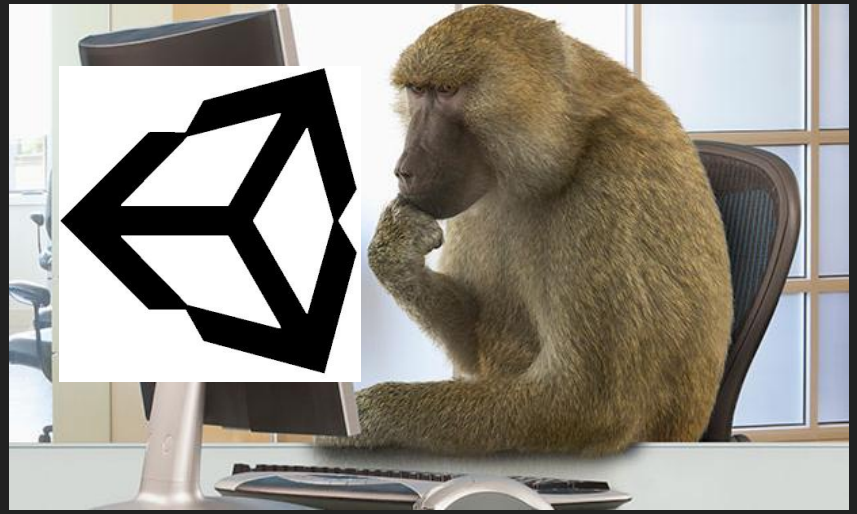
Unity & VR Best Practices

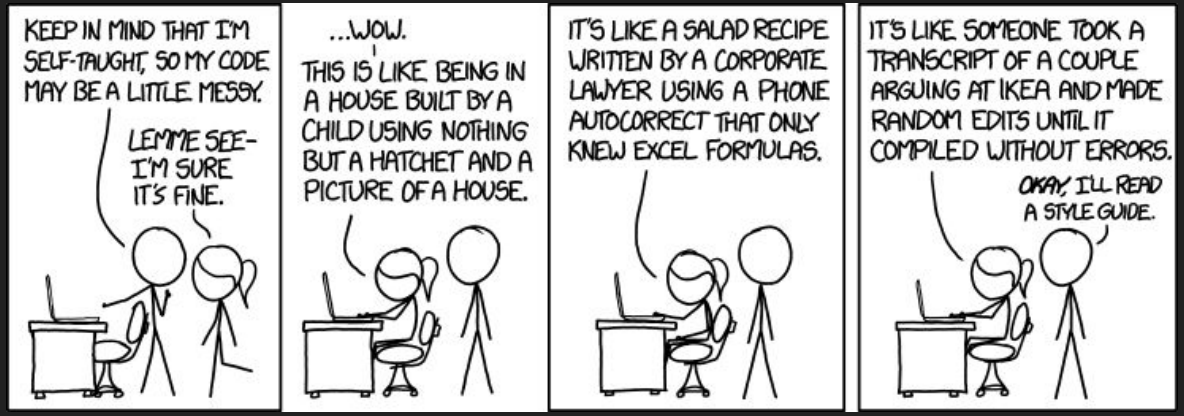


A long-winded discussion-lecture where we talk a lot and maybe someone learns something but probably not.

~Adapted From Slides by Victor Mouschovias~

Why are we here?

- Unity encourages *really friggin' awful* solutions to problems.
- People are super good at making super bad VR.
- Nobody reads the Oculus Best Practices Guide :'(



Friggin' Awful Solutions

Steve is a Minecraft character. He is making a 3D puzzle game in Unity. His game has one scene with a single puzzle. When players solve the puzzle, Steve wants to play some special effects baked into the environment (stars twinkle, confetti cannons erupt, etc.).

Steve has a ***PuzzleScript*** that handles all puzzle logic, including checking for the victory condition.

Steve is a computer science student, but he's already ***paid an art student*** to make all of his effects. How can Steve accomplish his goal?

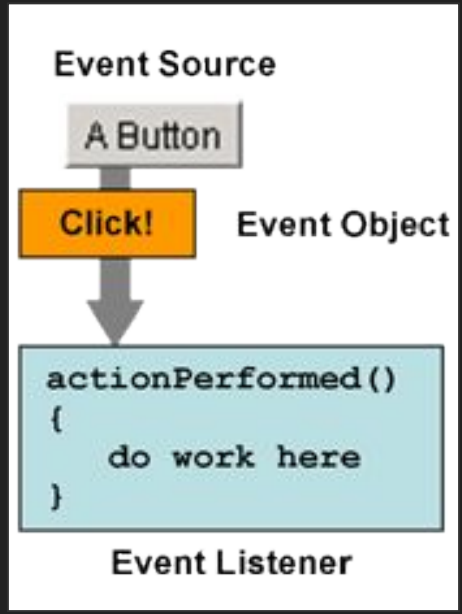
Friggin' Awful Solutions

Steve is still a Minecraft character. Now he wants to add 10 more scenes, each with the same puzzle and ***PuzzleScript***. Each scene requires its own, unique victory effects.

