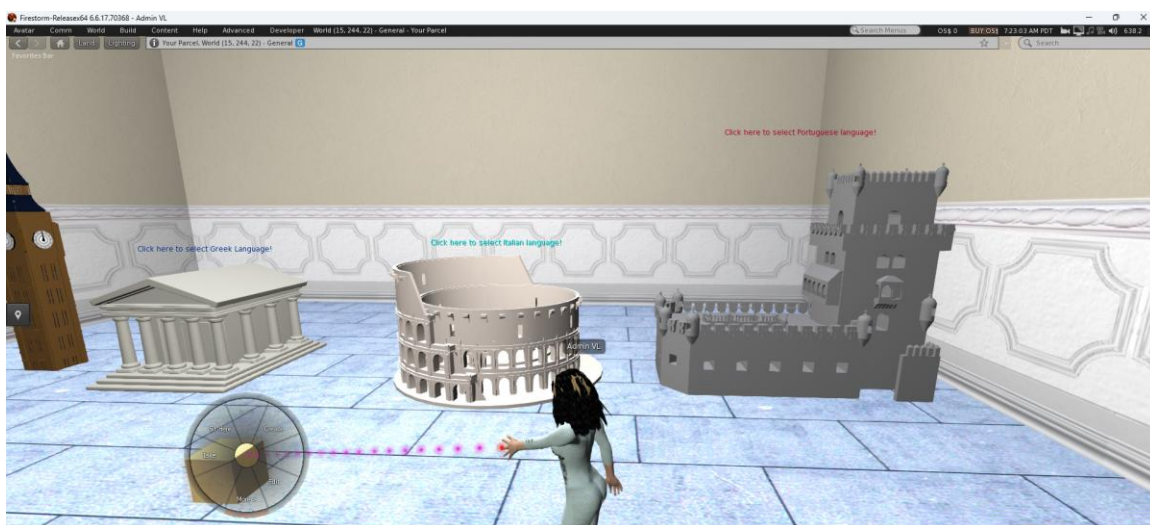


Figure 26: Right Click at created object and select edit button

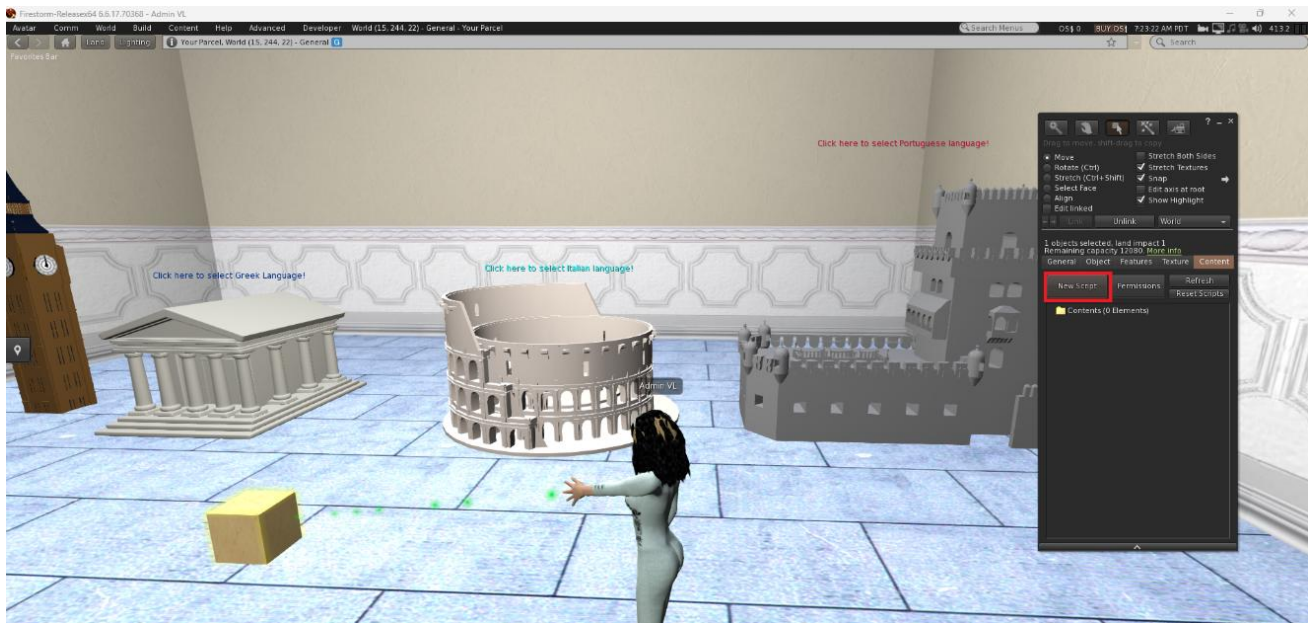


Figure 27: Click at Add New Script button

## 6. Adding Scenarios

To prioritise transferability, an Open Platform (OpenSimulator) was chosen for development rather than one of the available commercial platforms. Additionally, the system is designed to be scalable and open, allowing for the expansion to more languages and the inclusion of additional scenarios. This approach aims to maximize exploitability and ensure the environment's sustainability beyond the EU funding period. Adding a scenario is a simple process for those familiar with OpenSimulator or virtual worlds. OpenSimulator allows in-game object creation and scripting, enabling content creators to log in with their user account (with administrator privileges) and easily edit any existing items in the virtual world (Figure 28). The scripts and functionalities developed for the RAISE 3D VWLE can be easily copied or modified from existing scenarios to create new experiences. For instance, in Figure 29, an avatar with administrator privileges is shown copying an existing object to a nearby location.



Figure 28: Selecting an object to edit it



Figure 29: Copying an object and pasting it nearby

The list of items that can be modified or transferred from existing scenarios include:

- objects,
- buildings,
- interactive items,
- NPCs,
- animations,
- sounds
- educational material.



## 6.1 Game Mechanics Guidelines for New Scenarios

In the early stages of the project, the partners agreed that to fully realise the potential of the 3D VWLE, it was essential to create documentation outlining the capabilities of the RAISE 3D VWLE. This documentation would serve as a guideline for incorporating these features into the scenarios. To achieve this, UPatras, with input from all partners, developed a game mechanics guideline, which has been incorporated into this deliverable in its final form. This guideline is intended to assist anyone looking to use, modify, or create new scenarios that incorporate the game mechanics developed during the RAISE project. It includes information on:

- User interaction with the environment
- Game mechanics that can be integrated into scenarios
- Technical limitations
- Unique features developed specifically for the RAISE project

