Unisave

Documentation

Getting started

Features

Installation

Introduction

Workflow

Core

Facets

Entities

Authentication

Modules

Overview

Matchmaker

Advanced

Serialization

Introduction

- Server logic
- Calling server methods
- Data
- Where to go next

Server logic

No matter what backend you are building, you need to have server-side logic. This logic is grouped into so-called *facets*.

Facet is the face of your backend server. It is what your game client can call remotely.

Below is a `HomeFacet` that returns data about the logged-in player when a home scene is loaded:

```csharp
using System;
using Unisave;
using Unisave.Facades;

public class HomeFacet : Facet
{
    /// <summary>
    /// Returns information about the logged-in player
    /// </summary>
    public PlayerEntity GetPlayerEntity()
    {
        // obtain authenticated player ID from the session
        // and load player data from the database
        PlayerEntity entity = Auth.GetPlayer<PlayerEntity>();

        // send the data back to the game client
        return entity;
    }
}
```

Calling server methods

Of course, there needs to be some `MonoBehaviour` in the home scene that calls the facet method:

```
using System;
using Unisave;
using UnityEngine;

public class HomeSceneController : MonoBehaviour
{
    async void Start()
    {
        PlayerEntity player = await OnFacet<HomeFacet>
            .Call<PlayerEntity>(
                nameof(HomeFacet.GetPlayerEntity)
            );

        Debug.Log("Player: " + player.Nickname);
        Debug.Log("Coins: " + player.Coins);
    }
}
```

Data

An *entity* is a collection of data that can be stored in a database.

The `PlayerEntity` seen above is defined as follows:

```
using System;
using Unisave;

public class PlayerEntity : Entity
{
    /// <summary>
    /// Name displayed to other players
    /// </summary>
    public string Nickname { get; set; }

    /// <summary>
    /// Number of coins owned
    /// </summary>
    public int Coins { get; set; }
}
```

Entities can be created, saved, modified, and deleted by the server code. They can also be sent to the client and read.

Where to go next

Now you need to know, where to put this code, how to get it onto the server and how to run it. This is described in the next section on workflow.