the gamedesigninitiative
at cornell university

Memory Management

Gaming Memory (Generation 7)

- **Playstation 3**
  - 256 MB RAM for system
  - 256 MB for graphics card
- **X-Box 360**
  - 512 MB RAM (unified)
- **Nintendo Wii**
  - 88 MB RAM (unified)
  - 24 MB for graphics card
- **iPhone/iPad**
  - 1 GB RAM (unified)

Gaming Memory (Generation 8)

- **Playstation 4**
  - 8 GB RAM (unified)
- **X-Box One**
  - 12 GB RAM (unified)
  - 9 GB for games
- **Nintendo Wii-U**
  - 2 GB RAM (unified)
  - 1 GB only for OS
- **iPhone/iPad**
  - 2 GB RAM (unified)

Gaming Memory (Current Generation)

- **Playstation 5**
  - 16 GB RAM (unified)
  - **Speed** 448GB/s
- **X-Box Series X**
  - 16 GB RAM (unified)
  - **Speed** 560-336GB/s
- **Nintendo Switch**
  - 3 GB RAM (unified)
  - **Speed** 25.6 GB/s
- **iPhone/iPad**
  - 6 GB RAM (unified)
  - **Speed** 42.7 GB/s

Gaming Memory (Current Generation)

- **Playstation 5**
  - 16 GB RAM (unified)
  - **Speed** 448GB/s
- **X-Box Series X**
  - 16 GB RAM (unified)
  - **Speed** 560-336GB/s
- **Nintendo Switch**
  - 3 GB RAM (unified)
  - **Speed** 25.6 GB/s
- **iPhone/iPad**
  - 6 GB RAM (unified)
  - **Speed** 42.7 GB/s

You can make Switch quality games for iOS

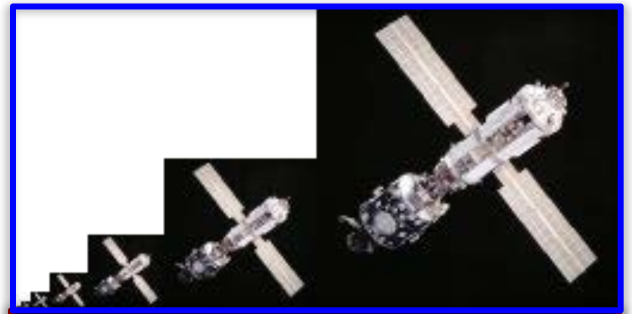
Memory Usage: Images

- Pixel color is 4 bytes
  - 1 byte each for r, b, g, alpha
  - More if using HDR color
- Image a **2D array** of pixels
  - 1280x1024 monitor size
  - 5,242,880 bytes ~ 5 MB
- More if using **mipmaps**
  - Graphic card texture feature
  - Smaller versions of image
  - Cached for performance
  - But can double memory use

Memory Usage: Images

- Pixel color is 4 bytes
  - 1 byte each for r, b, g, alpha
  - More if using HDR color
- Image a **2D array** of pixels
  - 1280x1024 monitor size
  - 5,242,880 bytes ~ 5 MB
- More if using **mipmaps**
  - Graphic card texture feature
  - Smaller versions of image
  - Cached for performance
  - But can double memory use

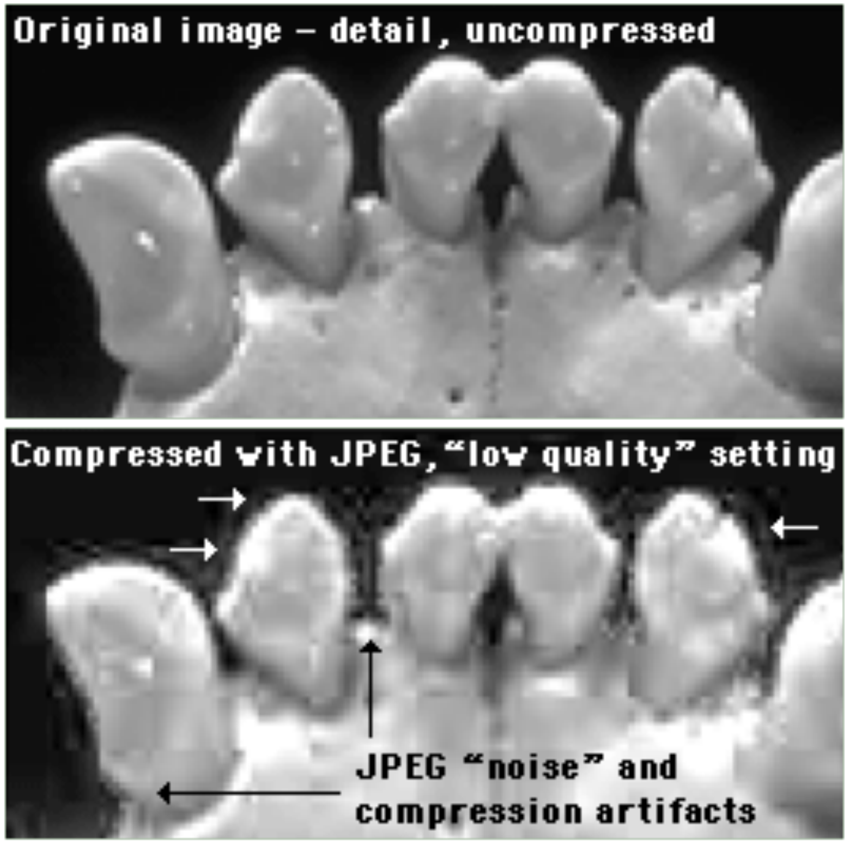
MipMaps



Original Image

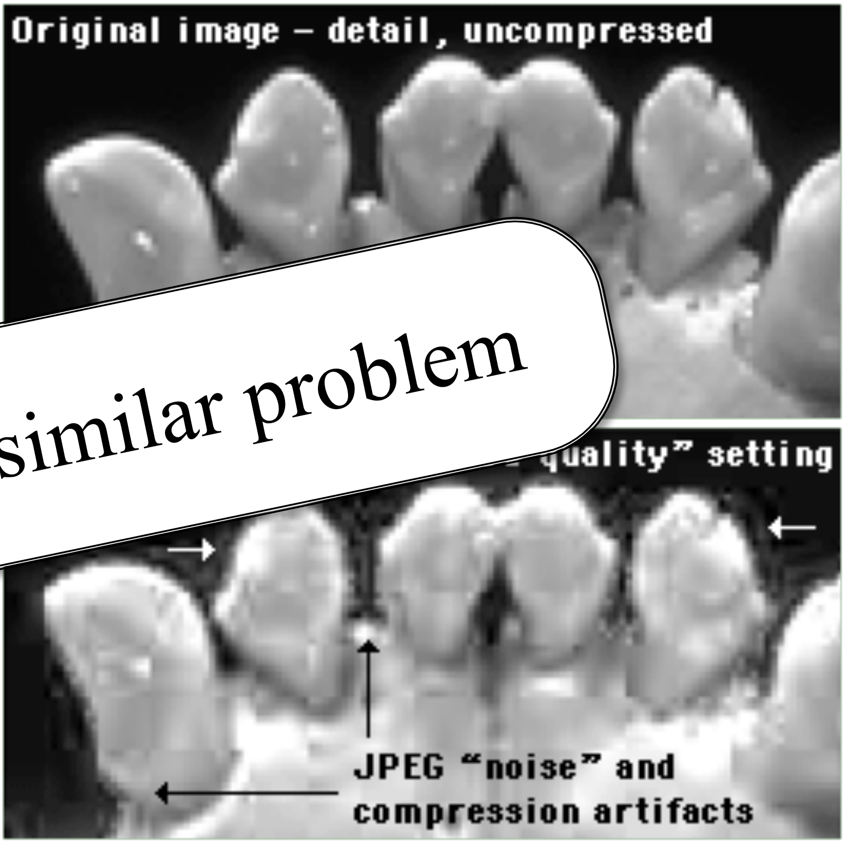
But My JPEG is only 8 KB!

- Formats often **compressed**
  - JPEG, PNG, GIF
  - But not always TIFF
- Must **uncompress** to show
  - Need space to uncompress
  - In RAM or graphics card
- Only load when needed
  - Loading is primary I/O operation in AAA games
  - Causes "texture popping"


Original image – detail, uncompressed
Compressed with JPEG, "low quality" setting
JPEG "noise" and compression artifacts

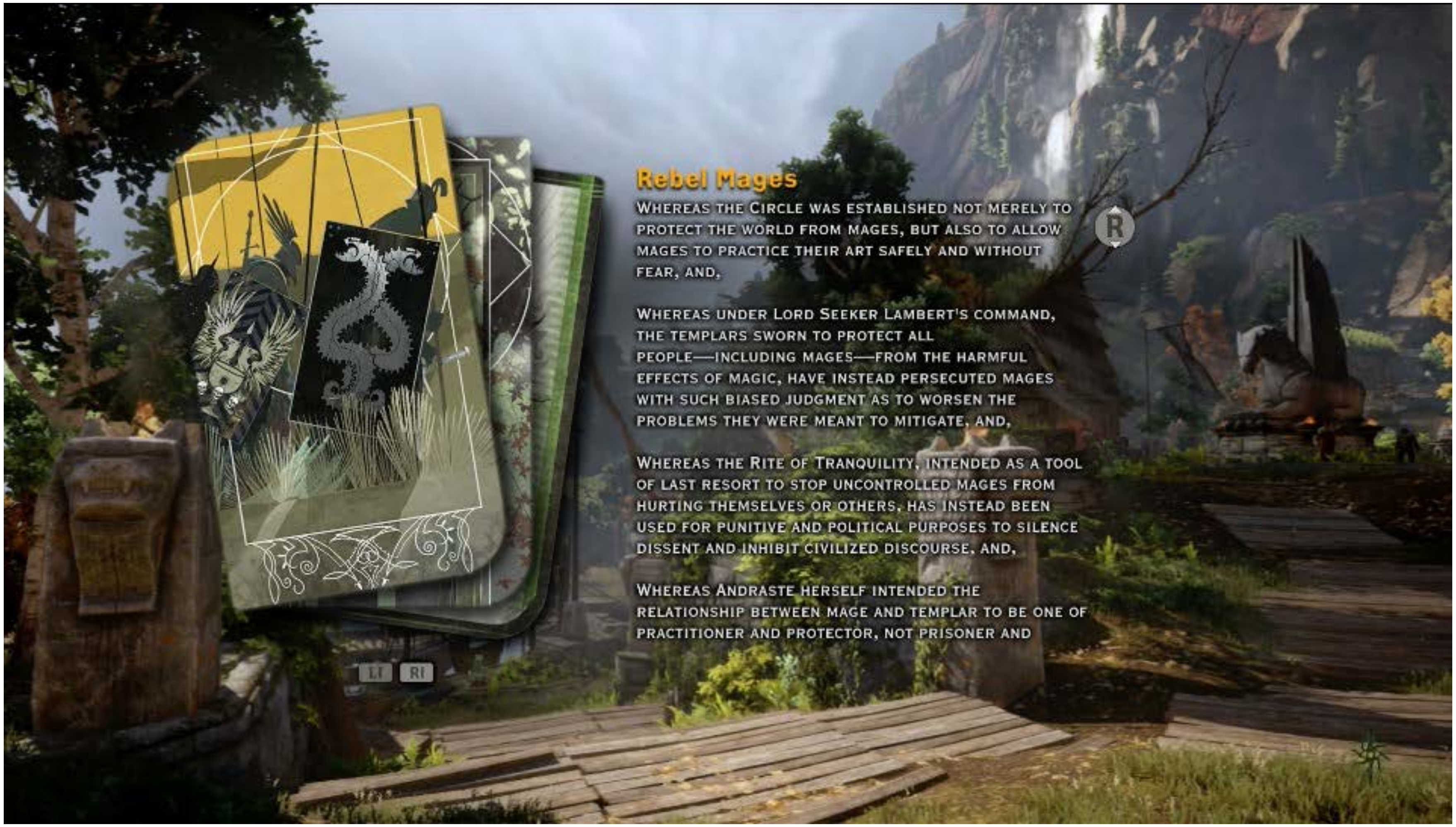
But My JPEG is only 8 KB!

