Andrei Inoue Hirata

Sistema de Recordes Genérico

Permitirá que você salve e carregue recordes

Para armazenar dados de um jogo Unity WebGL em um banco de dados MySQL, você não deve conectar o Unity diretamente ao MySQL. Isso exporia suas credenciais do banco de dados, o que é um grande risco de segurança.

A abordagem correta e segura é usar um serviço web intermediário (como scripts PHP, Node.js, Python, etc.) no seu servidor. O jogo Unity se comunicará com esses scripts via HTTP, e os scripts é que farão a interação segura com o banco de dados MySQL.

Sumário

Desenvolvimento WebService

Você precisará de um servidor web com suporte a PHP e acesso a um banco de dados MySQL.

Criando o Sql

1. Crie uma tabela para armazenar as pontuações. Você pode usar uma ferramenta como phpMyAdmin ou executar o seguinte comando SQL:

```sql
CREATE TABLE highscores (

    id INT AUTO_INCREMENT PRIMARY KEY,

    playerName VARCHAR(50) NOT NULL,

    score INT NOT NULL,

    submissionDate TIMESTAMP DEFAULT CURRENT_TIMESTAMP

);
```

Configuração do Banco de Dados (db_config.php)

Este arquivo conterá as credenciais do seu banco de dados e uma função para obter a conexão. Crie um arquivo chamado db_config.php.

1. Copie o código

```php
<?php

$servername = "localhost"; // Ou o host do seu DB

$username = "seu_usuario_mysql";

$password = "sua_senha_mysql";

$dbname = "seu_banco_de_dados";
```

```
// Criar conexão

$conn = new mysqli($servername, $username, $password, $dbname);

// Checar conexão

if ($conn->connect_error) {

    die("Connection failed: " . $conn->connect_error);

}

?>
```

2. Substitua 'seu_usuario_mysql', 'sua_senha_mysql', e 'seu_banco_de_dados' com suas credenciais reais.

Salvar/Atualizar um Recorde Individual (save_score.php)

Este script receberá dados via POST e salvará ou atualizará um recorde na tabela.

1. Escreva o código:

```
<?php

require 'db_config.php';

header('Content-Type: application/json');

header('Access-Control-Allow-Origin: *'); // Permite requisições de qualquer origem (para testes

WebGL). Em produção, restrinja ao seu domínio.

header('Access-Control-Allow-Methods: POST, GET, OPTIONS');

header('Access-Control-Allow-Headers: Content-Type, Authorization, X-Requested-With');

if ($_SERVER['REQUEST_METHOD'] === 'OPTIONS') {

    http_response_code(200);

    exit();

}

$response = array();

if ($_SERVER['REQUEST_METHOD'] === 'POST') {
```

```
    $playerName = isset($_POST['playerName']) ? $conn->real_escape_string($_POST['playerName']) :
null;
    $score = isset($_POST['score']) ? (int)$_POST['score'] : null;
    if ($playerName && $score !== null) {
        $stmt = $conn->prepare("INSERT INTO highscores (playerName, score) VALUES (?, ?)");
        $stmt->bind_param("si", $playerName, $score);
        if ($stmt->execute()) {
            $response['status'] = 'success';
            $response['message'] = 'Score saved successfully.';
            http_response_code(200);
        } else {
            $response['status'] = 'error';
            $response['message'] = 'Error saving score: ' . $stmt->error;
            http_response_code(500);
        }
        $stmt->close();
    } else {
        $response['status'] = 'error';
        $response['message'] = 'Invalid input.';
        http_response_code(400);
    }
} else {
    $response['status'] = 'error';
    $response['message'] = 'Invalid request method.';
    http_response_code(405);
}
$conn->close();
```

```
echo json_encode($response);

?>
```

Carregar um Recorde Individual (get_scores.php)

Este script busca as N melhores pontuações do banco e as retorna como JSON.

