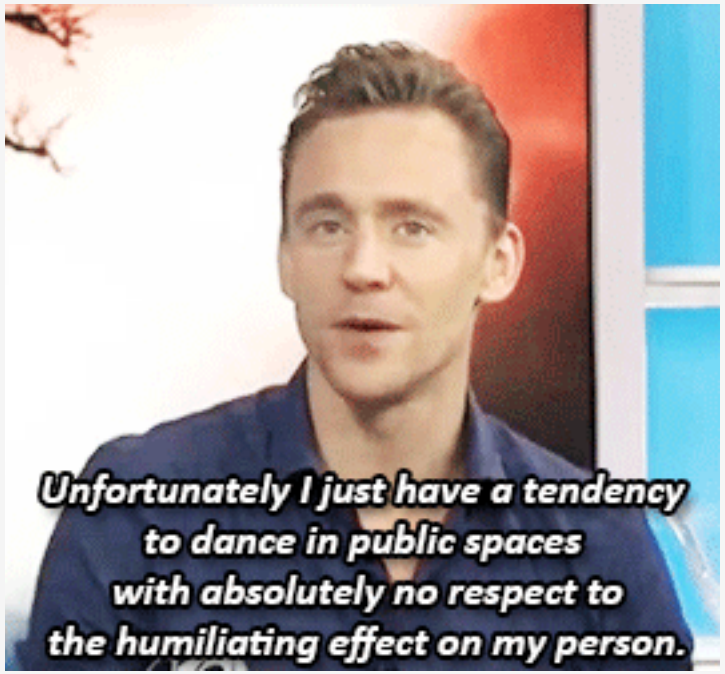
ARCHITECTING FOR LIVE SERVICE GAMES ON MOBILE

Anna Tito

#GameDevArchitecture #nzgdc18 @MythicalC

Language.

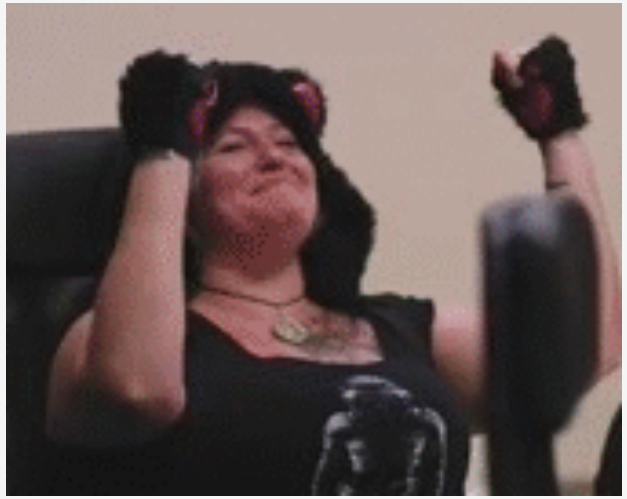
Who am I?

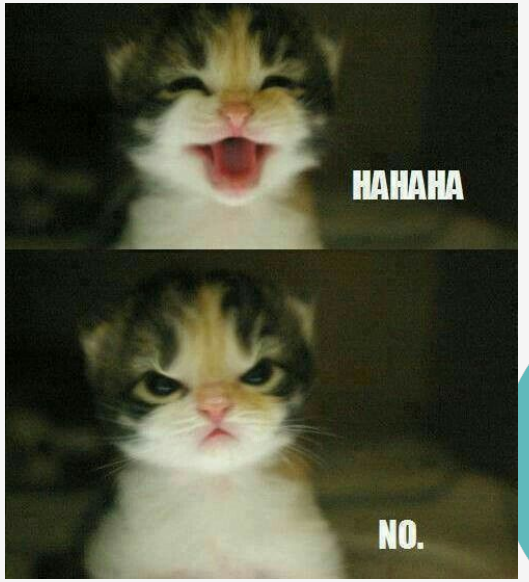


Technical Project Lead, Hipster Whale
(unannounced project)

Published Titles

Star Wars: Galaxy of Heroes (EA)
The Simpsons: Tapped Out (EA)
Ice Age: Village (Gameloft)
VEGA Conflict (KIXEYE)