- Formats often **compressed**
  - JPEG, PNG, GIF
  - But not always TIFF
- Must **uncompress** to show
  - Need space to uncompr
  - In RA
- Only lo
  - Loading is primary I/O operation in AAA games
  - Causes "texture popping"


Original image – detail, uncompressed
quality" setting
JPEG "noise" and compression artifacts

Sounds have a similar problem

Loading Screens



Rebel Mages

Whereas the Circle was established not merely to protect the world from mages, but also to allow mages to practice their art safely and without fear, and,

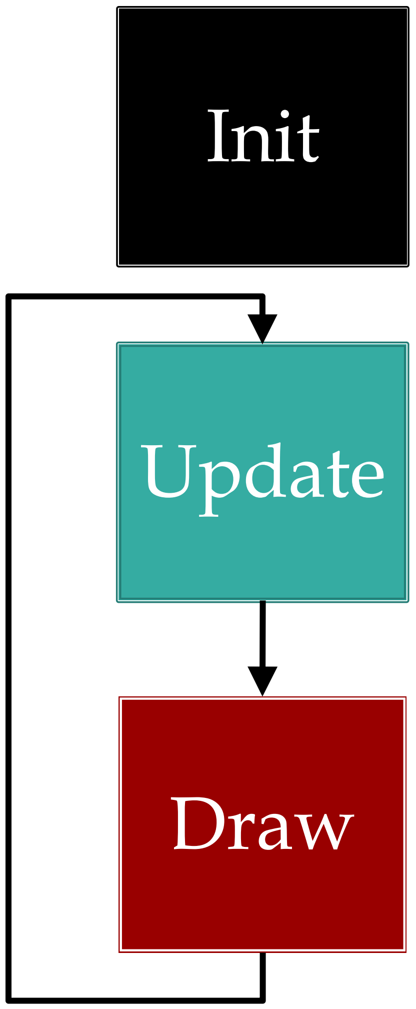
Whereas under Lord Seeker Lambert's command, the templars sworn to protect all people—including mages—from the harmful effects of magic, have instead persecuted mages with such biased judgment as to worsen the problems they were meant to mitigate, and,

Whereas the Rite of Tranquility, intended as a tool of last resort to stop uncontrolled mages from hurting themselves or others, has instead been used for punitive and political purposes to silence dissent and inhibit civilized discourse, and,

Whereas Andraste herself intended the relationship between mage and templar to be one of practitioner and protector, not prisoner and

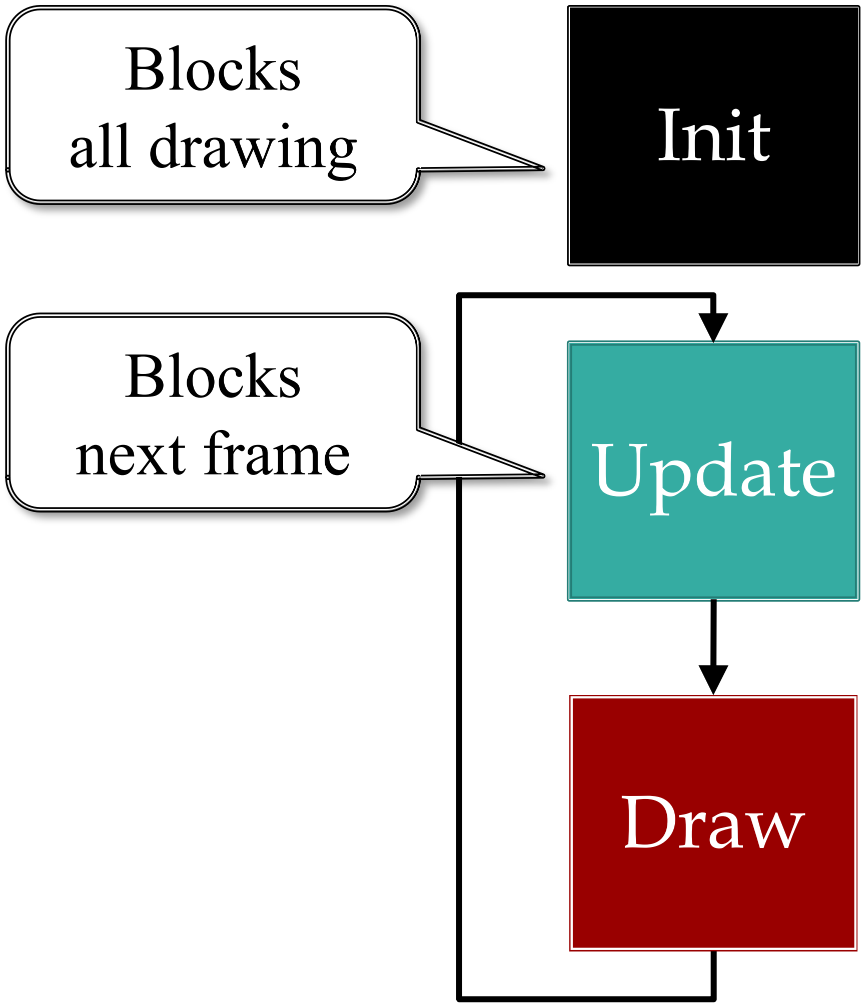
Problems with Asset Loading

- How to load assets?
  - May have a lot of assets
  - May have large assets
- Loading is **blocking**
  - Game stops until done
  - Cannot draw or animate
- May need to **unload**
  - Running out of memory
  - Free something first

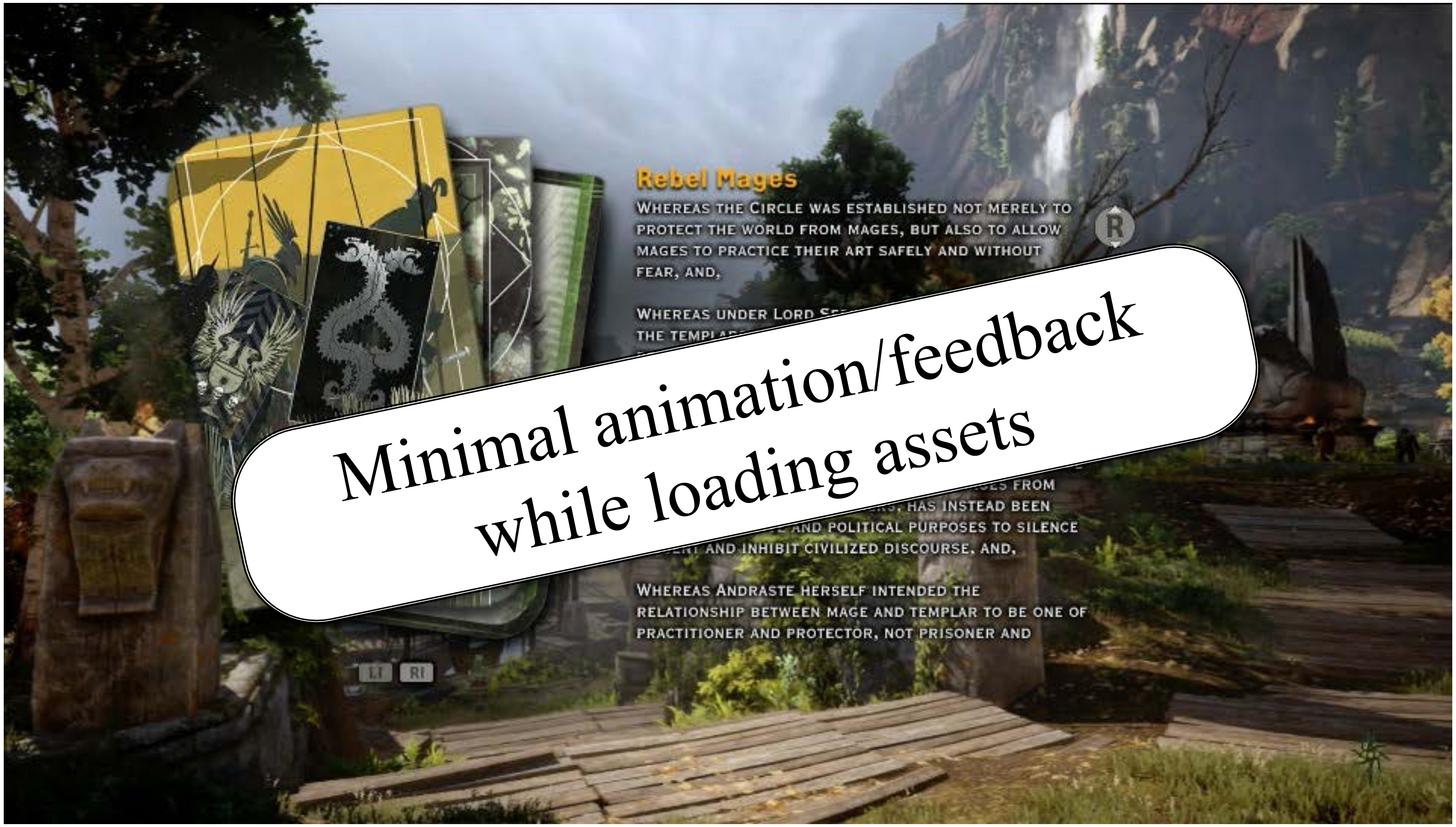


Problems with Asset Loading

- How to load assets?
  - May have a lot of assets
  - May have large assets
- Loading is **blocking**
  - Game stops until done
  - Cannot draw or animate
- May need to **unload**
  - Running out of memory
  - Free something first



Loading Screens



Rebel Mages

WHEREAS THE CIRCLE WAS ESTABLISHED NOT MERELY TO PROTECT THE WORLD FROM MAGES, BUT ALSO TO ALLOW MAGES TO PRACTICE THEIR ART SAFELY AND WITHOUT FEAR, AND,

