# BludBourne: A Retrospective

by Patrick Hoey

# What is BludBourne?

- *BludBourne* is a 2D Roleplaying Game (RPG) that I developed for my book *Mastering LibGDX Game Development* which was published in 2015 by Packt
- To date, over 1000 books have been sold with positive online reviews, and I have received scores of emails from fans
- BludBourne contains the following features:
  - Stats including strength, intelligence, hit points, magic points, and experience attributes with a level cap of 10
  - Player movement, animation, and collision detection
  - NPC interaction with dialog trees
  - Enemy NPCs with spawn points, and battle mechanics
  - A portal system for travelling around the world and transitioning from a town to an enemy castle
  - An inventory management system for item pickup and equip and a HUD for status updates
  - Save and load game profiles
  - Scripted cutscenes
  - A quest system with objectives and rewards
  - A shop UI where you can interact with NPC merchants to view, select, purchase, and sell items
  - Special effects that include a shake camera, static lighting, day to night cycle, particle effects, music and sound effects

# Gathering Requirements

- Packt wanted three things:
  - A finished technical book
  - with an RPG Game
  - using LibGDX

# Outline Submission

# Roadmap of Deliverables

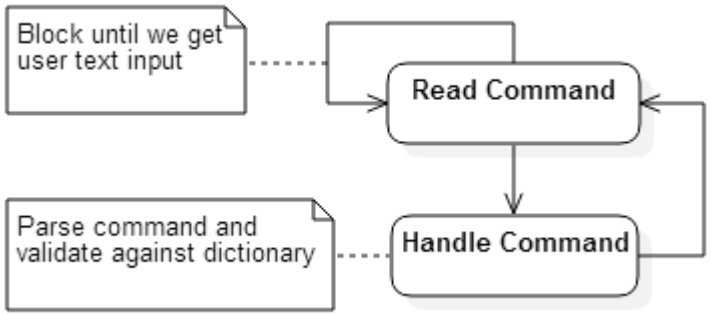
- Initial page count was projected to be 250 pages for a 10 chapter book
- I was assigned a different editor for each phase during production
  - First phase: 5-6 months for all rough drafts
  - Second phase: 1-2 months for final drafts
  - Third phase: 2-3 months finishing for publication

# My Plan

- Agile approach using Scrum methodologies
  - 2-3 week sprints
    - 1-2 weeks developing the game with features for the specific chapter
      - Includes time for fixing bugs discovered by technical reviewers
    - 4-7 days writing the chapter including creating all diagrams and images
- Schedule backup of laptop everyday to external HDD as well as cloud drive
- Stay disciplined with proper Git commit messages
  - Think “Past me is helping future me” not “Fixed stuff”
- With these tight deadlines, I followed these guiding principles:
  - “Don’t Repeat Yourself”
  - “Keep it Simple Stupid”
  - “Make it Work, Make it Right, Make it Fast”

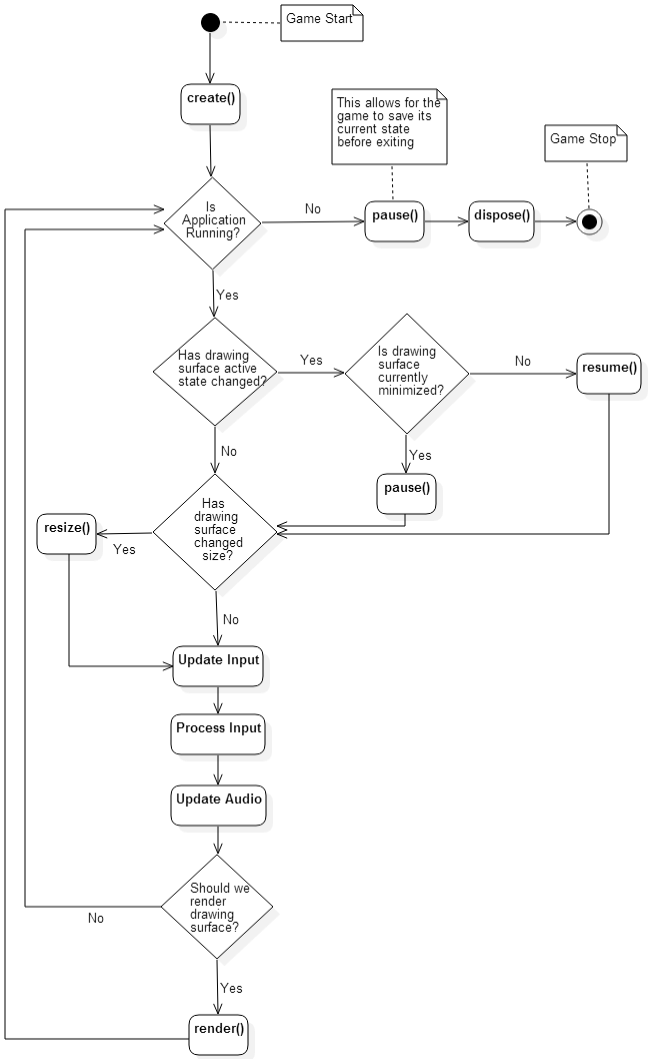
# Starting Development: Understanding the Game Loop