Moreover, during the development of a scenario, the scenario developer must take into consideration all the capabilities mentioned so far as well as some technical limitations that are naturally imposed by any platform. When writing a player - NPC interaction, the scenario developer must specify how each action results in a response. For example, choosing dialogue option 2 results in a new message, and option 3 results in a movement by the NPC and an alteration of the external environment.

## 6.2 General Guidelines when writing a scenario

When developing a scenario for the RAISE virtual world, the primary objective should always be to maximize the educational value while providing an engaging and interactive experience for the user. Scenarios should be designed to not only immerse students in the virtual environment but also help them learn key concepts related to environmental awareness. Below are the key considerations for writing a successful scenario ensuring that it is both pedagogically sound and effectively structured.

### 6.2.1 Defining Educational Objectives

The first step in designing any scenario is clearly defining the educational objectives. These objectives should align with the overarching goals of the RAISE project-namely, to raise environmental awareness. To achieve this, each scenario should introduce a specific environmental theme (e.g., renewable energy, waste management, biodiversity). Then, we should ensure that the learning outcomes are measurable, for example students should in the end of the scenario be able to explain how solar energy works or describe the lifecycle of a plastic bottle. Finally, the designers should consider breaking down complex topics into manageable sections, using in-game dialogue, interactive elements, and multimedia (such as image or audio) to communicate key points. For example, if a scenario is focused on teaching students about biodiversity, it could begin with a simple introduction about ecosystems,





followed by more complex content such as species interactions, food chains, and the importance of conservation.

### 6.2.2 Structuring the Scenario Flow

Good scenario design requires careful planning of the narrative and structure, ensuring that it flows naturally and keeps the player engaged throughout. A well-structured scenario begins with a brief introduction that provides the player with essential context. This introduction can be delivered through NPC dialogue, and notecards. It should clearly explain the purpose of the scenario, why the topic is important, and what the play through is expected to achieve by the end. For example, if the scenario is focused on teaching renewable energy, the introduction might explain the relevance of sustainable energy sources and what role the player will take in learning about them.