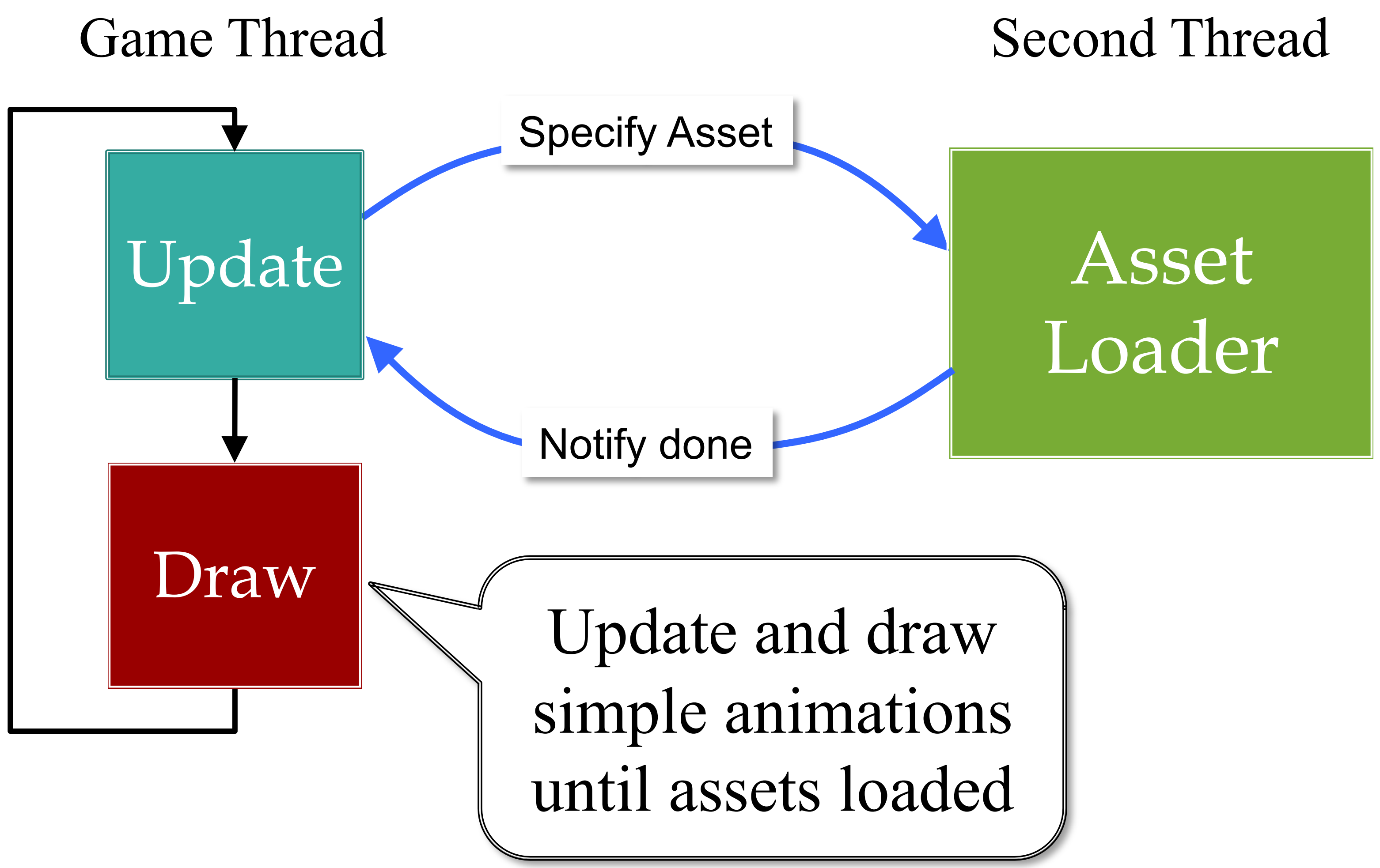
WHEREAS UNDER LORD SE
THE TEMPLA

FROM
HAS INSTEAD BEEN
AND POLITICAL PURPOSES TO SILENCE
AND INHIBIT CIVILIZED DISCOURSE, AND,

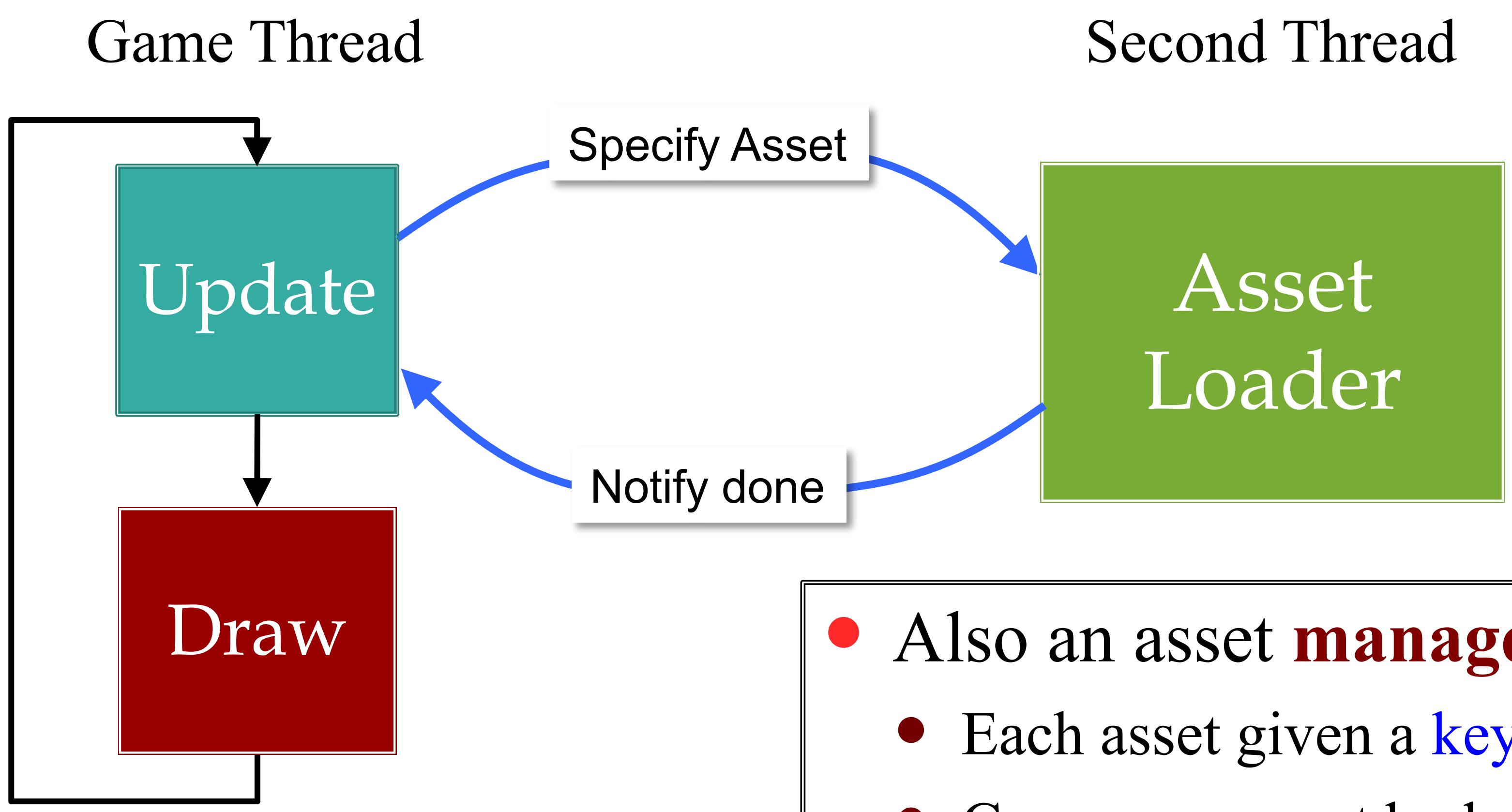
WHEREAS ANDRASTE HERSELF INTENDED THE RELATIONSHIP BETWEEN MAGE AND TEMPLAR TO BE ONE OF PRACTITIONER AND PROTECTOR, NOT PRISONER AND