- A basic game loop



# Starting Development: Understanding the Game Loop

- Game loop for a 2D graphics based RPG

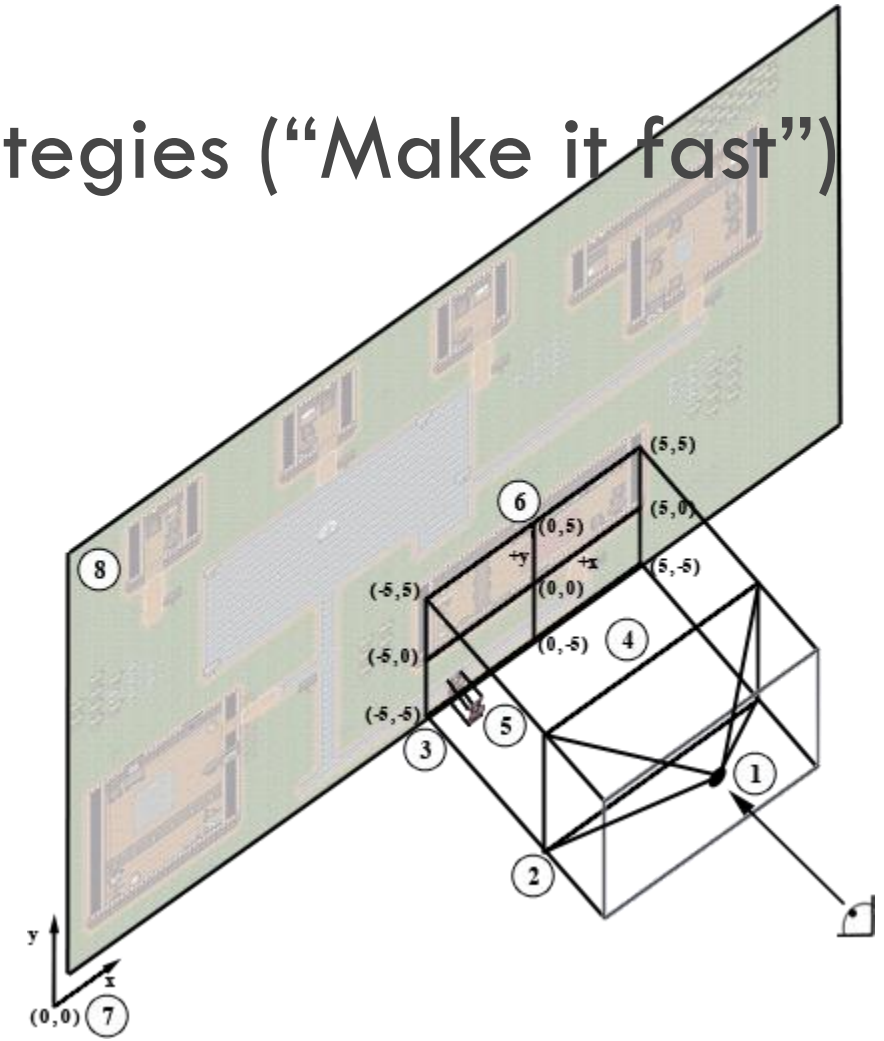


# Performance Considerations

- Sometimes, “Make it fast” is a luxury, especially with looming deadlines
- There are always tradeoffs between performance and design
- By giving myself up front restrictions in favor of performance, I allowed myself to just focus on “Make it Work, Make it Right”