Once the player understands the context, the scenario should be built around a series of tasks and quests that progressively challenge the player's understanding. These tasks must be educational in nature, guiding the player through increasingly complex concepts. For instance, in a renewable energy scenario, the player might begin with basic tasks like identifying the components of a wind turbine and they move on to more advanced challenges such as configuring a wind farm to optimize energy production. This gradual progression ensures that players are building on their knowledge while staying engaged with the scenario. Throughout the scenario, it is essential to provide immediate feedback and interaction to reinforce learning. This feedback can be delivered through NPC dialogue, system notifications, or visual cues in the environment. When the player successfully completes a task, an NPC might offer words of encouragement and explain the next steps, further reinforcing the educational point. Visual or audio cues can also be used to indicate progress or correct errors, helping the player stay on track and understand how their actions are contributing to the overall goal of the scenario. This continuous interaction between the player and the virtual world is crucial for maintaining engagement and ensuring that educational objectives are met.

### 6.2.3 Engagement and Motivation

One of the primary goals of the virtual world is to make learning both fun and interactive, ensuring that educational content fosters curiosity and motivates students to explore the scenarios further. To achieve this, it is important to gamify the learning experience by incorporating game features such as point system, badges, or trophies. These features reward players for completing tasks or reaching learning objectives, helping to motivate students to continue engaging with the content. For instance, in the RAISE virtual world, we implemented a trophy mechanism that awards players with a trophy for the completion of each scenario. These trophies are displayed in a special trophy room within the virtual world, offering students a tangible sense of accomplishment and reinforcing positive behavior as they progress through





the educational content. This system enhances motivation and encourages players to engage deeply with each scenario, knowing their efforts will be recognized and rewarded.

### 6.3 Ensuring Engagement

One of the most important aspects of creating effective educational scenarios in the RAISE virtual world is ensuring that learning is both engaging and entertaining. While the primary goal is to educate, scenarios must captivate the students' interest to prevent them from losing focus. This balance can be achieved by combining educational content with fun, interactive elements such as mini-games, quizzes, and challenges. By integrating these engaging features, scenarios become more dynamic, preventing players from becoming disengaged due to overly text-heavy or repetitive content. These interactive elements not only reinforce learning but also make the educational experience enjoyable.

Another crucial aspect is pacing. Proper pacing ensures that players neither feel rushed nor overwhelmed. The introduction of new concepts should be gradual, giving players enough time to absorb and apply the knowledge they gain before moving on to more advanced material. A scenario length of about 10 to 15 minutes is ideal, as it keeps players engaged while maintaining focus and retention. Well-paced scenarios give students a sense of accomplishment after completing each task, ensuring that they stay motivated and invested in the learning experience.

### 6.4 Assessing Learning Outcomes

To ensure that the educational objectives of each scenario are met, it is important to assess players' understanding at various stages of the scenario. This can be accomplished through multiple methods of assessment, starting with formative quizzes. These small, in-scenario quizzes provide real-time feedback on whether the player has grasped key points. For example, after explaining a concept like waste management, a short quiz can test if the player understands the steps of proper recycling or disposal. These quizzes help identify gaps in knowledge before moving on to the next section of the scenario.

At the end of each scenario, end-of-scenario assessments are vital in determining the overall understanding of the player. These assessments could take the form of a final challenge, where players must apply everything they've learned to a problem-solving task. For example, after completing a renewable energy scenario, the player might be tasked with designing an energy-efficient home, applying all the concepts they've learned about solar panels, wind turbines, and energy conservation.

In addition to these quizzes and assessments, feedback loops should be built into the scenarios. If a player answers a question incorrectly or fails a task, they should be given the opportunity to





retry with additional feedback explaining what went wrong. This not only reinforces learning but also encourages persistence, as players are guided toward understanding through constructive feedback rather than simply penalising mistakes.

### 6.5 Creating a Seamless Experience

To create an engaging and effective learning environment, it is essential that the entire scenario is seamless and intuitive for the player, from the initial setup to the final assessment. A well-designed user interface (UI) plays a key role in achieving this. All buttons, menus, and interactive objects should be clearly labelled and easy to use, with instructions readily available in all supported languages. This ensures that players can navigate the scenario smoothly without confusion, enhancing the overall learning experience.

