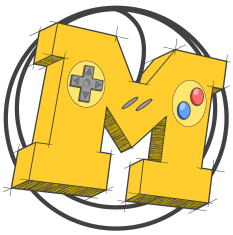

Lecture 5

Announcements

- Photos from Ohayocon 2023.
- On the occasion of Anthem's sunsetting, Mark Darrah explains the epic disaster from his perspective.
- Pet fish commits credit card fraud using Nintendo Switch.
- Recall our composition lecture from earlier. Components are meant to be flexible and re-usable, which is hard to do if you write 5,000-line components! (taken from Yandere Simulator). Keep your components a wee bit smaller than this.
- Q: How is p1_alpha going? Can you feel any momentum building?
    - It takes around two weeks to build muscle after achieving fatigue.
    - Please also note– you do not need to implement everything on your task sheet. Time growing short? Cut a task or two with no regrets.
    - Remember– we don't care about polish on tasks until p1_gold deadline.

System Dev : Toast

The exercise of implementing a simple "Toast" system into Flappy Bird will take us through Unity's powerful-but-quirky UI system, in addition to powerful design patterns such as Singleton and Observer.

- Intro Slides
- View finished version here
    - commercial use case example from Greek Tragedy (achievement system)
- Starter Project / Finished Project
- EECS 494 Repository : Flappy Bird Systems Design Exhibition (Toast)

Unity's UI system takes much of the pain out of building simple menus and gamestate-feedback mechanisms...once you understand its quirks!

- Toast System Implementation (Singleton Pattern)
- Example (Juicy-UI / Observer Pattern Flappy Bird)
- Rect Transforms vs Normal Transforms
- The CanvasScaler Component
    - Don't let your build be foiled by resolution-related UI-scaling bugs!

- AnchoredPosition

Delegates, Actions, Callbacks, and the Observer Pattern

- (demonstrate within flappy bird toast system project)
- One of the core tenants of engineering is the idea of modularity-- complex systems should be broken down into smaller ones (“modules”) with simpler objectives-- objectives that may be thoroughly tested, verified, and trusted (doing so simplifies the mental model necessary to understand the larger system). None of us need to think about the mono runtime (the technology that powers our games) in this course-- it is axiomatic to us that it will work perfectly, despite its immense complexity. This simplifies our own mental model of our games, making it p
- Possible for us to continue building upwards.
- Delegates, Actions, and Callbacks all find substantial use within the “Observer” design pattern-- a powerful tool for system design that allows our modules to remain cleanly isolated from one another (and therefore easy to maintain, test, and expand).
- Improving Flappy Bird (See above for download).

Unity Tip : Obtaining Prefab References

- (starter project) (finished example project)
- An alternative to “dragging and dropping references” (which sometimes isn’t possible, such as in static functions).
- Step 1 : Create a “Resources” folder anywhere.
- Step 2 : Move your prefabs to it.
- Step 3 : Obtain a reference to that prefab via…
    - GameObject prefab_ref = Resources.Load<GameObject>(<prefab_name>);