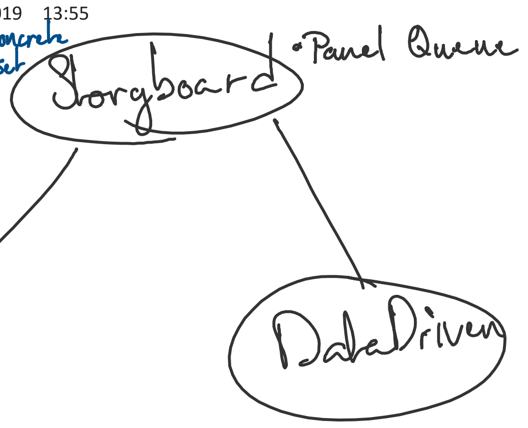
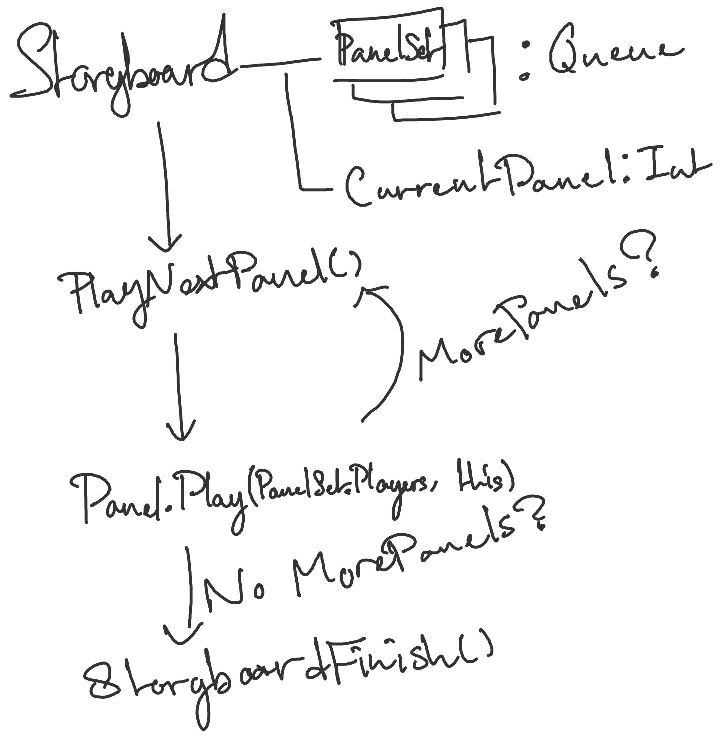
# Data Driven Storyboard

Monday, February 18, 2019 13:55

Storyboard is a set of concrete controls, while a panel set is a purely data based grouping



- Pre-configured panels
- Tree based arrangement
- Storyboard traverses tree



Can Storyboard and Panel Sets be merged?
- Data Parsing and queuing could be interfaced by the main storyboard queue Singleton

- Queue Singleton could manage a single, reusable Storyboard instance. Optionally, an override storyboard could be passed in as well.

? Something like
StoryboardQueue.Enqueue(data) //Overloaded
- Parse data into Storyboard Data Dictionary
- Initialize DataDrivenBoard with "Intro" Data
- Play and let be
- Kind of heavy (but not really)
- How will StoryboardData be managed?
- How do generated storyboard's tell the queuemanager to use the loaded data with minimal invalidation risk or an unwieldy interface?

Should all controls be setup beforehand to callback?
- Probably

- Data could be loaded continuously until the encounter ends. Seems kind of wasteful, but is flexible.

Panel Logic
```
> On First Panel
  > Parse data
  > Add intro to panel set queue
  > Return next panel
> On next panel
  > If current set in queue has no more panel info
    > Dequeue
    > if current in queue is null
      > return null
  > Get next panel info from current set in queue
  > Initialize panel game object from panel info
  > Disable unused objects
  > return panel
```

Branch Logic
```
> On Setup
  > For each owning player who is alive
    > get first true condition
    > add player to branch group
  > Order branch groups by initiative + base order    Add base order to data parsing
    > Add designated panel set to queue    What about death logic?
```