Minimal animation/feedback while loading assets

Solution: Asynchronous Loader

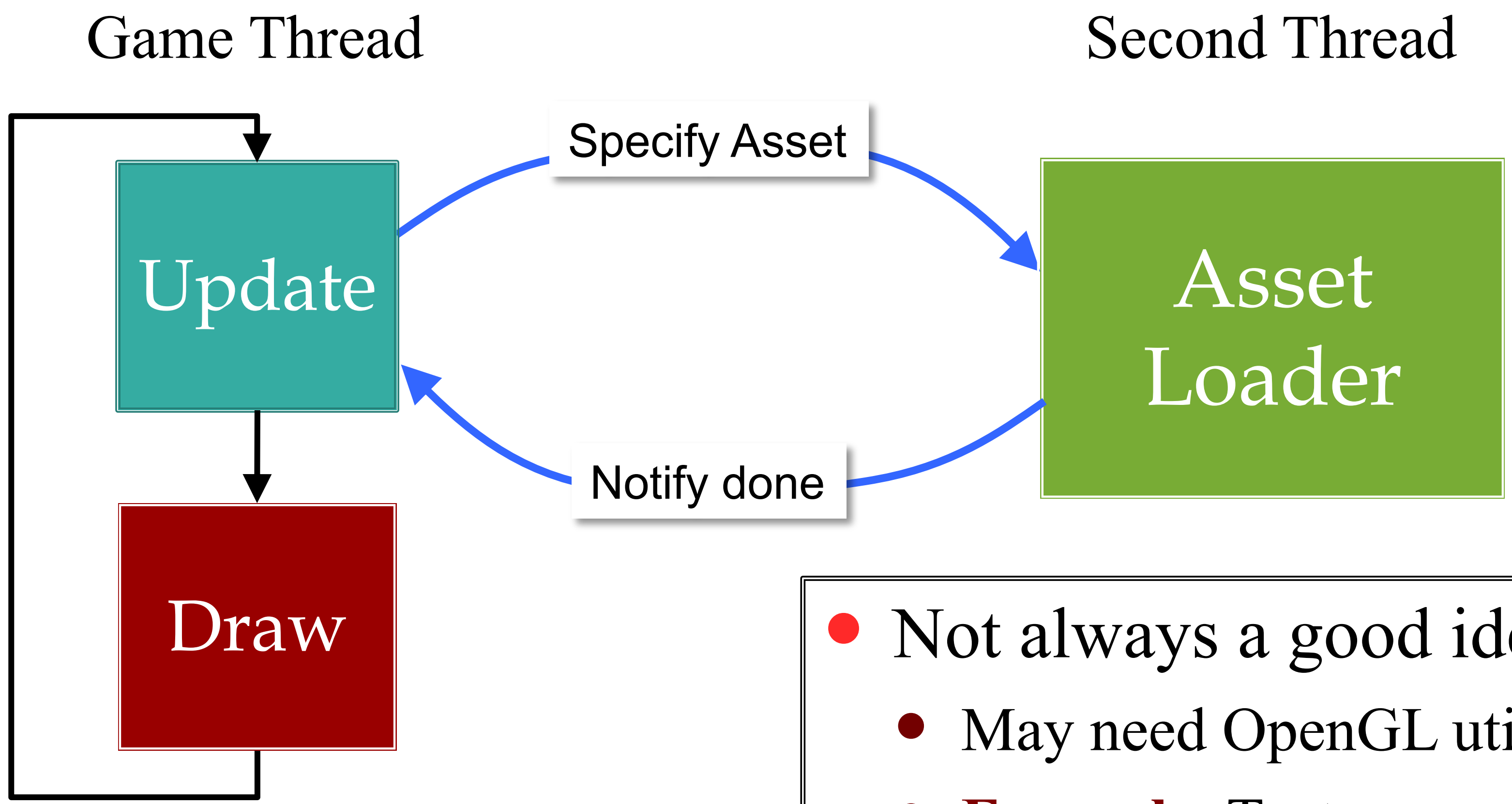


Solution: Asynchronous Loader



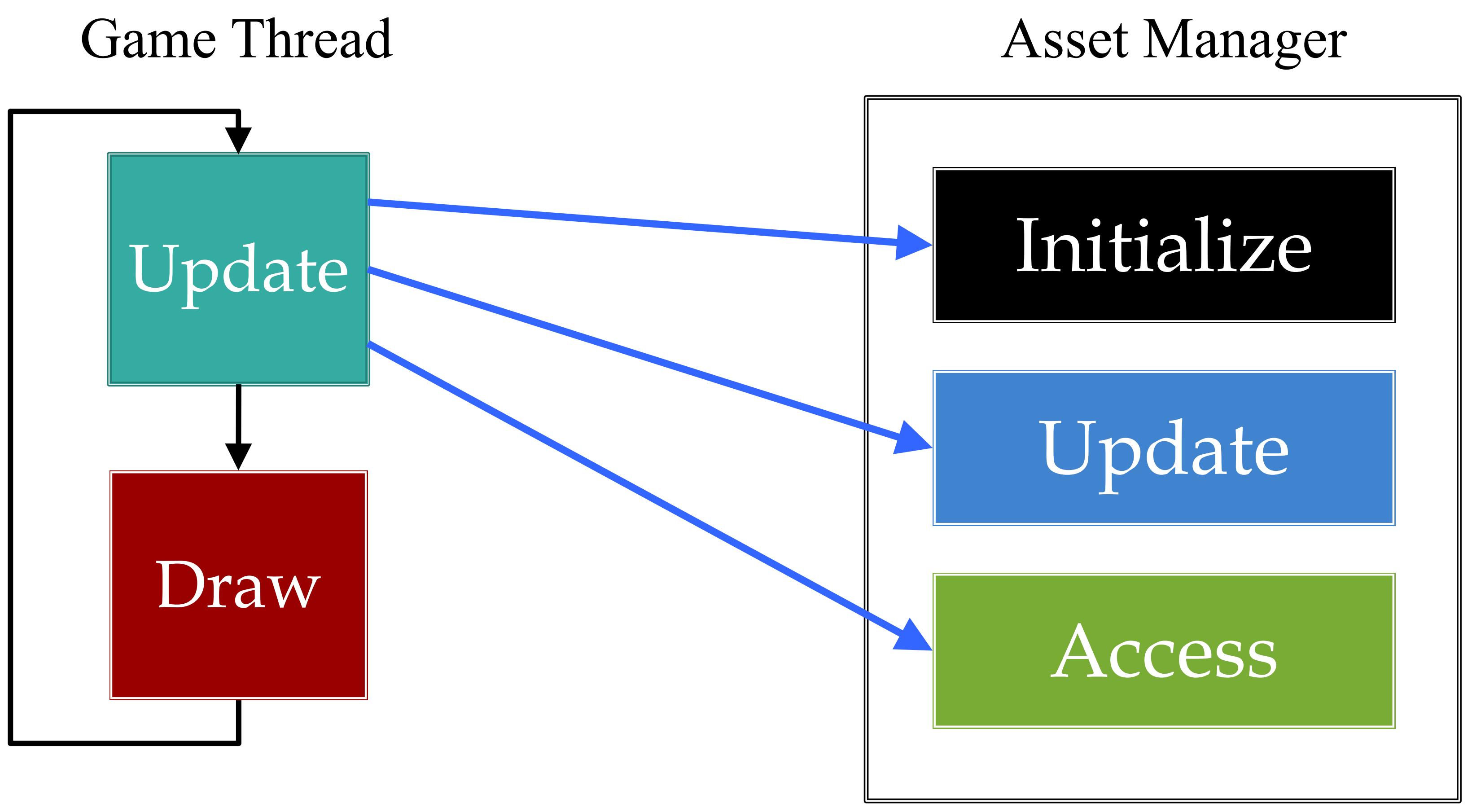
- Also an asset **manager**
  - Each asset given a key
  - Can access asset by key
  - Like a map/hash table

Solution: Asynchronous Loader



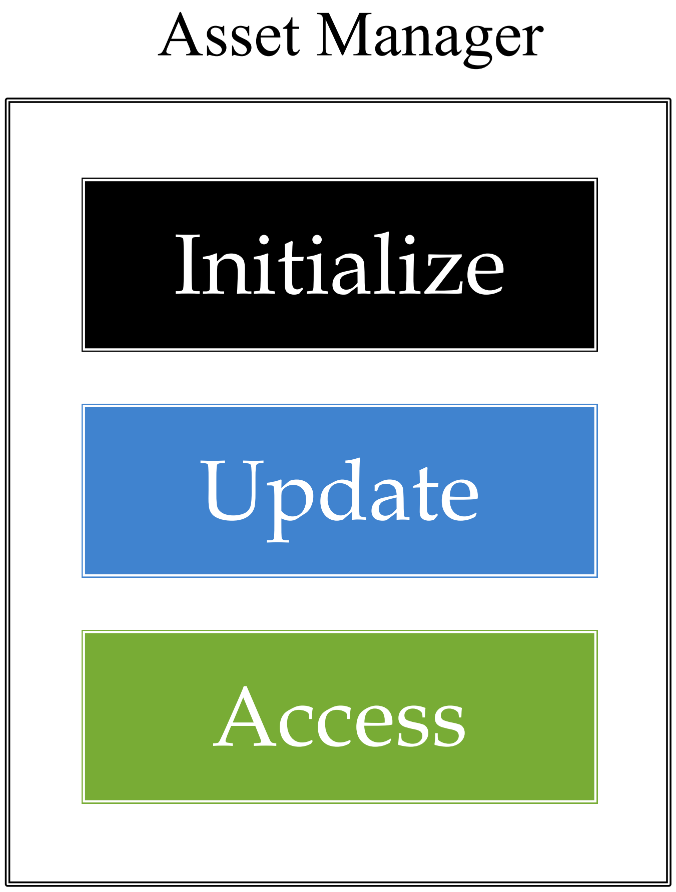
- Not always a good idea
  - May need OpenGL utils
  - **Example**: Textures
  - Limited to main thread

Alternative: Iterative Loader



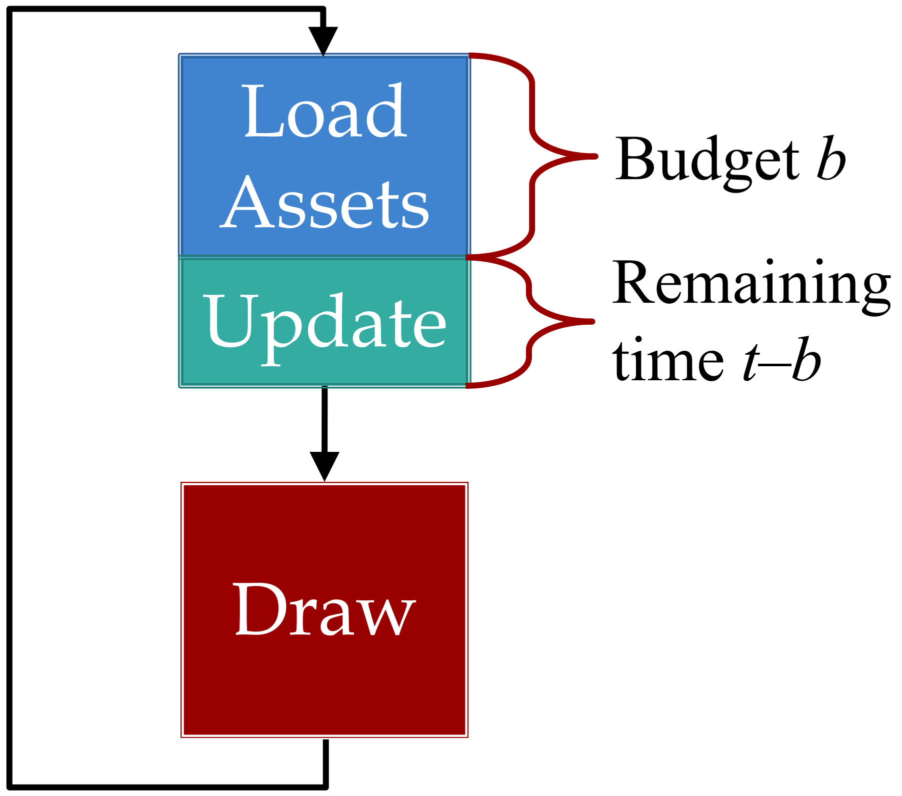
Alternative: Iterative Loader

- Uses a time budget
  - Give set amount of time
  - Do as much as possible
  - Stop until next update
- Better for OpenGL
  - Give time to manager
  - Animate with remainder
  - No resource contention
- LibGDX approach
  - But async behind scenes



Alternative: Iterative Loader

- Uses a time budget
  - Give set amount of time
  - Do as much as possible
  - Stop until next update
- Better for OpenGL
  - Give time to manager
  - Animate with remainder
  - No resource contention
- LibGDX approach
  - But async behind scenes



Assets Beyond Images

- AAA games have a lot of 3D geometry
  - Vertices for model polygons
  - Physics bodies **per polygon**
  - Scene graphs for organizing this data
- **How do we load these things?**
  - Managers handle built-in asset types
  - What if we need to make a custom data type?
- And exactly when do we load these?

CUGL Approach

AssetManager

- Map from keys to assets
    - All access is templated
    - `assets->get<Texture>("image")`
    - Keys unique *per asset*
- Requires attached loaders
    - `a->attach<T>(load1->getHook());`
    - `a->attach<F>(load2->getHook());`
- "Hook" is C++ workaround
    - For template subclassing
    - Make custom loaders easier

Loader

- `void read(key, src, cb, async)`
    - Reads asset from file `src`
    - `async` indicates if in sep thread
    - Callback `cb` executed when done
- `void read(json, cb, async)`
    - Values `key` and `src` now in json
    - As are other special properties
- `void materialize(key, asset, cb)`
    - Code to "finish" asset
    - Always in the main thread

CUGL Approach

AssetManager

- Map from keys to assets
  - All access i
  - assets->get<
  - Keys unique
- Requires att
  - a->attach<T):
  - a->attach<F);
- "Hook" is C++ workaround
  - For templat
  - Make custo

