Screencapturing system with C# and PHP in your Unity game by Roope Parkkinen

How do we start?

Make an Unity project, with somewhat character controller, in this tutorial a 3rd person controller and a camera. You can get those for free from Unity's standard assets(Download from Asset Store).

Then we build a screencapture-script in Unity with C#. We are going to save the screenshot to C:/tmp/, declare the path in your screencapture-script.

We are going to refer the other C# script from this script.

Attach the script to your player's camera.

You can find the example script with comments below. You can use the Texture2D –method, but in this tutorial we are going to use the ScreenCapture.

Example script for the screen capture

Then we need a website with a database.

I am using XAMPP in this tutorial.

We are going to use Apache and MySQL.

Remember to increase the maximum filesize at your php.ini file to like 10M, that should do the trick.

What do we need on the website and database (PHP, MySQL)?

First we need a database. Name it as you will. We will build one table there named "gallery" with 2 rows: *"id"* (**INT, AI and make it primary**) and *"image"*(**VARCHAR**).

We need two scripts, the page that displays your mighty screenshots and the script that takes care of posting the images and making connection to your database.

Building the form

Let's begin by building the form on the website after the previous steps.

Build the form on the display page.

We need two input fields, one for the file and one for the "upload-button". We are going to use POST as the submitting method.

Style the form as you will.

Example website

Moving onto the Unity again to send the actual screenshot from there

Make a script named “Networking.cs” in Unity.

Basically we need this script to connect to the website’s “post.php” -script and send the screenshot to the server as you press the key.

Example networking script

If you can’t get your images to the server, try uploading images manually to the website.

Then download Wireshark and use “tcp port 80” capture filter in there to see the headers in the POST-form, then try adding more headers to your C# -script.

Some problems occurred within the project:

1. C# doesn’t tell the PHP to press the uploading button, no matter how we tell it to do so. Ended up with deleting the upload button from the post.php file.

    Don’t know the reason to that yet. So make the upload button in the form but don’t tell PHP to check if it has been pressed.