photon

Search

This document is about: **QUANTUM 2**

SWITCH TO

Events vs Polling

Introduction

Reading information from the simulation while inside Unity in order to display relevant data to the user is a common practice in Quantum games.

To do this, there are two approaches that can be taken. The first approach is `polling`, which involves requesting information from Unity at regular intervals, such as inside the `Update` loop. The second approach is using Quantum events, which involves subscribing methods to Quantum's event system and using them to update the view.

Polling

A simple approach to polling could look like this:

C#

```
using Quantum;
using UnityEngine;

public class CharacterAnimations : QuantumCallbacks
{
  private EntityView _entityView;
  private Animator _animator;

  private void Awake()
```

```
    _animator = GetComponent<Animator>();
  }

  public override void OnUpdateView(QuantumGame game)
  {
    var frame = game.Frames.Predicted;
    var kcc = frame.Get<CharacterController3D>(_entityView.Entity
    _animator.SetFloat("Speed", kcc.Velocity.Magnitude.AsFloat);
  }
}
```

This snippet implements a callback for when the view is updated, which happens after the frames simulation has finished, then uses the `QuantumGame` from the callback itself to get a frame, from which the game state can be polled and applied in the view (in the snippet, specifically to animation).

Make sure to only use the Frame API for **ready only** operations, as writing to it from Unity would be non-deterministic.

Event Based

A simple approach to event based could look like this:

C#

```
private void Start()
{
    // subscribe to the simulation event
   QuantumEvent.Subscribe<EventOnDamaged>(this, OnDamaged);
}

private void OnDamaged(EventOnDamaged e)
{
    // play a particle effect to show a damage indication
  GetComponent<ParticleSystem>().Play();
}
```

Unity. It is even possible to send game data within the event class, which already contains a reference to the game from which the frame can be found.

For more information about events, visit Events & Callbacks.

Pros & Cons

Both of these methods have their own drawbacks.

Event based code can be more performant when the information doesn't need to be too frequently sent to Unity. Game situations that don't happen every frame and that have an impact on the game view are usually better represented by events. But if such game data needs to be sent every tick, event will likely perform worse than polling. However, Quantum events are ***fire-and-forget***, meaning that late joiners will not receive events which were executed before they joined. So anything visual that was created from events will need to be manually re-created when the client joins.

Polling code is generally simpler to write and easier to understand, and is usually better used to represent visual data that can change very frequently (every frame). Due to the nature of polling, initializing visuals for late joiners is automatic as it guarantees all the data to build the visuals are already contained in the game state that was received by the late joiner.

Predicted Vs Verified

While using both of these techniques, you have the option of using either `Predicted` or `Verified` frames. Both have their own drawbacks and benefits. While predicted frames will give you more immediate feedback, they may be susceptible to being inaccurate due to rollbacks. Whereas verified frames are concrete, but they will take a moment to take effect as it includes a round trip to the server to verify them. Generally a developer would use a mix of these, for example, they could use verified frames to display game score and predicted frames for effects such as jump clouds or hit particles.

Back to top

photon

photon

We Make Multiplayer Simple

Products

Fusion
Quantum
Realtime
Chat
Voice
PUN

Memberships

Gaming Circle
Industries Circle

Support

Gaming Circle
Industries Circle
Circle Discord
Circle Stack Overflow

Connect

Public Discord
YouTube
Facebook
Twitter
Blog
Contact Us

Documentation

Fusion
Quantum
Realtime
Chat
Voice
PUN
Bolt
Server
VR | AR | MR

Resources

Dashboard
Samples
SDK Downloads
Cloud Status

Languages

English
日本語
한국어
简体中文
繁体中文