How can Steve trigger each of these effects from one universal script?!?

Our Hero: Event-Based Programming

- A programming paradigm where functions are called by *event managers* in response to events.
- Very common in mobile & web development
- Unity has some “rigid” events
  - OnCollisionEnter
  - Start
  - Update



We need a more Flexible Structure!

What we want:

What we have:

```
delegate int SomeDelegate(int x);
```

Delegates

- “Points” to a certain type of function.
- Can be assigned like any variable.
- Can “remotely” call a function.

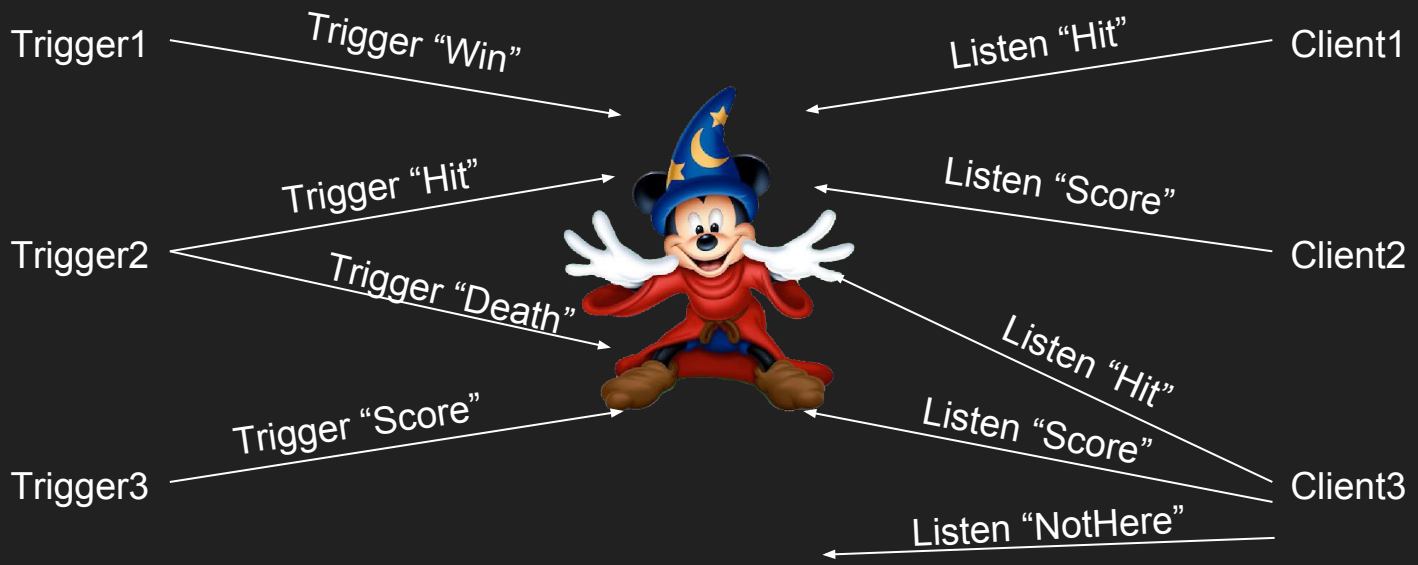
```
delegate void MyDelegate ();

void iWantToBeCalled() {
    // Awesome stuff.
}

MyDelegate deleg = new MyDelegate(iWantToBeCalled);
deleg();
```