Error handling is another critical component of creating a seamless experience. Common user errors, such as selecting the wrong answer in a quiz or failing a task, should be handled gracefully. Rather than penalising players, scenarios should provide clear explanations for mistakes, guiding players toward the correct solution and allowing them to try again. This positive reinforcement helps maintain engagement and ensures that learning continues even when errors are made. Finally, every scenario must be thoroughly tested to identify and fix any bugs or glitches before release. Technical issues can significantly hinder the learning process, frustrating players and detracting from the educational experience. Scenarios should be tested in various conditions to ensure smooth gameplay, reliable performance, and the absence of any disruptive errors. By thoroughly testing and refining scenarios, developers can create an environment that promotes learning while minimizing frustrations and distractions.





## 7 Local Setup of RAISE virtual world

This chapter is a detailed step by step tutorial on how to setup RAISE virtual world on your local machine. Before starting to verify that your local system has meet the following prerequisites:

1. Firestorm Viewer is installed. If firestorm is not installed, check this [video](#).
2. Docker Desktop is installed. If Docker is not installed, check this [tutorial](#).

**Step 1:** Clone or download the raise virtual world initialization files. You can find the raise initialization files [here](#).

**Step 2:** Open you cmd and create a docker network with the following command: `docker network create opensim-network`

**Step 3:** create a docker volume with the following command: `docker network create opensim-vol`

**Step 4:** Run the following command at the file path that you have the docker-compose yml file: `docker-compose up -d`

**Step 5:** Build your docker containers to the file path that you have your docker file with the following command: `docker build -t diva`.

**Step 6:** To run the docker containers with the following command:

1. `docker run --name=diva --network=opensim-network -p 9000:9000 -p 9000:9000/udp -it diva` (This command if you don't want the volume attached to the storage of your local machine, but it will not be persistent)
2. `docker run --name=diva --volume=C:\Users\ (your user path) \Dev\localopensim\backup:/diva-r09110/bin/backup --network=opensim-network -p 9000:9000 -p 9000:9000/udp -it diva` (with attached storage and at the myword.ini file the admin user password exists)

**Step 7:** Download the latest raise oar version from [here](#).

**Step 8:** Load the oar at the opensim cmd with the following command: `load oar <filename>`

**Step 9:** Verify that all php files have been loaded to the `raise_lang_1` docker container at path `var/www/html`

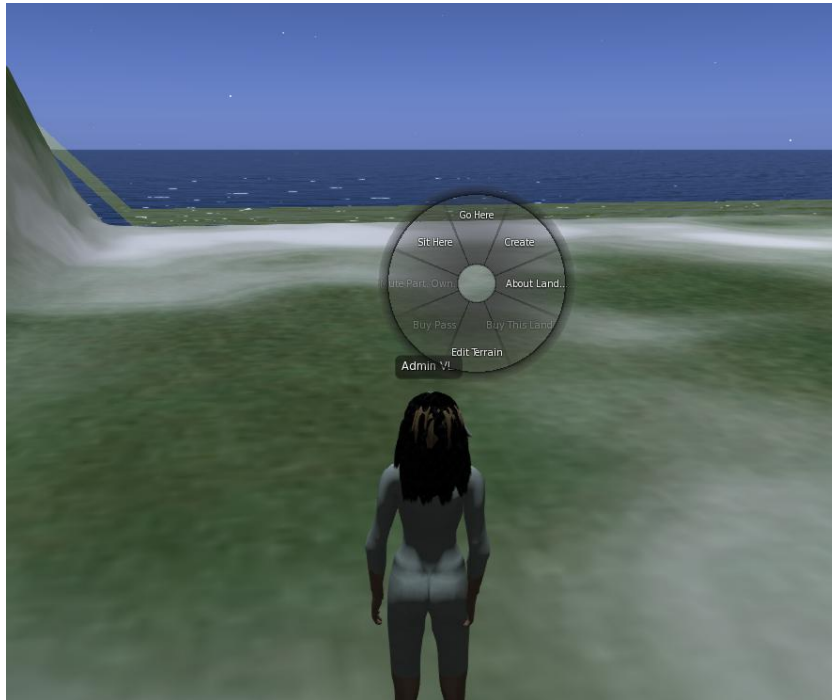
**Step 10:** Initialize the new database by setting the following script to an object.