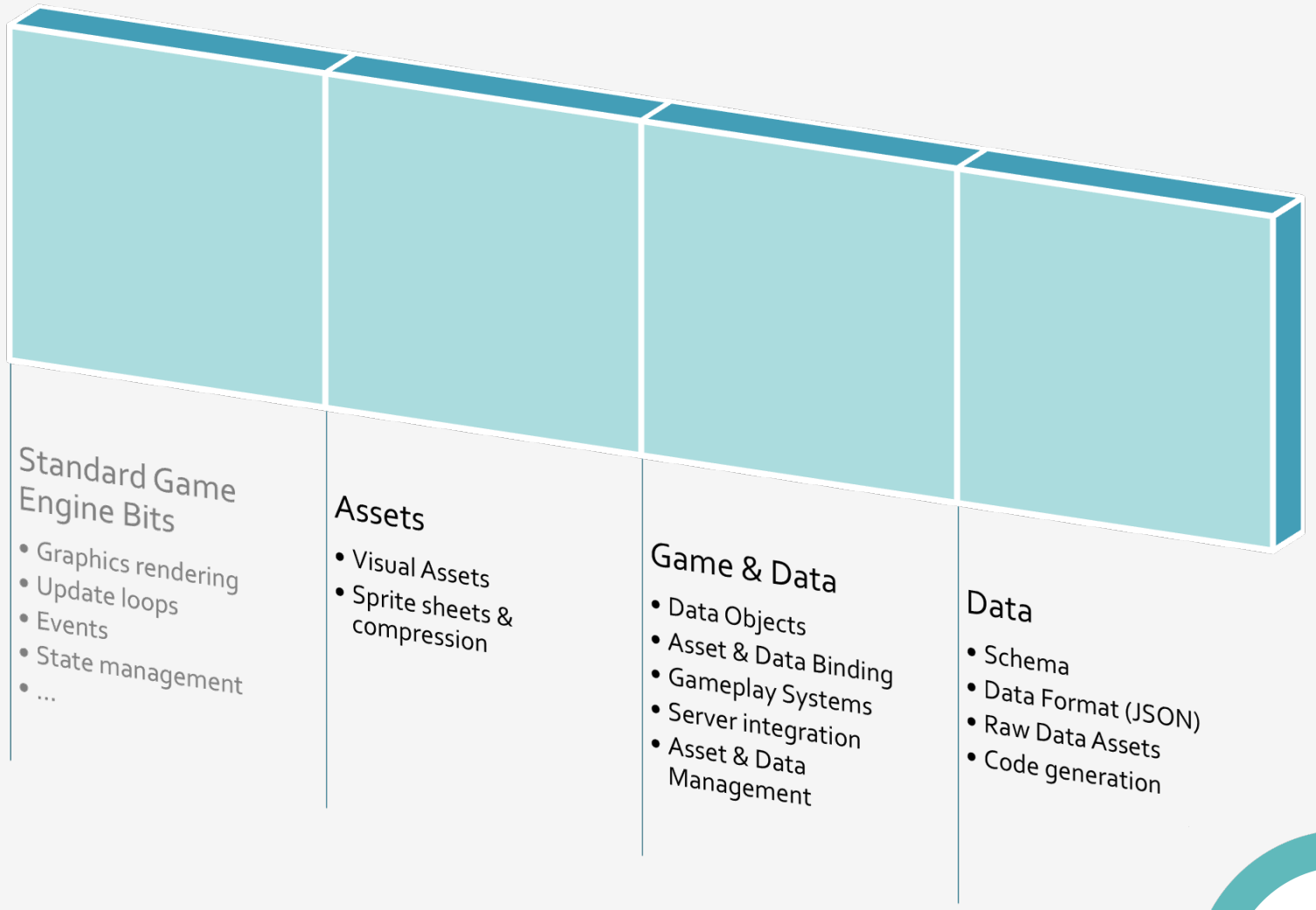




I do not have THE answers.

Before we begin, you must all be warned...

Object-Oriented Programming & Data-Driven Design



*Context is **everything***

- Features are units of functionality - It shouldn't matter to the code how those units are mixed and matched.
- Context controls what units are active when and how it they behave & interact.

Now, let us begin.

Why?

A successful games as a service needs

1. To be on their device (downloaded and not deleted)
2. To be used regularly
3. To be updated regularly
4. Lower support and production costs
5. High retention and conversion rates

Respect

(players & your team)

Respect Your Players

- Their time
- Their resources

Respect Your Team

- Their time
- Their skills
- Their humanity

To be on their device

(downloaded and not deleted)

- **App size**
- **Initial launch time**
- **On Device storage**

App Size

"..exceeding the OTA limit can reduce your downloads by up to 50%, and impact first session after install numbers just as much."

https://api.playfab.com/blog/size-matters-how-file-size-impacts-installs-mobile-games

"For every 6 MB increase to an APK's size, we see a decrease in the install conversion rate of 1%."

https://medium.com/@avijeet.dutta13/ios-app-size-analysis-76206a18a666

"From these results we estimate a linear decrease in install conversion rate below the cellular download limit from 3–99MB at 0.45% per MB. Above the cellular download limit we estimate a linear decrease in install rate of 0.32% per MB."

https://segment.com/blog/mobile-app-size-effect-on-downloads/

Hide it behind an initial loading screen….

Contextual Assets

(download as and when you need them)

How are your assets used/accessed?