The Callback Machine

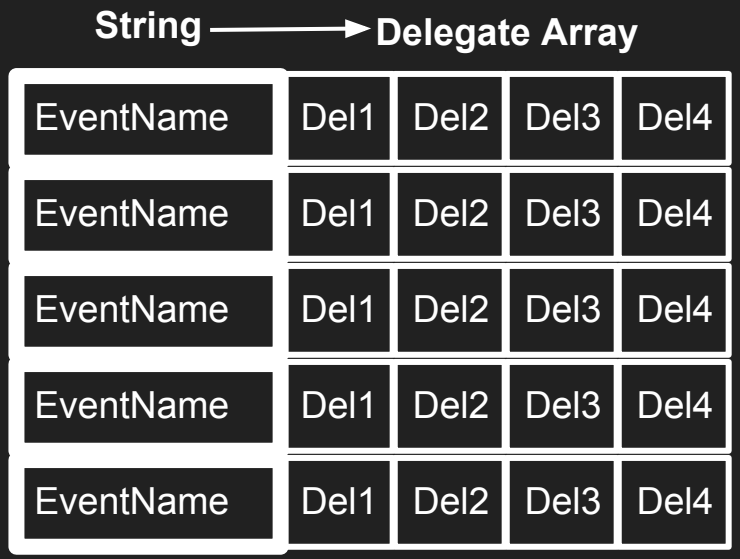
- We can do better than storing a single function to callback...



The Callback Machine

registerListener

- Args:
  - EventName (String)
  - Callback (Delegate)
- Place delegate in the dictionary under “EventName”



triggerEvent

- Arg:
  - EventName (String)
- Iterate through “EventName” entry and call each delegate.

The Callback Machine

How can we improve it?

- How could we add global visibility?
- How could we add argument support?

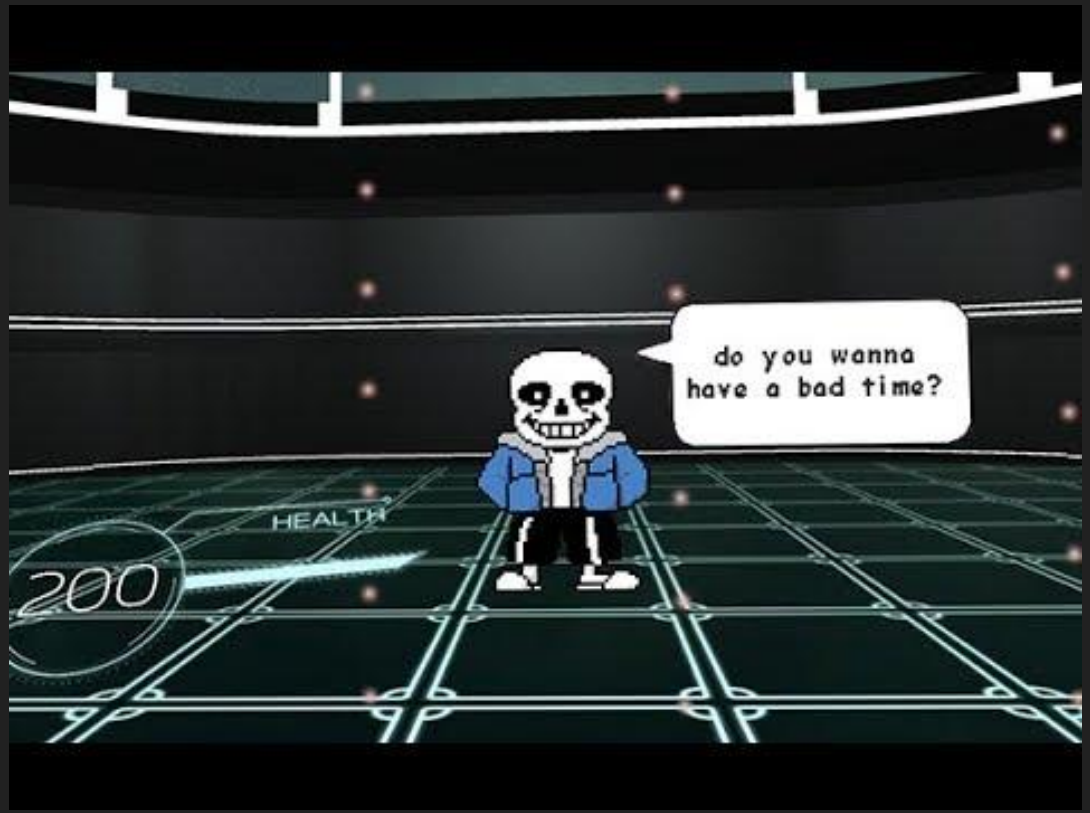
Alternatives

- Unity Events
- https://docs.unity3d.com/ScriptReference/Events.UnityEvent.html
- https://docs.unity3d.com/Manual/UnityEvents.html
- Tutorials
  - https://unity3d.com/learn/tutorials/topics/scripting/events
  - https://unity3d.com/learn/tutorials/topics/scripting/events-creating-simple-messaging-system
- Our solution works quite well, however
- CSharpMessenger Extended
- https://wiki.unity3d.com/index.php?title=CSharpMessenger_Extended

That’s not all, Folks!