# Optimization Strategies ("Make it fast")

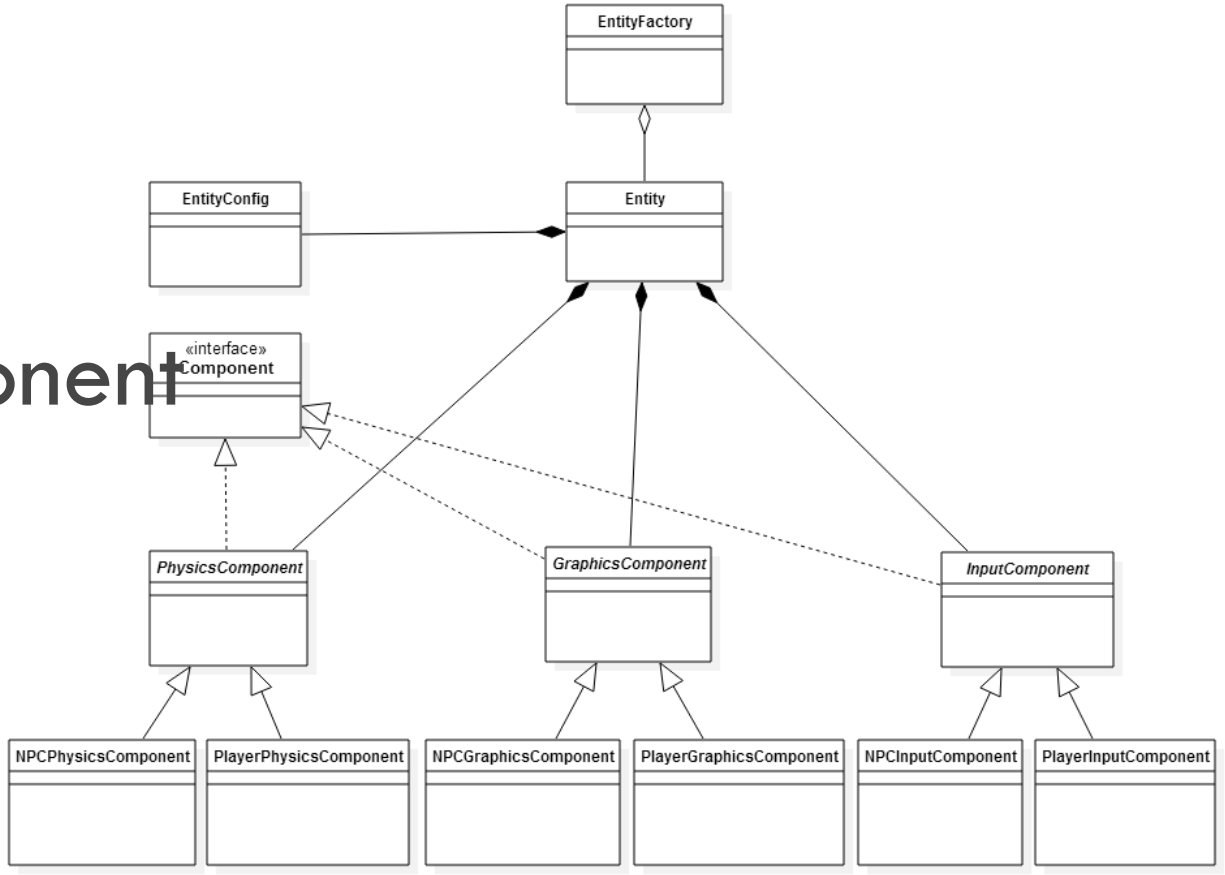
- Double Buffering
- Frustum Culling
- Data Locality
- Object Pools
- Update Pattern
- Define Art Budget