1. Digite o código

```
<?php

require 'db_config.php';

header('Content-Type: application/json');

header('Access-Control-Allow-Origin: *'); // Permite requisições de qualquer origem. Restrinja em

produção.

header('Access-Control-Allow-Methods: GET, OPTIONS');

header('Access-Control-Allow-Headers: Content-Type, Authorization, X-Requested-With');

if ($_SERVER['REQUEST_METHOD'] === 'OPTIONS') {

    http_response_code(200);

    exit();

}

$limit = isset($_GET['limit']) ? (int)$_GET['limit'] : 5; // Default 5 scores

if ($limit < 1) $limit = 1;

if ($limit > 10) $limit = 10; // Limite máximo de 10

$scores = array();

$sql = "SELECT playerName, score FROM highscores ORDER BY score DESC LIMIT ?";

$stmt = $conn->prepare($sql);

$stmt->bind_param("i", $limit);

$stmt->execute();

$result = $stmt->get_result();
```

```
if ($result) {

    while ($row = $result->fetch_assoc()) {

        $scores[] = $row;

    }

    http_response_code(200);

    echo json_encode(array("scores" => $scores));

} else {

    http_response_code(500);

    echo json_encode(array("status" => "error", "message" => "Error fetching scores: " . $conn-

>error));

}

$stmt->close();

$conn->close();

?>
```

Observação Importante: As linhas header('Access-Control-Allow-Origin: *'); permitem que seu jogo WebGL (que rodará em um domínio diferente durante o desenvolvimento ou mesmo em produção) acesse esses scripts PHP. Para produção, é mais seguro substituir * pelo domínio específico do seu jogo.

Configuração no Unity (TextMeshPro)

Nessa área iremos trabalhar com a Unity 3D

Importar TextMeshPro

1. Se for um projeto novo, o Unity geralmente pergunta se você quer importar os "TMP Essentials". Caso contrário, vá em Window -> TextMeshPro -> Import TMP Essential Resources.

Criar a Interface do Usuário (UI)