1. Login as admin user through firestorm to the opensim server.





2. Create an object by right clicking and selecting the option create



3. Right click to the object and select the edit option



4. Go to the content tab and click the new script button and then a new script will be created.
5. Copy the following script and paste it:

```
default
{
    state_entry()
    {

    }

    touch_start(integer total_number)
    {
        llHTTPRequest("http://lang:80/osInitiate.php", [HTTP_METHOD,
"POST"], "");
    }
}
```

6. Click the scripted object.
7. Verify that changes passed by checking the UserAccounts table.
8. Open a new cmd and connect to the database cmd by writing the following command:  
docker exec -it <your docker container name> mysql -u root -p.
9. Default password of MySQL is: mariadb
10. Then write the following command: USE divaopensim;
11. Then write the following command: DESC UserAccounts; and the following answer  
should appear

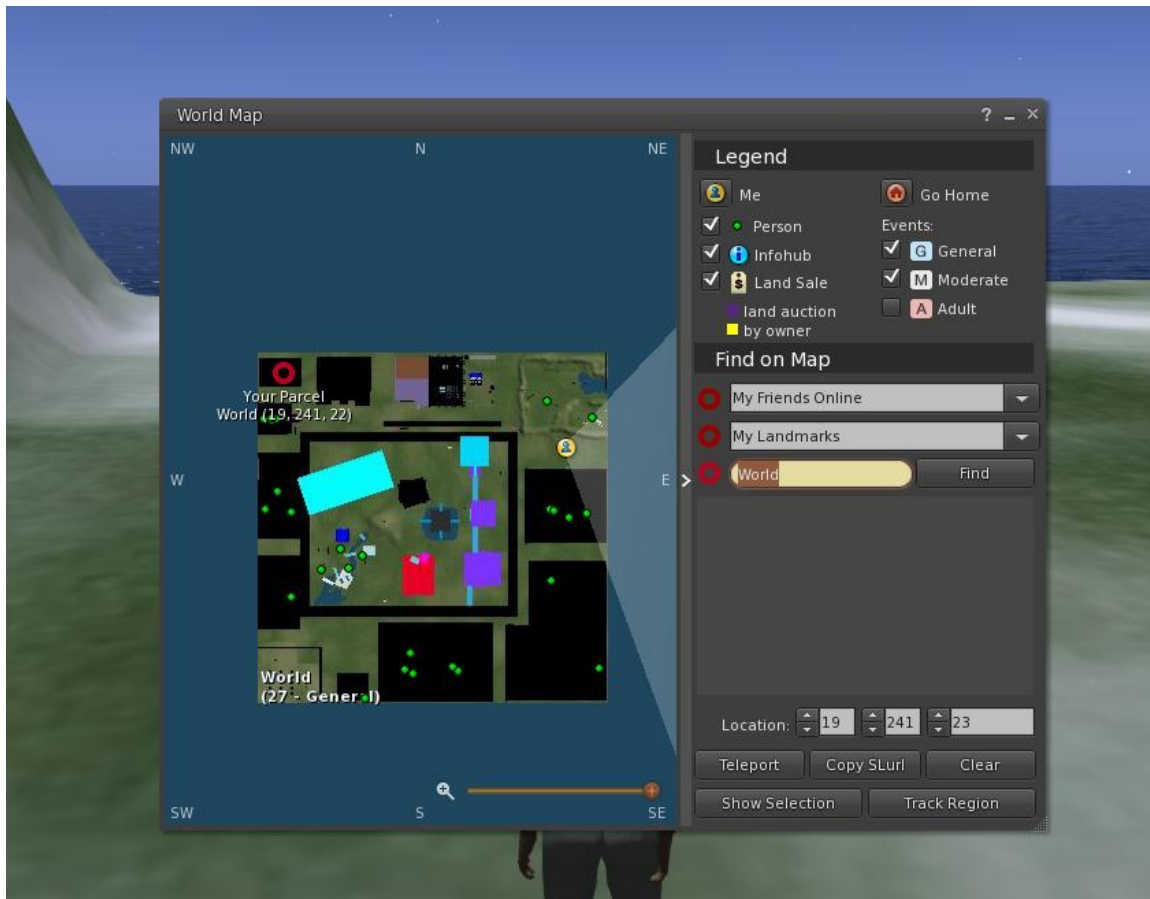
```
MariaDB [divaopensim]> DESC UserAccounts;
```