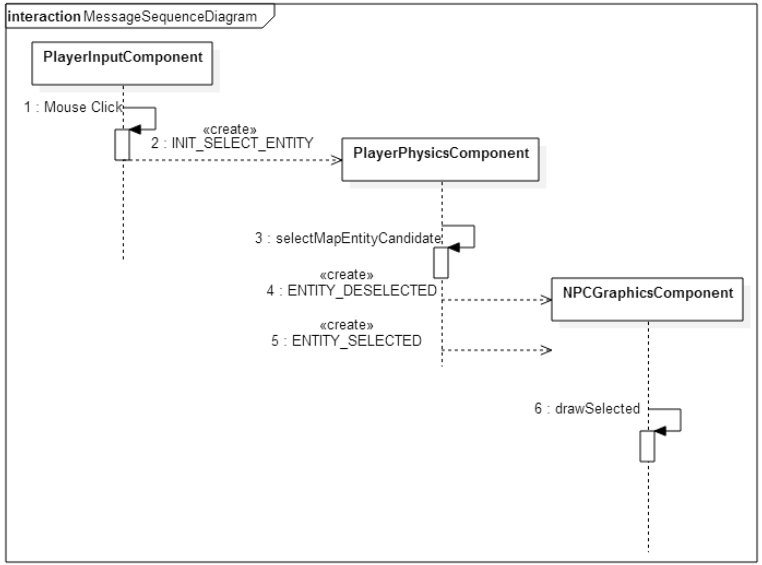
# Key Design Decisions

- Entity Component System Pattern
- Observer Pattern
- Conversation Graph
- Inventory UI