Thread Safe

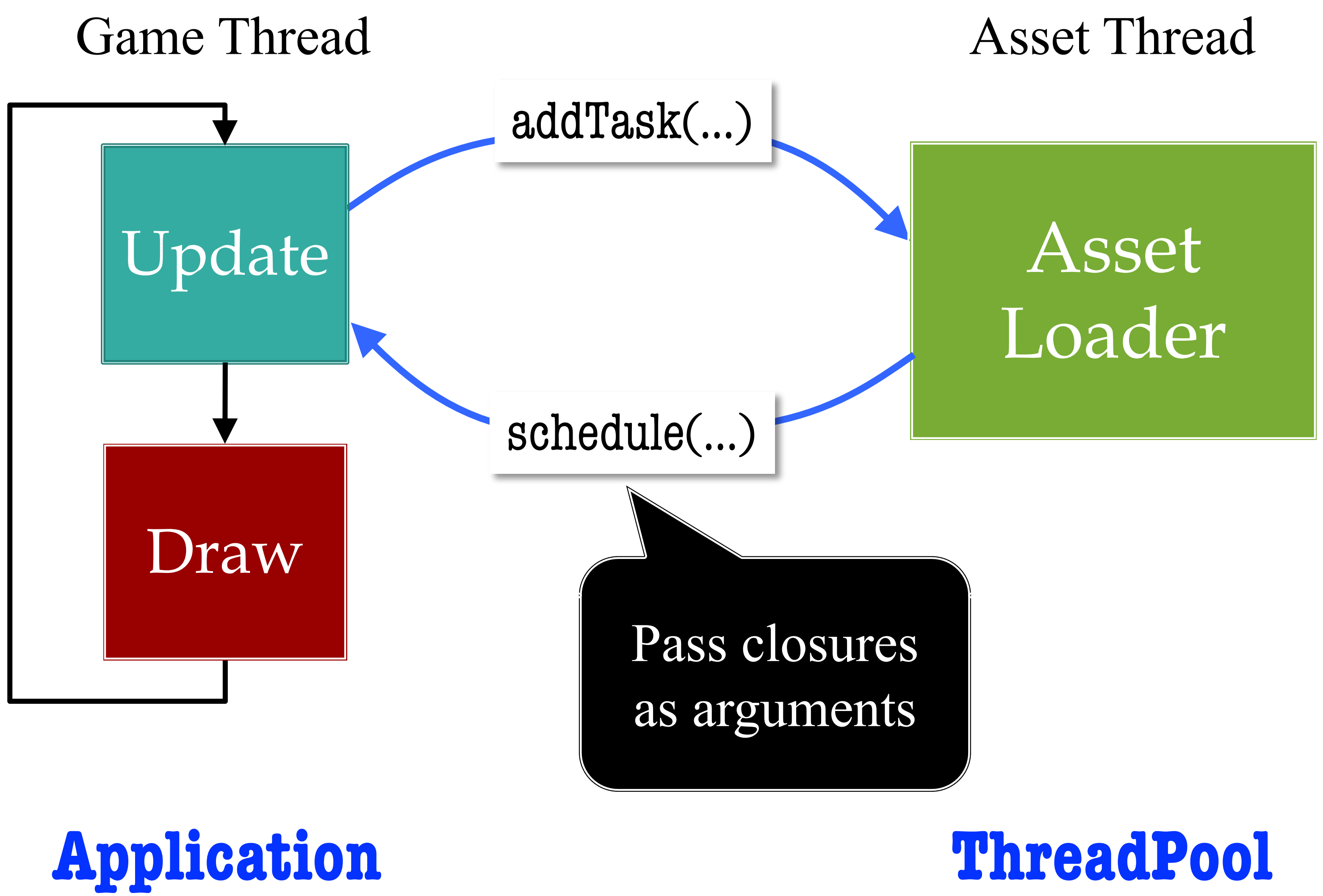
Thread Safe

Main Thread Only

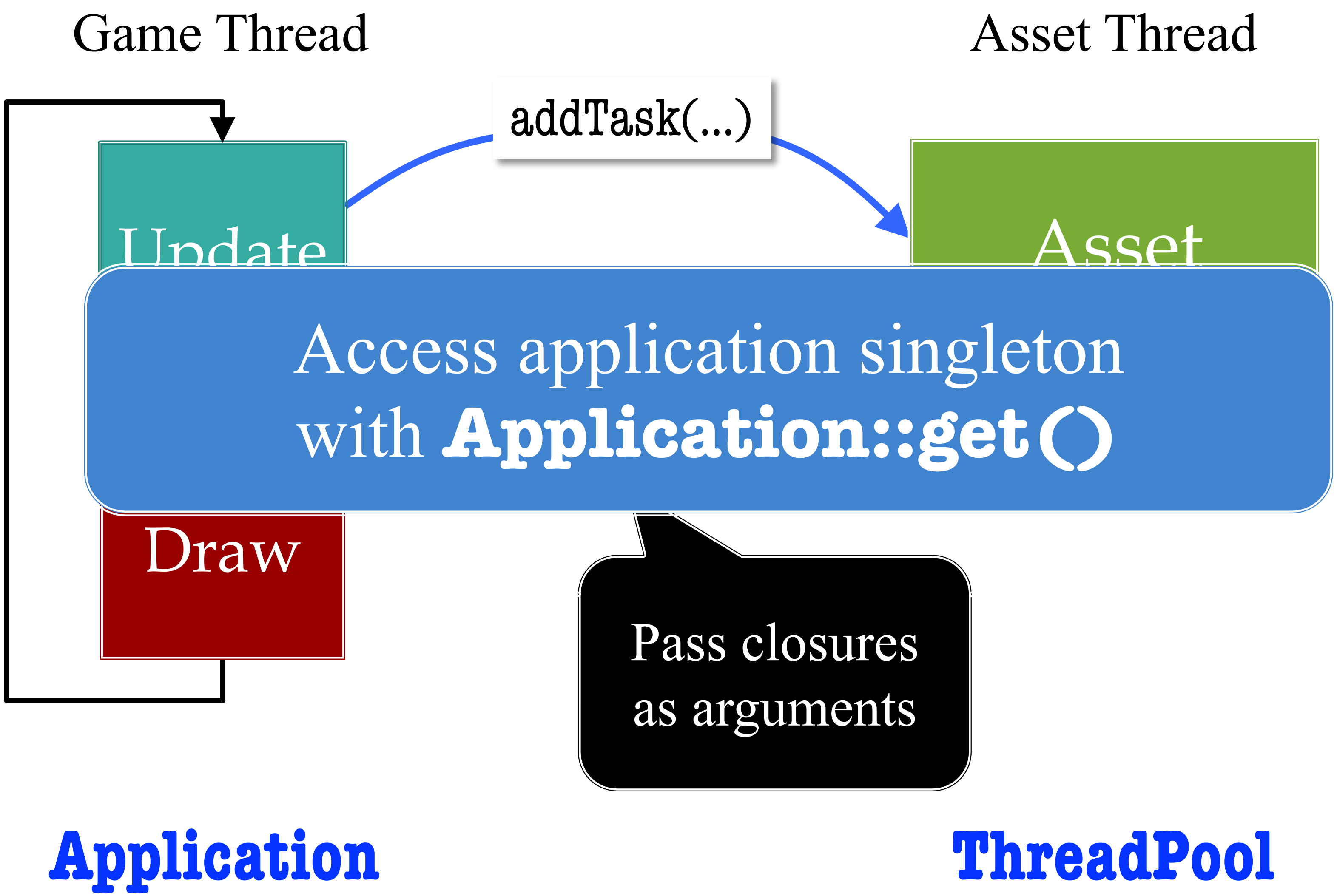
Loader

- void read(key, src, cb, async)
  - Reads asset from file src
  - async indicates if in sep thread
  - Callback cb executed when done
- void read(json, cb, async)
  - Values key and src now in json
  - As are other special properties
- void materialize(key, asset, cb)
  - Code to "finish" asset
  - Always in the main thread

CUGL Approach: Asynchronous



CUGL Approach: Asynchronous



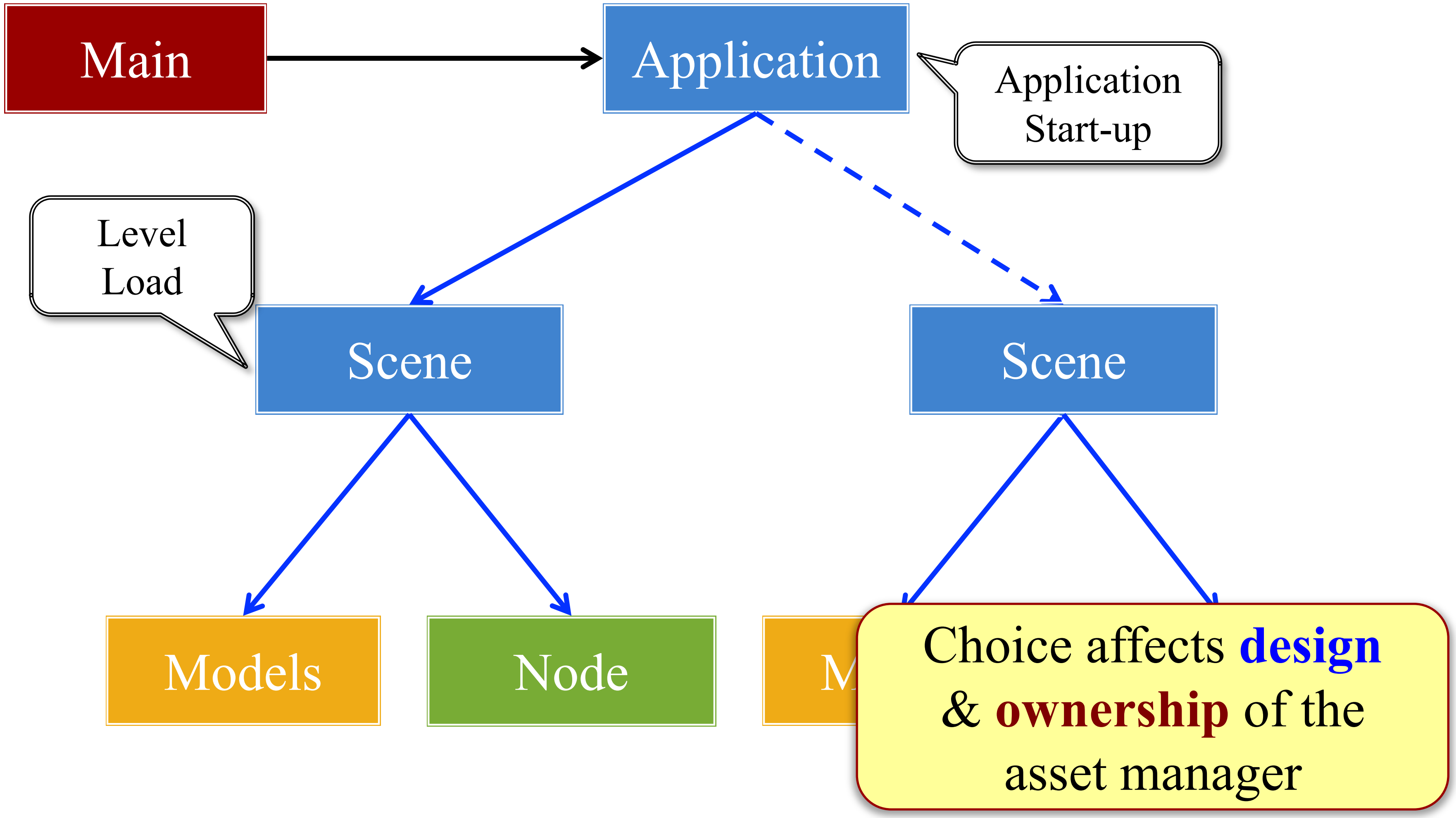
Access application singleton with **Application::get()**