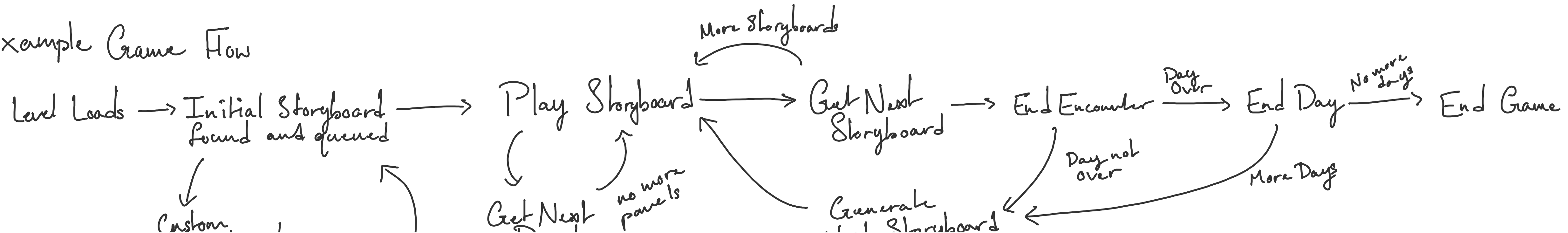
→
- Override/Extend HandlePlayerDeath
- Load Appropriate Death Panel Set
- Queue Immediate

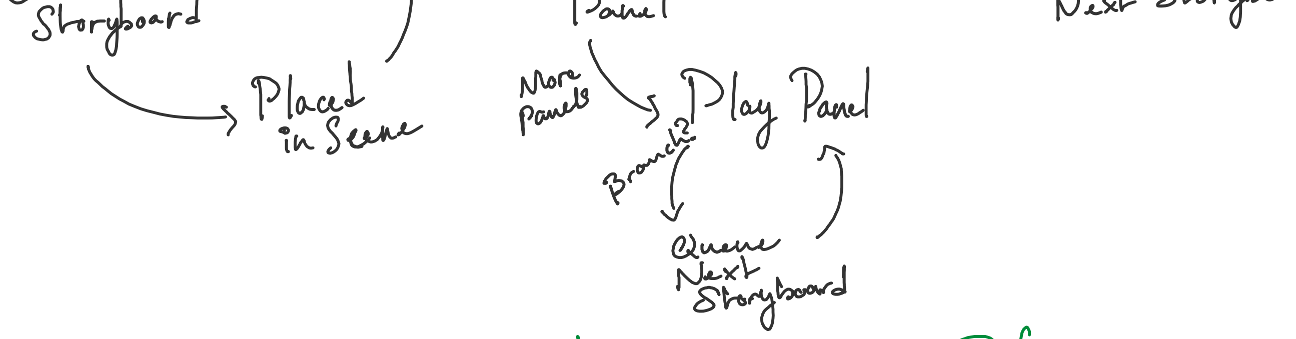
Decision Logic
```
> Populate DecisionControl "Decisions"
  ◦ Decision text
  ◦ OnClick => Add {player} to {decision} group
  ◦ Condition
> Add OnComplete Callback
  => For each decision group
    => If decision has matching PanelSet
      => Queue Panel Set with Selecting Players
```

} Decision control will have to be modified to remove storyboard assumption

Narration Panel
```
> Populate text and owning players
> Set timer control based on length of text } Could try normalizing for non-word text
```

Example Game Flow





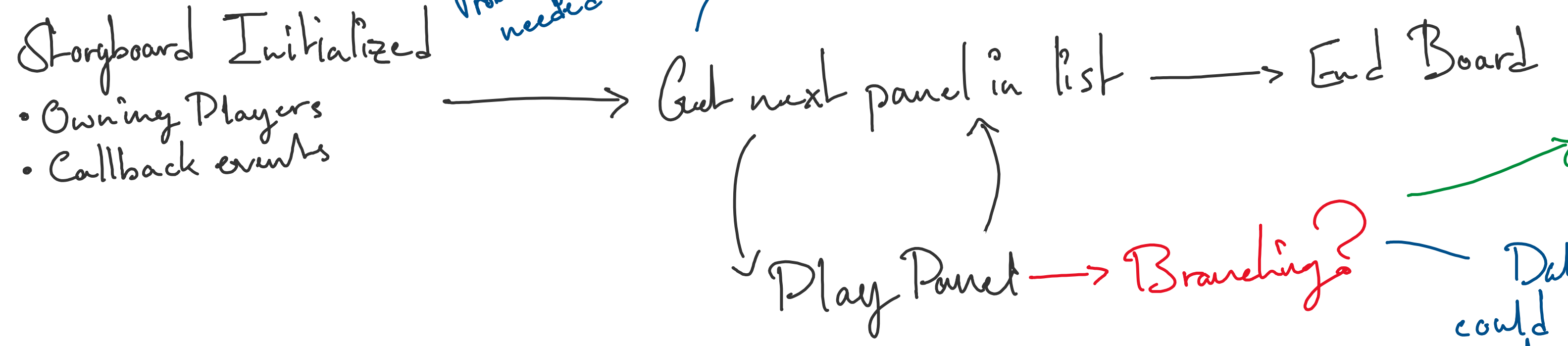
## Custom SB Flow

! Could implement path merging in SBQ. Before instatiating, check if SB is already queued. If so, merge player sets.