1. Crie um Canvas: Clique com o botão direito na Hierarquia -> UI -> Canvas.
2. Campo de Nome: No Canvas, clique com o botão direito -> UI -> TMP_InputField. Nomeie-o como NameInputField.
3. Exibição de Pontuações:
    a. Crie um GameObject vazio dentro do Canvas como container (ex: ScoresPanel).
    b. Dentro do ScoresPanel, adicione vários objetos UI -> TMP_Text para exibir as pontuações (ex: 10 deles, nomeados ScoreText_0, ScoreText_1, ... ScoreText_9). Organize-os verticalmente usando um Vertical Layout Group no ScoresPanel se desejar.
4. Botão de Enviar: No Canvas, clique com o botão direito -> UI -> Button - TextMeshPro. Nomeie-o como SubmitButton. Altere o texto do botão para "Enviar Pontuação".
5. (Opcional) Input para Número de Pontuações a Mostrar:
    a. No Canvas, adicione um UI -> TMP_InputField chamado LimitInputField. Defina seu "Content Type" para "Integer Number". Você pode adicionar um texto placeholder como "3, 5 ou 10".
6. Crie um novo script C# chamado HighscoreManager e cole o código abaixo:

```
using UnityEngine;

using UnityEngine.Networking;

using UnityEngine.UI; // Para Button comum

using TMPro; // Para TextMeshPro

using System.Collections;

using System.Collections.Generic; // Para List

// Classes para ajudar a desserializar o JSON

[System.Serializable]

public class ScoreEntry

{

    public string playerName;

    public int score;
```

```
}

[System.Serializable]

public class ScoreList

{

    public List<ScoreEntry> scores;

}



public class HighscoreManager : MonoBehaviour

{

    [Header("URLs dos Scripts PHP")]

    public string saveScoreURL = "http://seuservidor.com/api/save_score.php";

    public string getScoresURL = "http://seuservidor.com/api/get_scores.php";

    [Header("Referências da UI")]

    public TMP_InputField playerNameInput;

    public Button submitButton; // Se estiver usando o botão padrão, ou TMP_Button se preferir

    public TMP_Text feedbackText; // Um TMP_Text para mostrar mensagens como "Salvando..." ou erros

    [Header("Exibição de Pontuações")]

    public List<TMP_Text> scoreDisplayTexts; // Arraste seus TMP_Text de pontuação aqui

    public TMP_InputField scoresToShowInput; // Input para definir quantos scores mostrar

    private int currentScoresToShow = 5; // Valor padrão

    // Variável para armazenar a pontuação atual do jogador (exemplo)

    private int currentPlayerScore = 0;



    void Start()

    {

        if (submitButton != null)

        {
```

```
            // Exemplo: O botão chama SubmitCurrentScore quando clicado
            // Você precisará definir a pontuação 'currentPlayerScore' em algum lugar no seu jogo
            submitButton.onClick.AddListener(() => SubmitCurrentScore(Random.Range(100, 1000))); //
Pontuação de exemplo
        }

        if (scoresToShowInput != null)
        {
            scoresToShowInput.onEndEdit.AddListener(UpdateScoresToShow);
            // Carrega o valor inicial do input, se houver
            UpdateScoresToShow(scoresToShowInput.text);
        }
        else
        {
            FetchHighscores(); // Busca com o valor padrão se não houver input
        }
    }

    void UpdateScoresToShow(string value)
    {
        if (int.TryParse(value, out int newLimit))
        {
            currentScoresToShow = Mathf.Clamp(newLimit, 1, 10); // Limita entre 1 e 10
        }
        // Se o input estiver vazio ou inválido, mantém o valor anterior ou default
        // ou poderia resetar para um default aqui se preferir.
        scoresToShowInput.text = currentScoresToShow.ToString(); // Atualiza o campo visualmente
```

```
        FetchHighscores();
    }
    public void SetPlayerScore(int score)
    {
        currentPlayerScore = score;
    }


    // Chame esta função quando o jogador terminar o jogo e você quiser submeter a pontuação
    public void SubmitCurrentScore(int scoreToSubmit)
    {
        string playerName = playerNameInput.text;
        if (string.IsNullOrEmpty(playerName))
        {
            if(feedbackText) feedbackText.text = "Por favor, insira seu nome!";
            Debug.LogError("Player name is empty!");
            return;
        }
        StartCoroutine(SaveScoreRoutine(playerName, scoreToSubmit));
    }
    IEnumerator SaveScoreRoutine(string playerName, int score)
    {
        if(feedbackText) feedbackText.text = "Salvando pontuação...";
        WWWForm form = new WWWForm();
        form.AddField("playerName", playerName);
        form.AddField("score", score.ToString());
        using (UnityWebRequest www = UnityWebRequest.Post(saveScoreURL, form))
        {
```

```
            yield return www.SendWebRequest();
            if (www.result == UnityWebRequest.Result.ConnectionError || www.result ==
UnityWebRequest.Result.ProtocolError)
            {
                Debug.LogError("Error saving score: " + www.error);
                if(feedbackText) feedbackText.text = "Erro ao salvar: " + www.error;
            }
            else
            {
                Debug.Log("Score saved successfully! Response: " + www.downloadHandler.text);
                if(feedbackText) feedbackText.text = "Pontuação salva!";
                // Após salvar, atualize a lista de pontuações
                FetchHighscores();
            }
        }
    }

    public void FetchHighscores()
    {
        if(feedbackText) feedbackText.text = "Buscando pontuações...";
        string urlWithParams = getScoresURL + "?limit=" + currentScoresToShow;
        StartCoroutine(GetScoresRoutine(urlWithParams));
    }
    IEnumerator GetScoresRoutine(string url)
    {
        using (UnityWebRequest www = UnityWebRequest.Get(url))
        {
```

```
            yield return www.SendWebRequest();
            if (www.result == UnityWebRequest.Result.ConnectionError || www.result ==
UnityWebRequest.Result.ProtocolError)
            {
                Debug.LogError("Error fetching scores: " + www.error);
                if(feedbackText) feedbackText.text = "Erro ao buscar pontuações: " + www.error;
                 ClearScoreDisplay(); // Limpa a exibição em caso de erro
            }
            else
            {
                Debug.Log("Scores fetched successfully! Response: " + www.downloadHandler.text);
                if(feedbackText) feedbackText.text = "Pontuações carregadas!";
                try
                {
                    ScoreList scoreList = JsonUtility.FromJson<ScoreList>(www.downloadHandler.text);
                    UpdateScoreDisplay(scoreList.scores);
                }
                catch (System.Exception e)
                {
                    Debug.LogError("Error parsing scores JSON: " + e.Message);
                    if(feedbackText) feedbackText.text = "Erro ao processar pontuações.";
                    ClearScoreDisplay(); // Limpa a exibição em caso de erro de parse
                }
            }
        }
    }
```

```
    void UpdateScoreDisplay(List<ScoreEntry> scores)
    {
        for (int i = 0; i < scoreDisplayTexts.Count; i++)
        {
            if (i < scores.Count)
            {
                scoreDisplayTexts[i].text = $"{i + 1}. {scores[i].playerName} - {scores[i].score}";
                scoreDisplayTexts[i].gameObject.SetActive(true);
            }
            else
            {
                scoreDisplayTexts[i].gameObject.SetActive(false); // Esconde os TMP_Text não usados
            }
        }
    }

    void ClearScoreDisplay()
    {
        foreach (TMP_Text textField in scoreDisplayTexts)
        {
            textField.text = ""; // Ou alguma mensagem como "N/A"
            textField.gameObject.SetActive(false); // Opcional: esconder se não houver dados
        }
    }
}
```