Pass closures as arguments

Assets Beyond Images

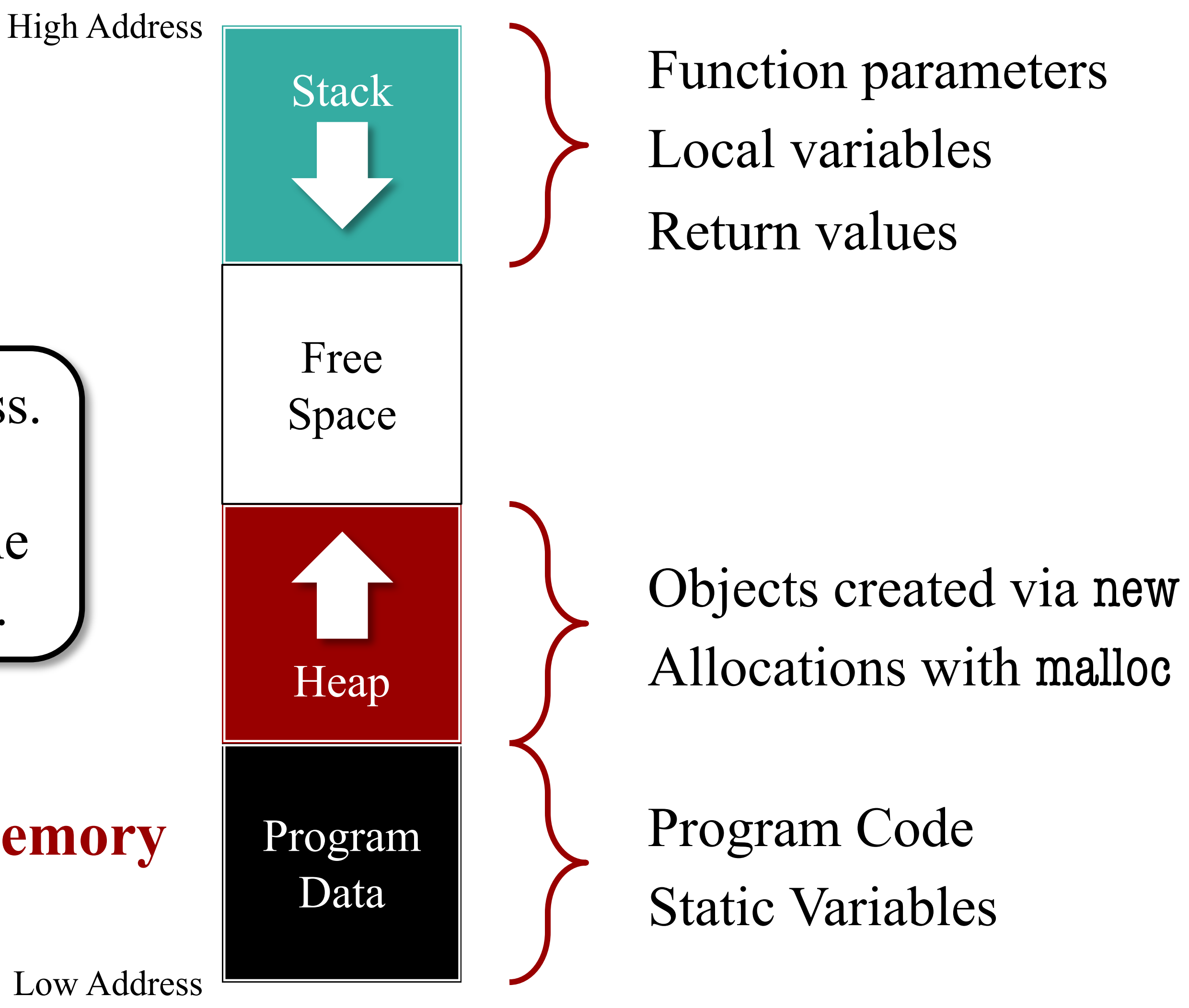
- AAA games have a lot of 3D geometry
  - Vertices for model polygons
  - Physics bodies **per polygon**
  - Scene graphs for organizing this data
- How do we load these things?
  - Managers handle built-in asset types
  - What if we need to make a custom data type?
- **And exactly when do we load these?**

Loading and Architecture



Choice affects **design** & **ownership** of the asset manager

Traditional Memory Organization



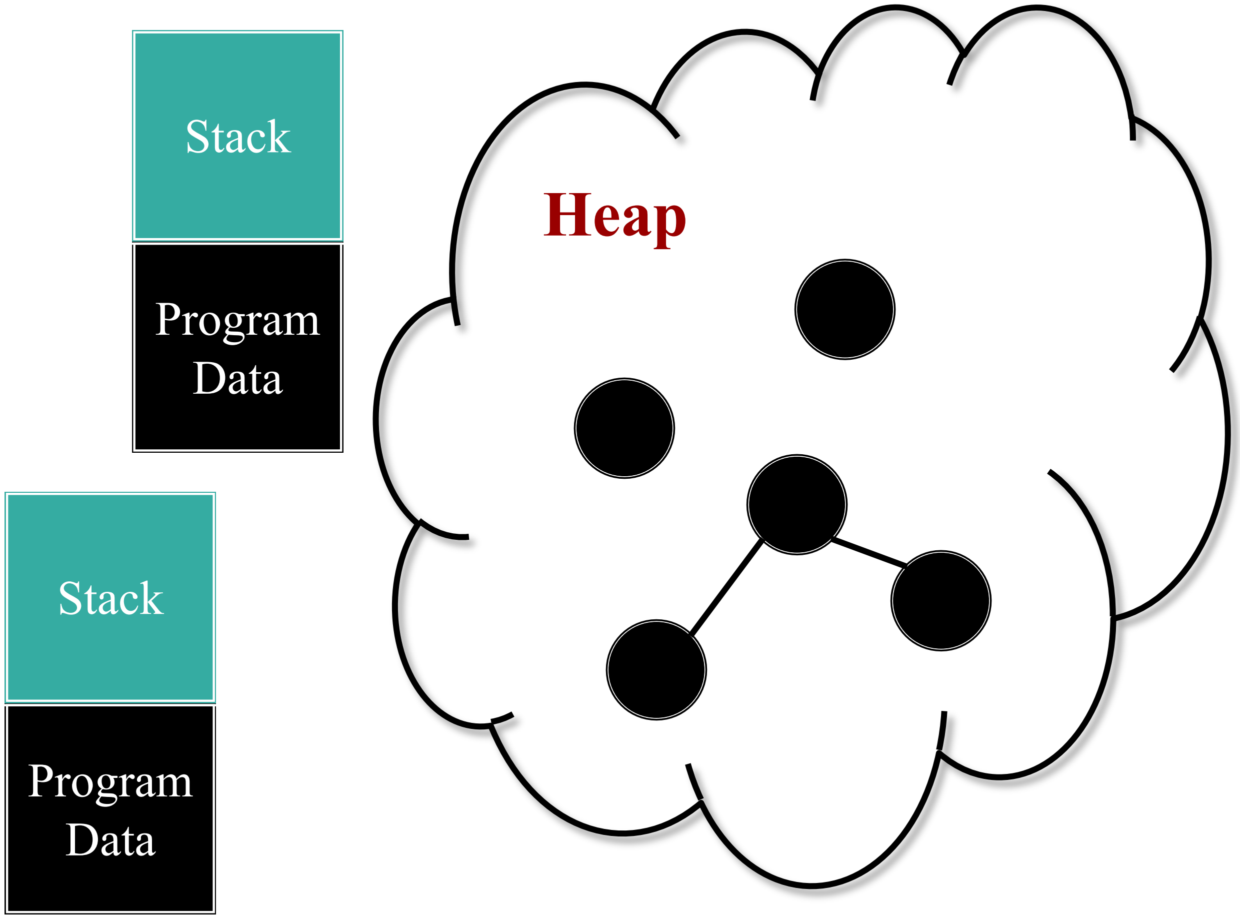
Dedicated to process.

Consists of machine addressable space.

Leverages Virtual Memory

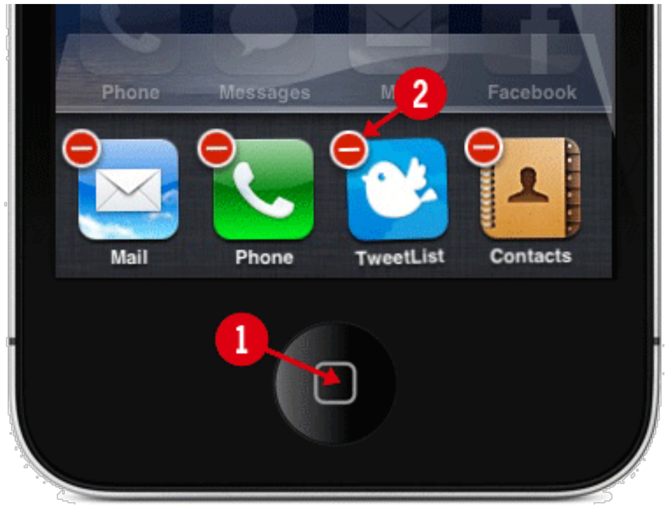
Mobile Memory Organization

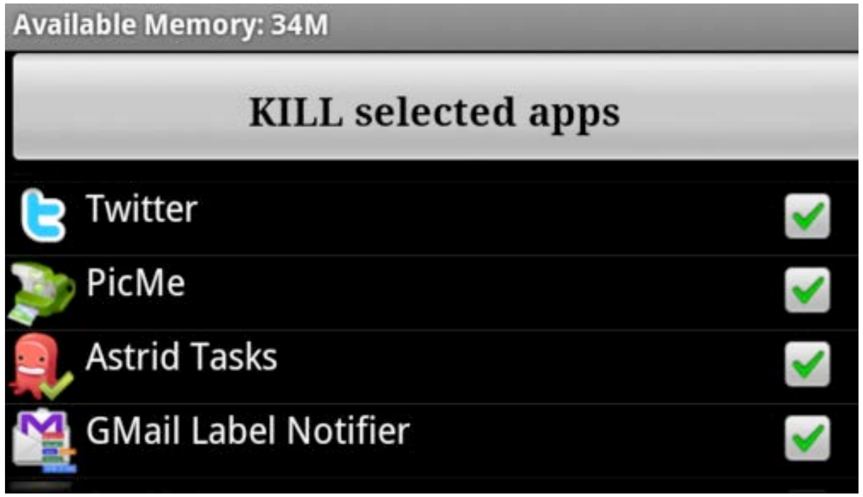
Device Memory



How Do Apps Compete for Memory?

- Active app takes what it can
  - Cannot steal from OS
  - OS may *suspend* apps
- **App Suspension**
  - App quits; memory freed
  - Done only as needed
- Suspend apps can *recover*
  - OS allows limited paging
  - Page out on suspension
  - Page back in on restart



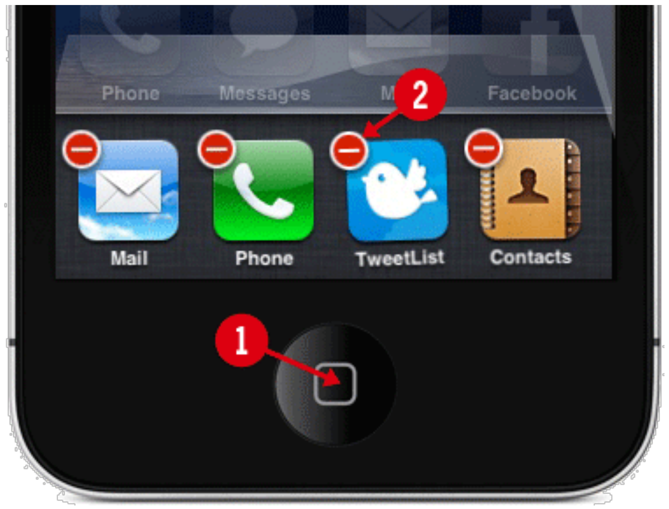

Available Memory: 34M

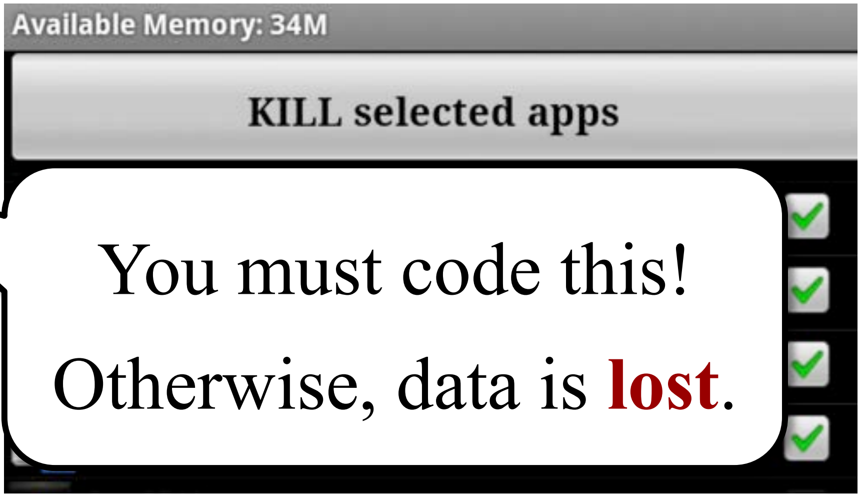
KILL selected apps

Twitter	✓
PicMe	✓
Astrid Tasks	✓
GMail Label Notifier	✓

How Do Apps Compete for Memory?