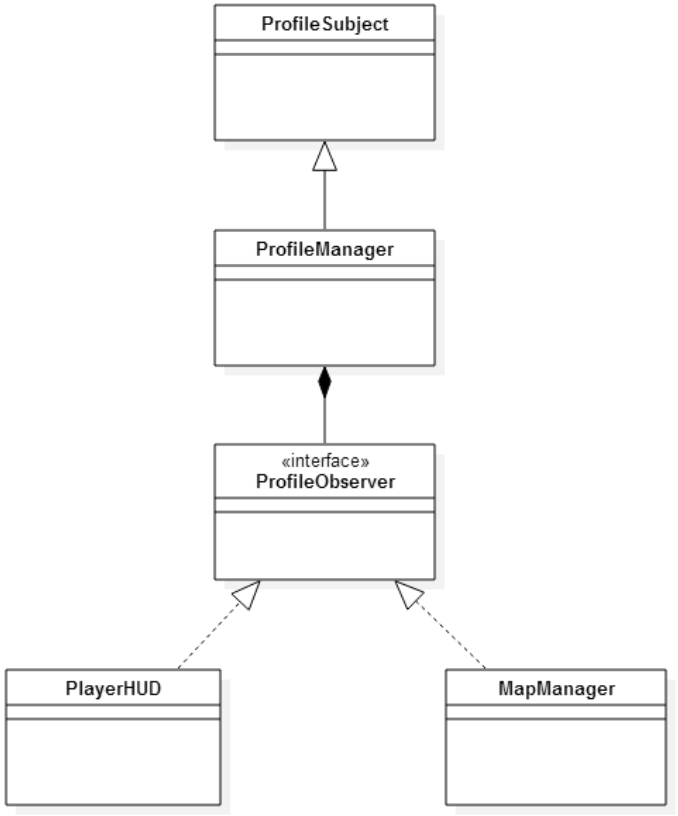
# Entity Component System Pattern



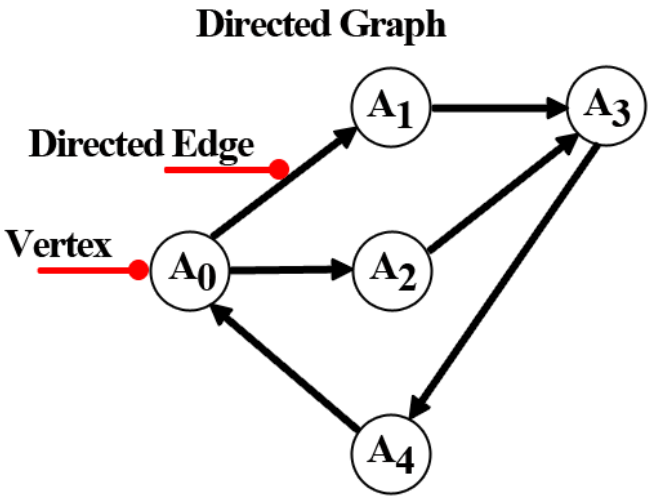
# Entity Component System Pattern (Message Bus)