~~Make sure SBs aren't instantiated before queuing~~

Probably not needed

Storyboards will be treated as linear sets of panels.



Mostly same as before, but now handled by a specialized component which is triggered by events.

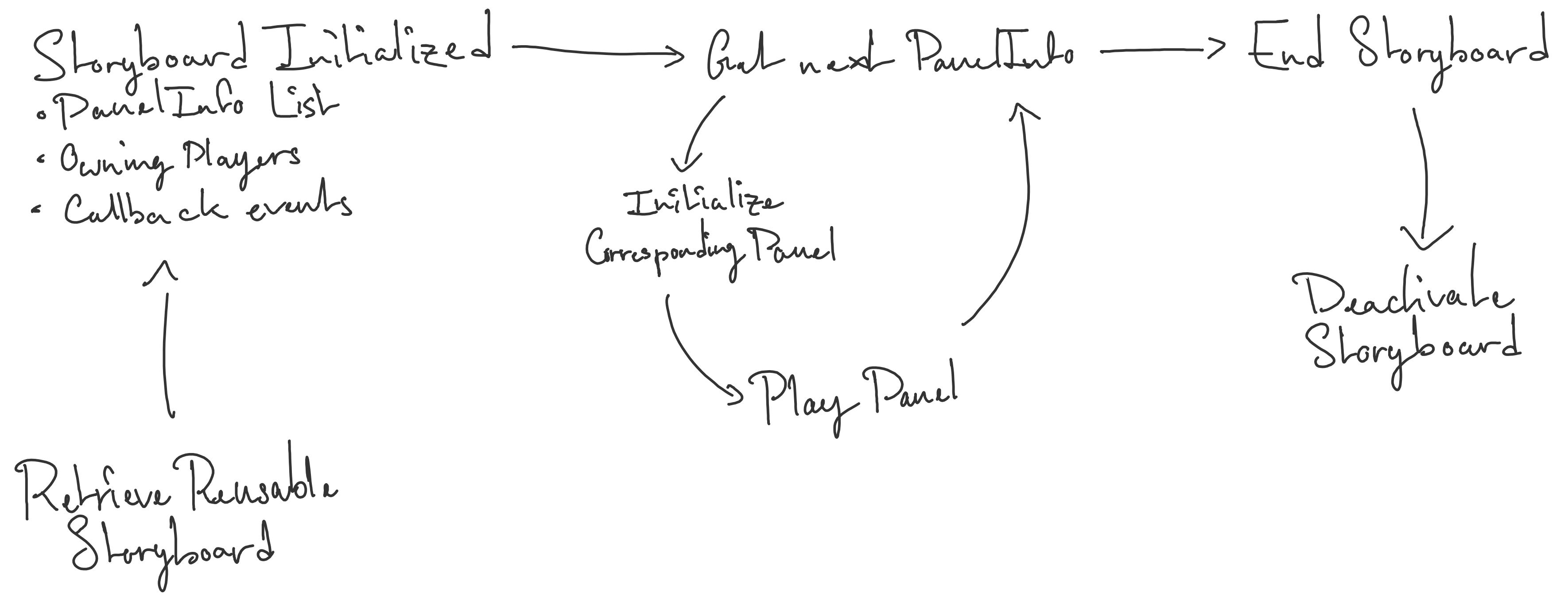
Data based storyboards could be loaded by a standalone caches, which keys a set of storyboards to the name of the file they're loaded from.

Could potentially standardize the interface for loading storyboard data to a URI

It's a good idea and I like it!

This would also allow for DataDriven SBs to refer to Custom ones by URL

## Data Driven SB Flow



## Even More Detailed Flow?

```
StoryboardQueue.OnStart()
 ↳ Find Objects Of Type<Storyboard>()
 ↳ For each Storyboard
    ↳ Set OwningCharacters to CharacterManager.LivingPlayers
    ↳ Disable Storyboard
    ↳ Enqueue(Storyboard, merge=false)

StoryboardQueue.Enqueue(Storyboard, ShouldMerge?)
 ↳ If ShouldMerge?
    ↳ Find MatchingStoryboard in queue
       ↳ If exists, merge Players with Match.Players
 ↳ Add Storyboard to Queue
 ↳ If Queue.Count == 1
    ↳ PlayNextBoard()  ——>
```

Make an extension

```
StoryboardQueue.PlayNextBoard()
↳ if Queue.Count < 1
   ↳ OnQueueDrained.Invoke()
   ↳ return
↳ Queue.Current.OnEnd.Add => {
   ↳ Queue.Dequeue
   ↳ PlayNextBoard()
   }
↳ Storyboard.Play()  ——>  Storyboard.Play()
```

Check if any other steps occur in the current queue logic. If so try to push into the Play() method.

Finish this →