- Active app takes what it can
  - Cannot steal from OS
  - OS may *suspend* apps
- **App Suspension**
  - App quits; memory freed
  - Done only as needed
- Suspend apps can *recover*
  - OS allows limited paging
  - Page out on suspension
  - Page back in on restart

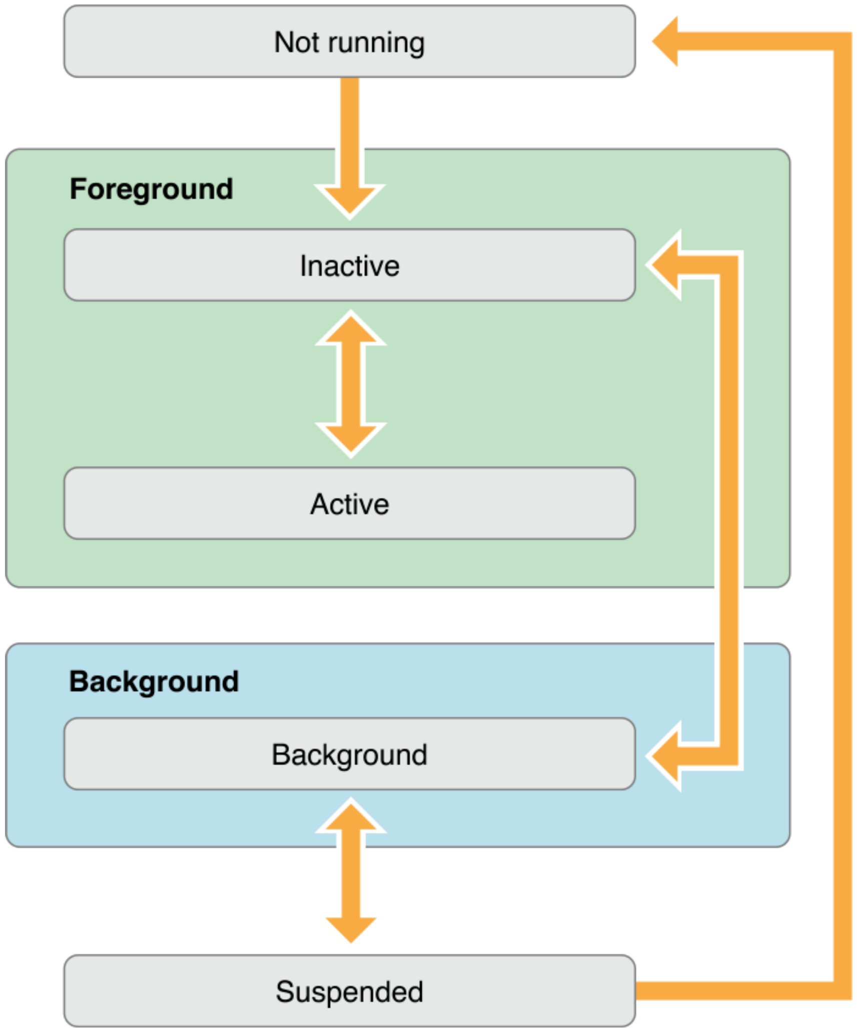




You must code this!

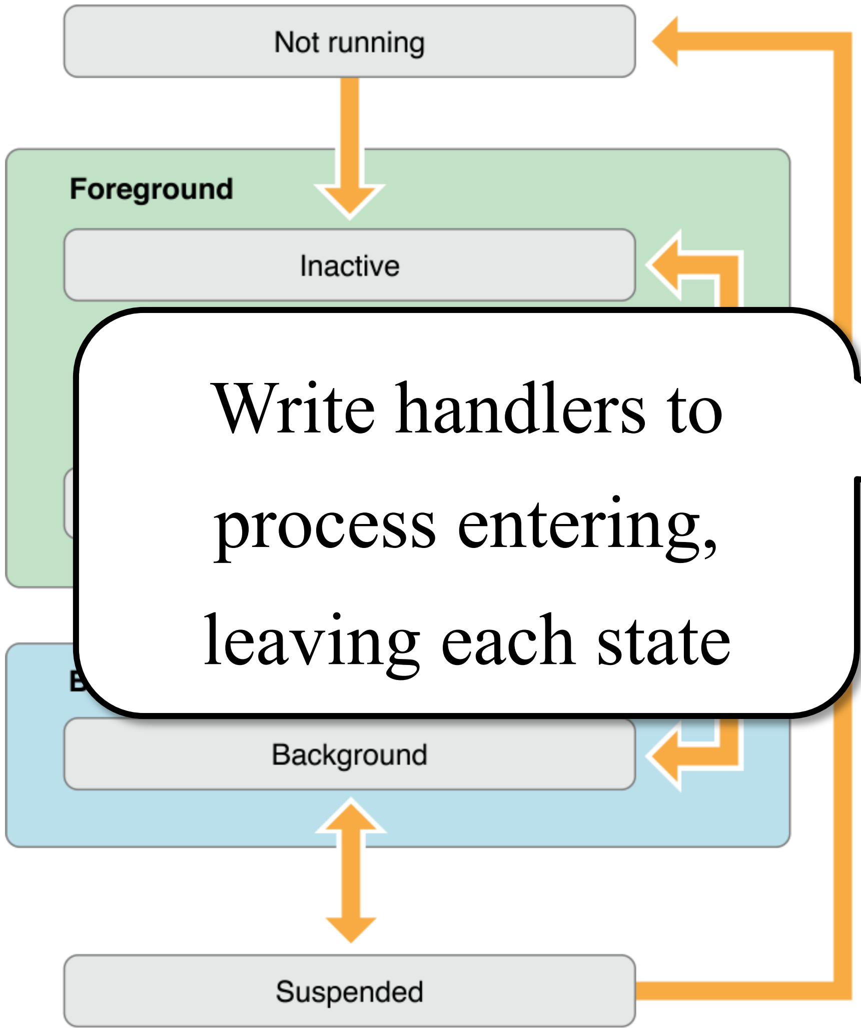
Otherwise, data is **lost**.

State Management in iOS



- **Active**
  - Running & getting input
- **Inactive**
  - Running, but no input
  - Transition to suspended
- **Background**
  - Same as inactive
  - But apps can stay here
  - **Example**: Music
- **Suspended**
  - Stopped & Memory freed

State Management in iOS



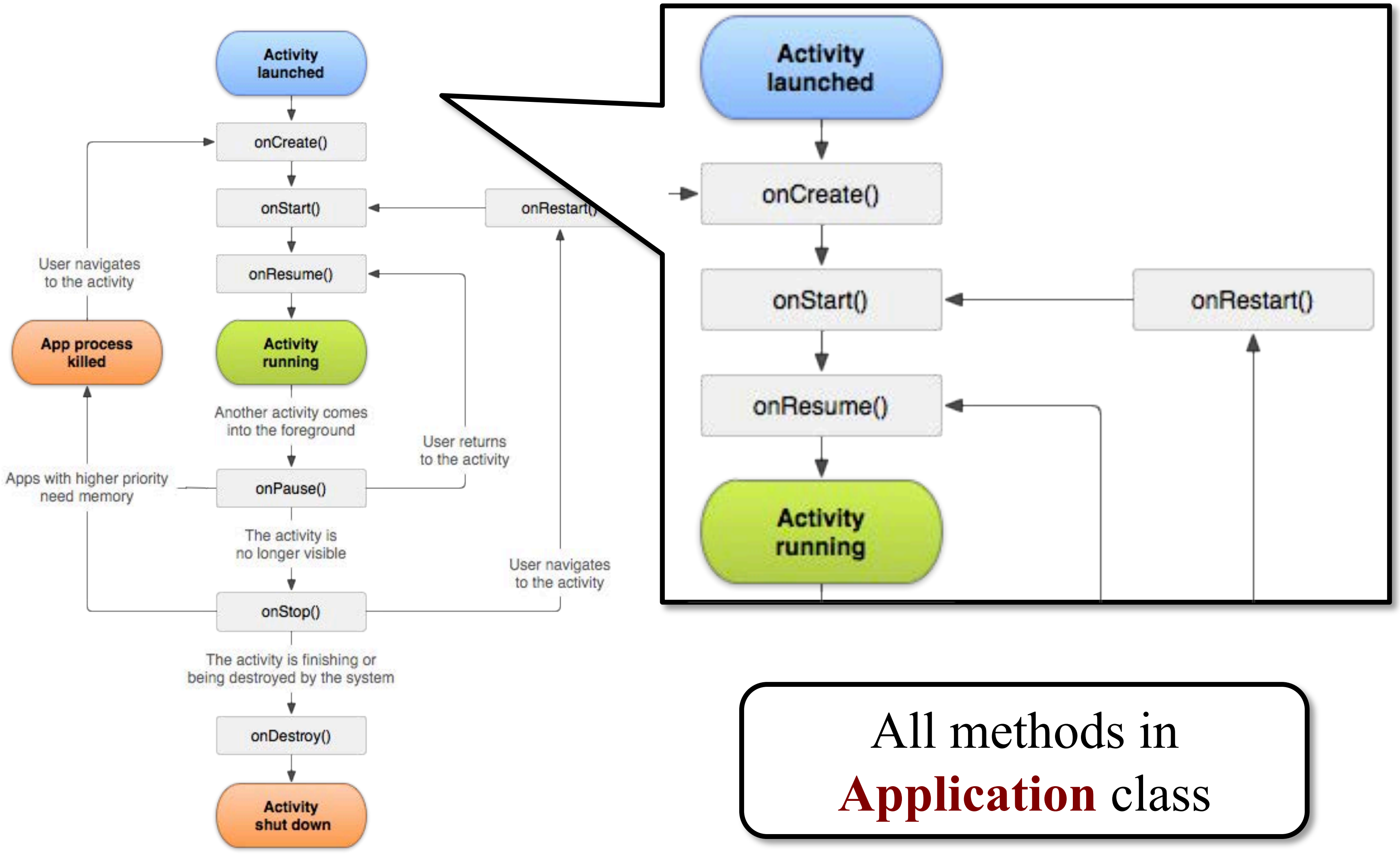
Write handlers to process entering, leaving each state