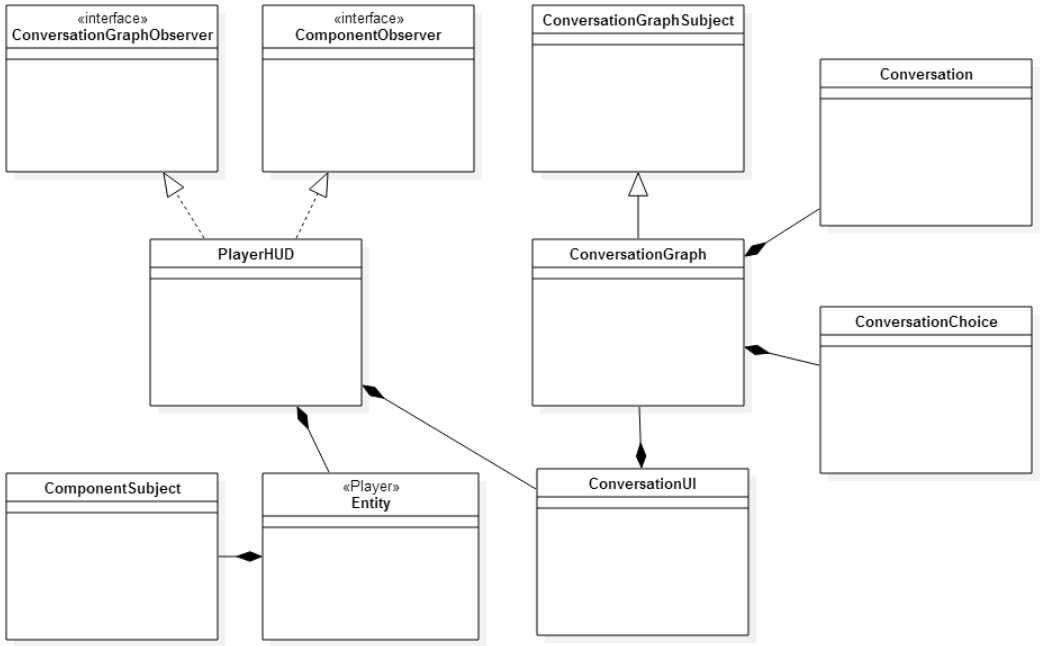


# Observer Pattern

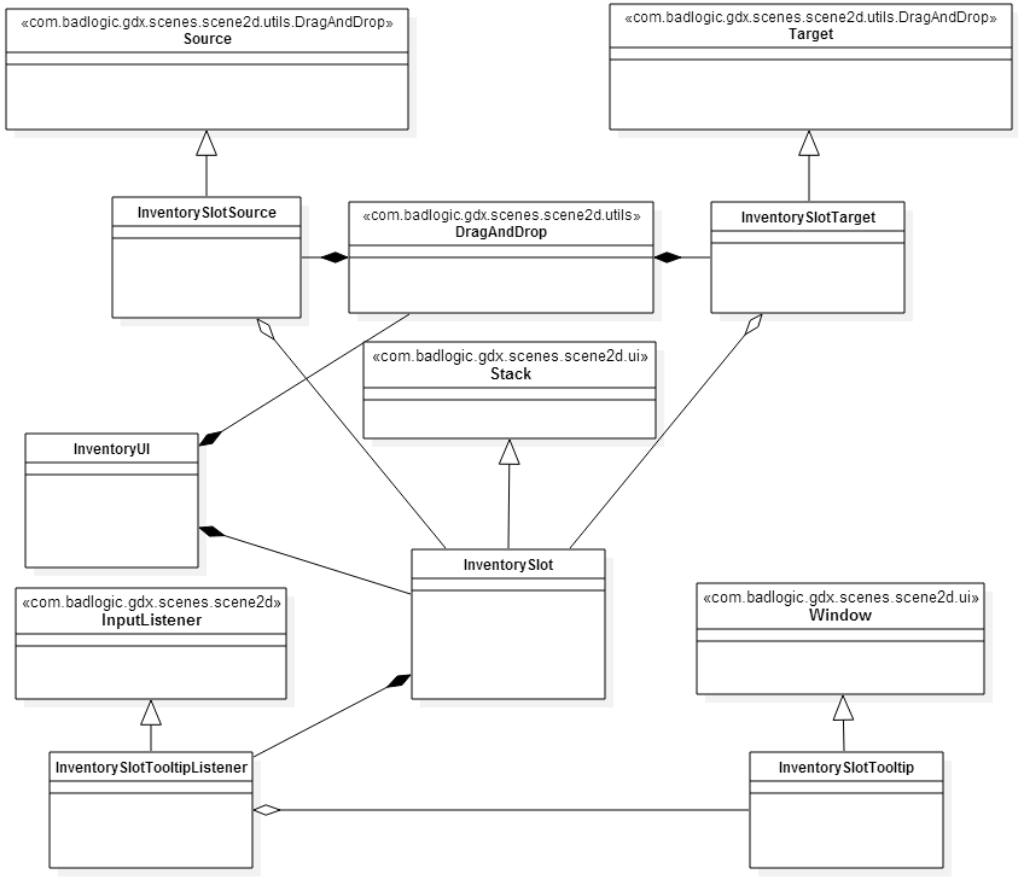


# Conversation Graph



# Inventory UI




| InventoryItem |
| --- |
| -itemAttributes: int<br>-itemUseType: int<br>-itemShortDescription: String |
| «constructor»+InventoryItem(textureRegion: TextureRegion, itemAttributes: int, itemTypeID: ItemTypeID, itemUseType: int)<br>«constructor»+InventoryItem()<br>«constructor»+InventoryItem(inventoryItem: InventoryItem)<br>+getItemTypeID(): ItemTypeID<br>+setItemTypeID(itemTypeID: ItemTypeID): void<br>+getItemAttributes(): int<br>+setItemAttributes(itemAttributes: int): void<br>+getItemUseType(): int<br>+setItemUseType(itemUseType: int): void<br>+getItemShortDescription(): String<br>+setItemShortDescription(itemShortDescription: String): void<br>+isStackable(): boolean<br>+isSameItemType(candidateInventoryItem: InventoryItem): boolean |

# Nice to Haves

- Book
  - Chapter on Networking (Client/Server)
  - Chapter on 3D objects
- Project Development
  - Support for mobile devices
    - I did discuss trade-offs
  - Quest Editor
  - Cutscene Editor
  - Unit Tests
  - Continuous Integration
  - User Testing (Minimal with Tech Reviewers)

# Final Result

- 9 months of development
  - 7 days a week
  - 12 hours a day
- A page count of 387
- 121 images and diagrams (including front and back covers)
- 182 art assets (images, maps, sprites, music, and sound)
- A final project with 82 Java classes

# Video of Edited Gameplay Footage

# References


- https://github.com/patrickhoey/BludBourne
- http://patrickhoey.com/blog/portfolio-items/bludbourne/
- Hallford, Neal, and Jana Hallford. Swords & Circuitry: A Designer's Guide to Computer Role Playing Games. Roseville, CA: Prima Tech, 2001. Print.
- "Game Programming Patterns Paperback – November 2, 2014." Game Programming Patterns: Robert Nystrom: 9780990582908: Amazon.com: Books. N.p., n.d. Web. 04 Apr. 2015

# The End