We’ve just solved a surprising amount of problems…

- Non-Blocking control flow
- Too many Singletons
- Need for Global Data
- Messy code



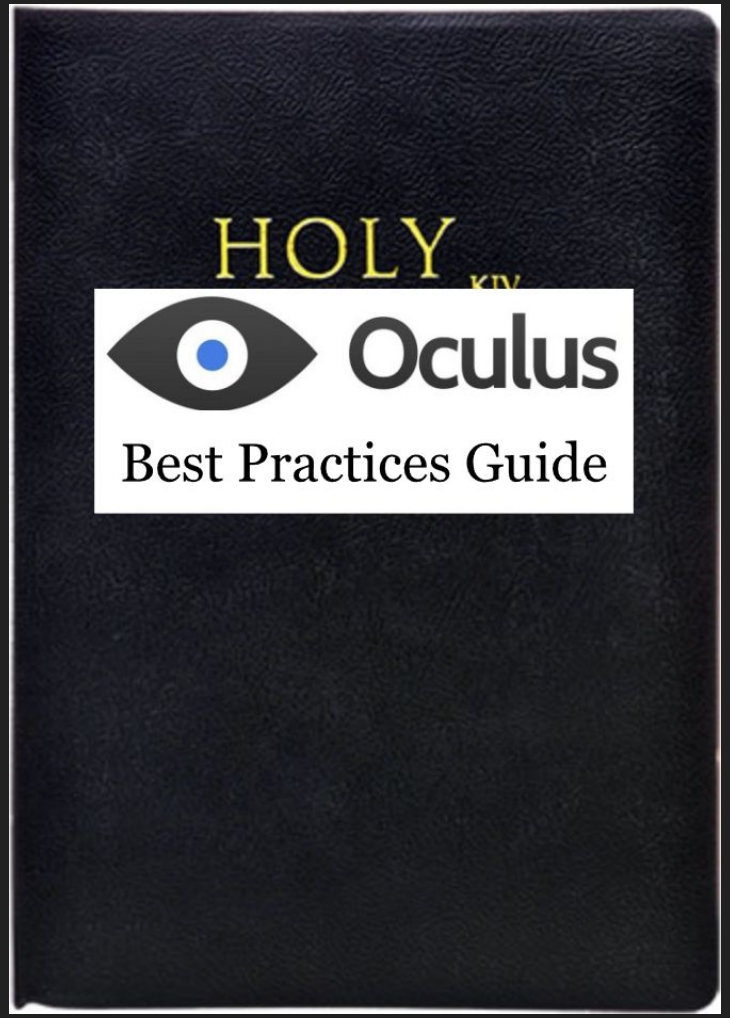
Other Dangers

- “Fluid” Component Structures
  - Strict class hierarchies have their benefits.
- Loading resources as needed, dynamic resolutions
  - Resources.Load, GameObject.Find, etc.
- Using basic Unity for *everything*
  - Right tool for the right job.
- Networking

Let’s Talk “Bad” VR

Let’s Talk “bad” VR

- High Production Value != Good Design
  - Testing is expensive!
  - Testers usually don’t identify issues directly.
- We need to make better designs...


HOLY KJV
Oculus
Best Practices Guide

Minimizing Latency

- FPS can’t drop *no matter what*.
- Games with sandbox elements might have some issues...

Accelerations

- Vection, vestibular system, vestibulo-ocular mismatch
- Easy to forget about:
  - Rotations
  - Teleportation effects
  - Preparing users for motion
    - More on this soon

More Accelerations

Accelerations, Field of View

- The less they see, the less they feel!
- ***Very, very, very*** useful to provide constant frame of reference.
- Users may have to move their head more, so watch out.

Accelerations, Movement

- Movement in-line with the viewing direction is optimal.
- Preparing the body for movement goes a long way...

Third-Person Cameras

- Subject to the same accelerations issues as first-person.
    - Camera swings!
- We lose some Field of View control.
- We can decouple camera and avatar movement!
    - Flight sims can benefit from this!

User Interface

- Part of the 3D world.
    - **NOT RIGIDLY ATTACHED TO USER'S HEAD! >:(**
- Sits 2-3 meters in front of eyes.
- Doesn't require eye-swivels.
    - Put UI in middle ⅓ of viewing area.
    - Or allow head movements to examine UI features.

Other Stuff

- Sound cues
- Content
  - Don’t rely on stereoscopic vision
- Altitude
  - “*Visual flow*” of pixels
- Gorilla arms
  - Small gestures > big

Questions? How are your projects going?


FRIDAY'S