| Field       | Type        | Null | Key | Default             | Extra |
|-------------|-------------|------|-----|---------------------|-------|
| PrincipalID | char(36)    | NO   | PRI | NULL                |       |
| ScopeID     | char(36)    | NO   |     | NULL                |       |
| FirstName   | varchar(64) | NO   | MUL | NULL                |       |
| LastName    | varchar(64) | NO   | MUL | NULL                |       |
| Email       | varchar(64) | YES  | MUL | NULL                |       |
| ServiceURLs | text        | YES  |     | NULL                |       |
| Created     | int(11)     | YES  |     | NULL                |       |
| UserLevel   | int(11)     | NO   |     | 0                   |       |
| UserFlags   | int(11)     | NO   |     | 0                   |       |
| UserTitle   | varchar(64) | NO   |     |                     |       |
| active      | int(11)     | NO   |     | 1                   |       |
| lang        | tinyint(4)  | NO   |     | 0                   |       |
| room        | int(11)     | NO   |     | 0                   |       |
| role_id     | tinyint(4)  | NO   |     | 0                   |       |
| Trophies    | varchar(30) | NO   |     | 0:0:0:0:0:0:0:0:0:0 |       |

```
15 rows in set (0.01 sec)
```



**Step 11:** Open minimap with shortcut Ctrl + M and then enter the language selection room.



**Step 12:** Select your preferred language by clicking on the corresponding monument.

**Step 13:** You have completed the local setup of the RAISE Virtual World.



## 7.1 Repository Structure and Development Tools for RAISE Virtual World

In this chapter, we provide an overview of the GitHub repository that contains all the scripts and resources used in the development of the **RAISE virtual world**. The repository is organized into several key folders, each playing a vital role in the overall development process. It also includes various tools and scripts used for testing, automation, and backend integration, which ensured the smooth operation of the virtual world and supported the localization and language selection functionalities.

After setting up the language selection functionality, we have also created a GitHub repository containing all the scripts that were incorporated into the RAISE virtual world. The repository is organized into several folders, each corresponding to different aspects of the virtual world:

- **Scenario Folders:** Each folder represents a separate scenario with all relevant scripts.
- **Documentation Folder:** This folder includes basic documentation about the structure of the virtual world, as well as draft notes created by the developers during the development process to automate procedures.
- **General Scripts Folder:** This folder contains various scripts that helped developers during the development phase. These scripts were used primarily for testing, such as those for creating or deleting NPCs or moving them to specific locations.
- **Scenario Room Folder:** This folder holds the scripts used in the central scenario room.
- **Set Language Room Folder:** This folder contains the scripts for the development of the set language room, which is where players can choose their preferred language.
- **PHP Files Folder:** This folder contains the PHP files that handle backend processes, specifically those that manage requests between the virtual world and the database.

Additionally, the repository includes five Python scripts that were used to automate various tasks:

- **Automated Text File Generation:** These scripts were used during the process of translating content.
- **update\_auto\_images.py:** A Python script that resizes images exceeding a certain size limit.
- **get\_breakfast\_list.py:** This script was responsible for generating the images used in the breakfast mini game.

Lastly, there is a Trophy Script Folder, which contains the scripts used in the trophy room for displaying user achievements.