- Bundle by usage
- Remove assets as they are no longer needed
- Clearly message to user when downloading

Integrate asset management with your data system

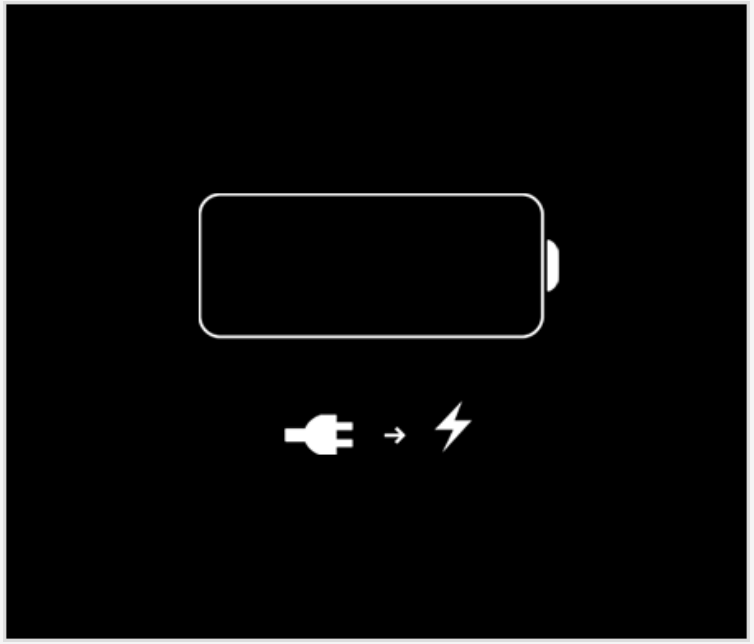
- Data context can provide the usage pattern.
- Allow the data determine if an asset is required.
- Split data and visual assets.
- More targeted bundles ideally mean less patching across multiple bundles and subsequently smaller updates.

To be used regularly

Device resource usage matters

- Battery use
- Data

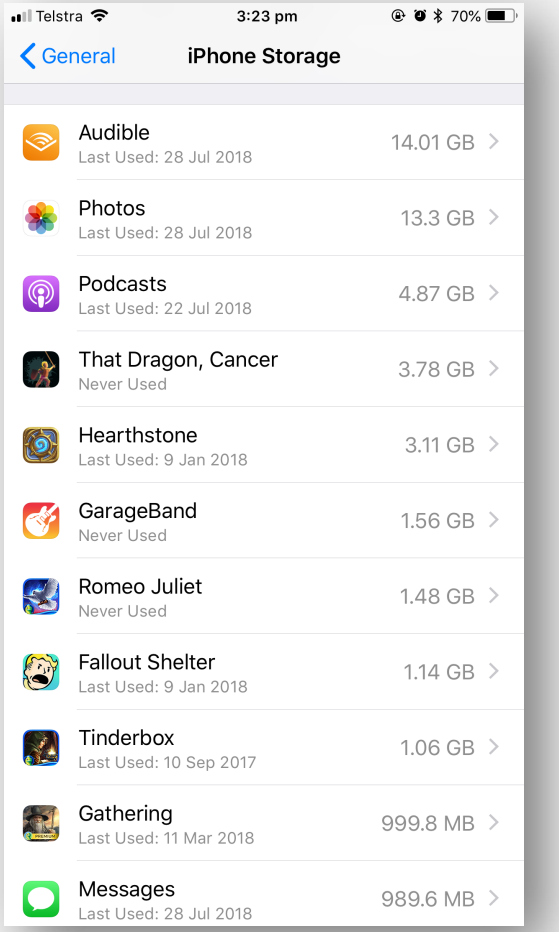
Battery use



You don't want to be the top user

- **Have configurable options to help keep usage low**
  - Dynamic frame rates
  - Brightness
  - GPS use
  - Graphics – post processing effects etc
  - Cellular data use
  - Reading and writing from persistent storage
- **Power save mode**

Data



Data Storage

- Keep only what you need
- Contextualized data allows you to remove unneeded assets.

Cellular Data

- Be respectful of their data – in many countries data limits are still quite low.
- Keep save back ups small - send only modified data
- Verify changes against the server

Regular high quality updates

- **Data driven functionality**
- **Remote configuration**
- **Kill switches**
- **Keep update sizes small**

Data driven functionality

Rules & Actions

- Rule
  - Evaluate
  - Positive Actions
  - Negative Actions
- Requires a system to evaluate against usually some form of persistent variables and/or world state
- Can use your existing data system to define rules and actions e.g. JSON, XML

Full scripting framework

  - Use an existing e.g. Lua, python etc
  - Roll your own

Kill switches & remote configuration

- **Remote Configuration**
  - rewards
  - Non premium costs e.g. resource
  - XP levels
  - Events
  - Promotions
  - transaction reasons etc.
- **Kill switches**
  - New features
  - Events
  - Promotions

Kill switch any new system that could block player engagement with the game

Keep update sizes small

- **Contextual asset bundles**
- **Split data and visual assets**
  - Visual assets are your largest consumer of resources, you may want to split them further by sub context.
- **Configurable values and flags for events**

Respecting your team

Lower support and production costs

- **Tools**
  - Manual data entry costs time
  - Simple bugs cost time e.g. missing ‘<‘ extra ‘,’
  - Auto- complete reduces preventable errors like typos
- **Automated testing & data validation**
  - The data structure can tell you what is valid format
  - The data can provide what is valid data
  - Trace the data links and validate structure
- **Reduce server costs – Reads & Writes**
  - Data context can provide the usage pattern allowing you to only download what you need rather than all the bundles or assets.
  - By sending only modified data you, cache the changes and only write when the session is complete.

You don't know, what you don't know

ANALYTICS

Questions?

Anna Tito

@MythicalC