Integrando o Script na Cena

1. Crie um GameObject vazio na sua cena (ex: "HighscoreSystem").
2. Anexe o script HighscoreManager.cs a este GameObject.
3. No Inspector do "HighscoreSystem":
    a. URLs: Insira as URLs completas dos seus scripts save_score.php e get_scores.php.
    b. Player Name Input: Arraste o NameInputField da Hierarquia para este campo.
    c. Submit Button: Arraste o SubmitButton para este campo.
    d. Feedback Text: Crie um TMP_Text na UI para feedback e arraste-o para cá.
    e. Score Display Texts: Aumente o tamanho da lista (Size) para corresponder ao número de TMP_Text que você criou para exibir as pontuações. Arraste cada ScoreText_X da Hierarquia para os elementos da lista.
    f. Scores To Show Input: Arraste o LimitInputField (se você o criou) para este campo.
4. Configurar o Botão:
    a. Selecione o SubmitButton na Hierarquia.
    b. No Inspector, no componente "Button", encontre a seção On Click ().
    c. Clique no +.
    d. Arraste o GameObject "HighscoreSystem" (que tem o script HighscoreManager) para o campo None (Object).
    e. No menu dropdown, selecione HighscoreManager -> SubmitCurrentScore(int).

    Importante: A função SubmitCurrentScore(int) no script de exemplo pega a pontuação como parâmetro. Você precisará de uma lógica no seu jogo para definir currentPlayerScore ou chamar SubmitCurrentScore com a pontuação correta quando o jogo terminar. O exemplo atual no Start() usa Random.Range apenas para demonstração. Você deve remover isso e chamar SubmitCurrentScore do seu script de gameplay.
5. Por exemplo, seu script de jogo poderia ter:

```
// No seu script de gameplay
// public HighscoreManager highscoreManager; // Arraste o HighscoreSystem aqui
// ...
```

```
// void GameOver(int finalScore) {
//    highscoreManager.SetPlayerScore(finalScore); // Define a pontuação
//    // O botão Submit fará o resto, ou você pode chamar
highscoreManager.SubmitCurrentScore(finalScore) diretamente

// }
```

Testando

1. Certifique-se de que seu servidor web com os scripts PHP e o banco de dados MySQL esteja funcionando e acessível.
2. Preencha as URLs corretas no script HighscoreManager.
3. Dê Play no Editor do Unity. Tente inserir um nome, definir o número de scores a mostrar (se implementou o campo) e clique no botão "Enviar Pontuação" (lembre-se que a pontuação no exemplo é aleatória; adapte para a pontuação real do seu jogo).
4. Verifique seu banco de dados para ver se os dados foram inseridos.
5. As pontuações devem ser carregadas e exibidas.