Upon request, we can grant access to this private repository to have a complete view of the scripts that have been incorporated to the RAISE virtual world.





|                                                                                                             |                                                               |               |
|-------------------------------------------------------------------------------------------------------------|---------------------------------------------------------------|---------------|
|  Change at input mechanism | 5261803 · last month                                          | 🕒 352 Commits |
| 📁 .vs                                                                                                       | End of translation until the Visiting Green Town              | 6 months ago  |
| 📁 Carbon_Heroes                                                                                             | Final Version RAISE world                                     | 2 months ago  |
| 📁 Custom DAE objects                                                                                        | Add files via upload                                          | 2 months ago  |
| 📁 Documentation                                                                                             | Update Pap items list.txt                                     | 5 months ago  |
| 📁 Eco_Farm-explorer_Room                                                                                    | Small change at set canvas name reset mechanism               | 2 months ago  |
| 📁 General scripts                                                                                           | Update of languages                                           | 6 months ago  |
| 📁 Green Champions                                                                                           | Continue with translation                                     | 6 months ago  |
| 📁 House_eco-explorer_Room                                                                                   | Continue the Visiting to Green Town                           | last year     |
| 📁 Polluting_Factory                                                                                         | Continue with translation                                     | 6 months ago  |
| 📁 Scenario_room                                                                                             | Change at input mechanism                                     | last month    |
| 📁 Set_Language_Room                                                                                         | Update of languages                                           | 6 months ago  |
| 📁 Visiting_Green_town                                                                                       | End of translation until the Visiting Green Town              | 6 months ago  |
| 📁 Water_treasure                                                                                            | Add Trophies mechanism                                        | 7 months ago  |
| 📁 Wind_Farm_Challenge_Room                                                                                  | Update of languages                                           | 6 months ago  |
| 📁 php_files                                                                                                 | Final Version RAISE world                                     | 2 months ago  |
| 📄 auto_create_txt_files.py                                                                                  | Pretty much to the end of translation                         | 6 months ago  |
| 📄 get_breakfast_list.py                                                                                     | Update of breakfast minigame and upload of new oar            | last year     |
| 📄 resizer_python_script.py                                                                                  | Creation of Beekeeper minigame script and update of inteca... | last year     |
| 📄 test.py                                                                                                   | Update Carbon Heroes                                          | last year     |
| 📄 update_auto_images.py                                                                                     | Update Visiting Green town scenario                           | last year     |

Figure 30: Screenshot from GitHub repository





This branch is 31 commits ahead of master . #45

|                                                                                                                         |                                                                 |                                                                                                 |
|-------------------------------------------------------------------------------------------------------------------------|-----------------------------------------------------------------|-------------------------------------------------------------------------------------------------|
|  added on-demand sensor to scenario 9. | b5b0fc4 · 2 months ago                                          |  245 Commits |
|  Custom DAE objects                    | Add files via upload                                            | last year                                                                                       |
|  Documentation                         | fixed conflicting channel use, and added first minigame impl... | last year                                                                                       |
|  Eco_Farm-explorer_Room                | Update of Cycle lane minigame                                   | last year                                                                                       |
|  General scripts                       | added condition to tutorial exit, and feedback to quizzes.      | 9 months ago                                                                                    |
|  House_eco-explorer_Room               | Continue the Visiting to Green Town                             | last year                                                                                       |
|  Lost_something_in_waste               | added on-demand sensor to scenario 9.                           | 2 months ago                                                                                    |
|  Scenario_room                         | Begin Visiting Green town scenario                              | last year                                                                                       |
|  Set_Language_Room                     | Update final alpha version                                      | last year                                                                                       |
|  Trophy scripts                        | trophy and forest translations done.                            | 6 months ago                                                                                    |
|  Visiting_Green_town                   | End of Scenario 4 mechanisms and save new oar version 0.34....  | last year                                                                                       |
|  Walk_in_the_Forest                    | added on-demand sensors.                                        | 2 months ago                                                                                    |
|  Wind_Farm_Challenge_Room             | Continue the Visiting to Green Town                             | last year                                                                                       |
|  php_files                           | added trophy functionality to vw.                               | 8 months ago                                                                                    |
|  .gitignore                          | trophy and forest translations done.                            | 6 months ago                                                                                    |
|  resizer_python_script.py            | Creation of Beekeeper minigame script and update of inteca...   | last year                                                                                       |

Figure 31: Screenshot from GitHub repository



Co-funded by  
the European Union

**Erasmus+**  
Enriching lives, opening minds.



2022-1-EL01-KA220-SCH-000088295

---

**Disclaimer:** *“The material of the project reflects only the author’s views. The European Commission’s support for the production of this publication does not constitute an endorsement of the contents which reflects the views only of the authors, and the Commission or the Hellenic National Agency cannot be held responsible for any use which may be made of the information contained therein.”*