- **Active**
  - Running & getting input
- **Inactive**
  - Running, but no input
  - Transition to suspended
- **Background**
  - Same as inactive
  - But apps can stay here
  - **Example**: Music
- **Suspended**
  - Stopped & Memory freed

iOS State Handling

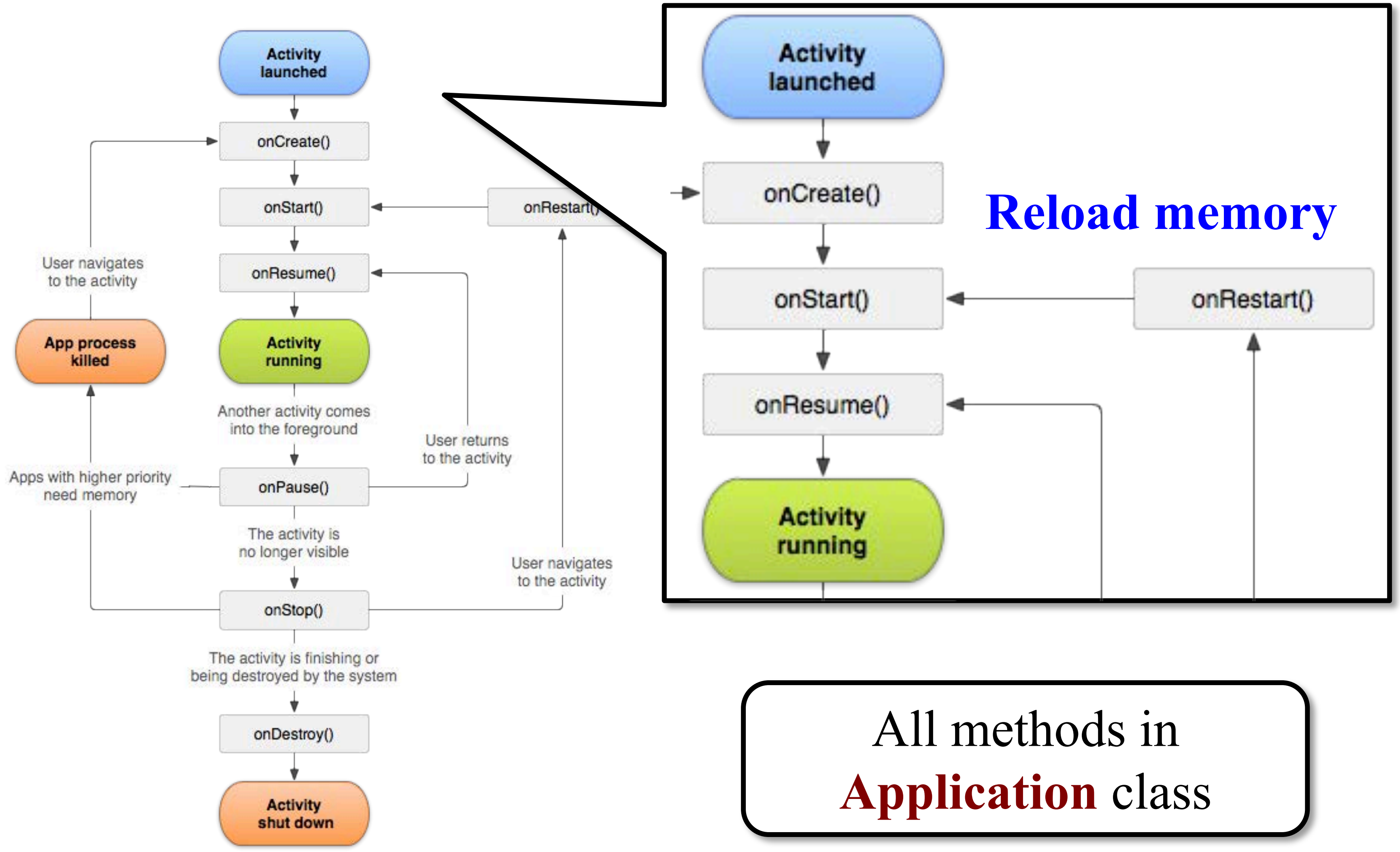
- `applicationDidBecomeActive:`
  - Your app became (resumed as) the foreground app.
  - Use this to recover memory state.
- `applicationWillResignActive:`
  - Your app will switch to inactive or background.
  - Stop the game loop and page out memory.
- `applicationDidEnterBackground:`
  - Your app is in the background and may be suspended.
- `applicationWillEnterForeground:`
  - Your app is leaving the background, but is not yet active.

Android State Handling


Activity launched
onCreate()
onStart()
onRestart()
onResume()
User navigates to the activity
App process killed
Activity running
Another activity comes into the foreground
User returns to the activity
Apps with higher priority need memory
onPause()
The activity is no longer visible
User navigates to the activity
onStop()
The activity is finishing or being destroyed by the system
onDestroy()
Activity shut down

All methods in **Application** class

Android State Handling

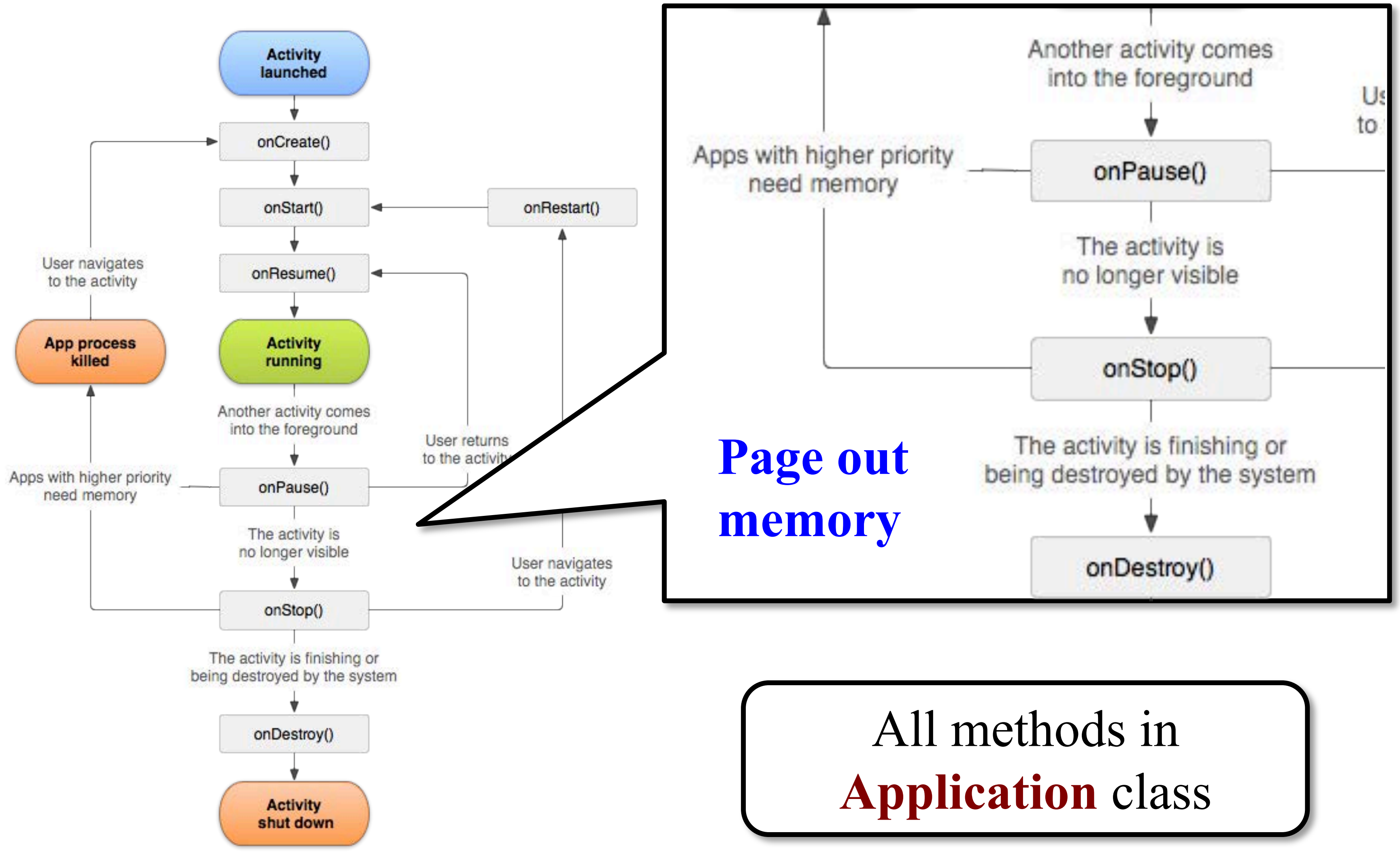


Activity launched
onCreate()
onStart()
onRestart()
User navigates to the activity
onResume()
App process killed
Activity running
Another activity comes into the foreground
User returns to the activity
Apps with higher priority need memory
onPause()
The activity is no longer visible
User navigates to the activity
onStop()
The activity is finishing or being destroyed by the system
onDestroy()
Activity shut down
Activity launched
onCreate()
onStart()
onRestart()
onResume()
Activity running

Reload memory

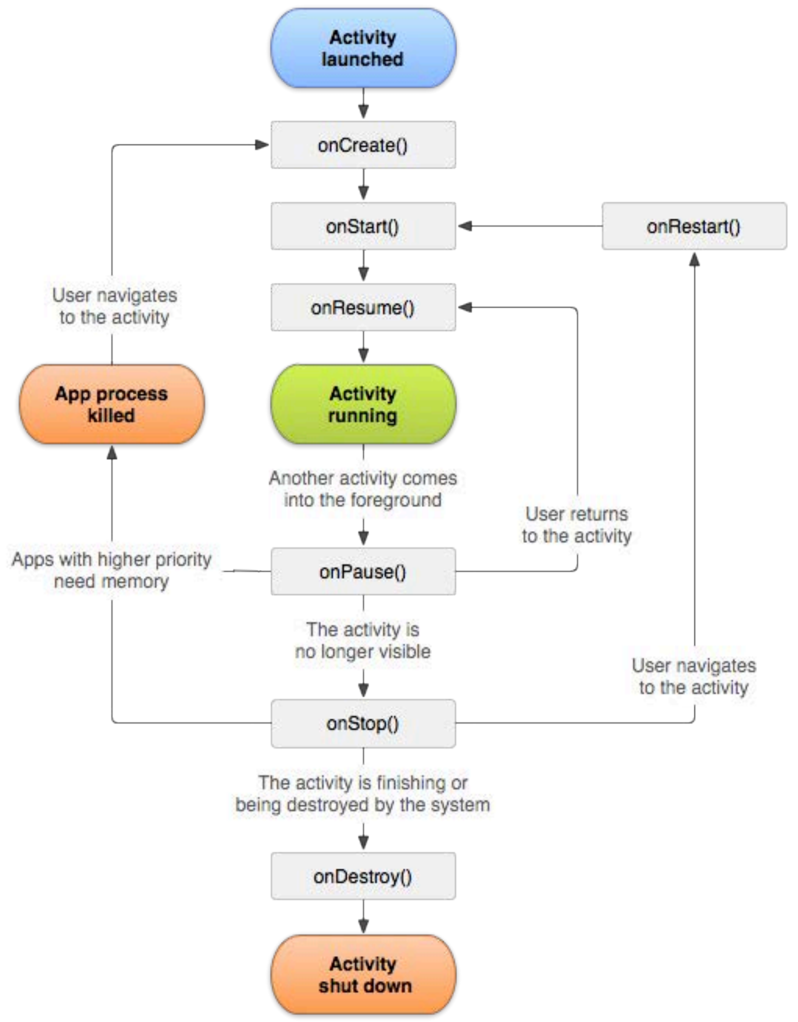
All methods in **Application** class

Android State Handling


Activity launched
onCreate()
onStart()
onRestart()
User navigates to the activity
onResume()
App process killed
Activity running
Another activity comes into the foreground
User returns to the activity
Apps with higher priority need memory
onPause()
The activity is no longer visible
User navigates to the activity
onStop()
The activity is finishing or being destroyed by the system
onDestroy()
Activity shut down
Another activity comes into the foreground
Apps with higher priority need memory
onPause()
The activity is no longer visible
onStop()
The activity is finishing or being destroyed by the system
onDestroy()
Page out memory

All methods in **Application** class

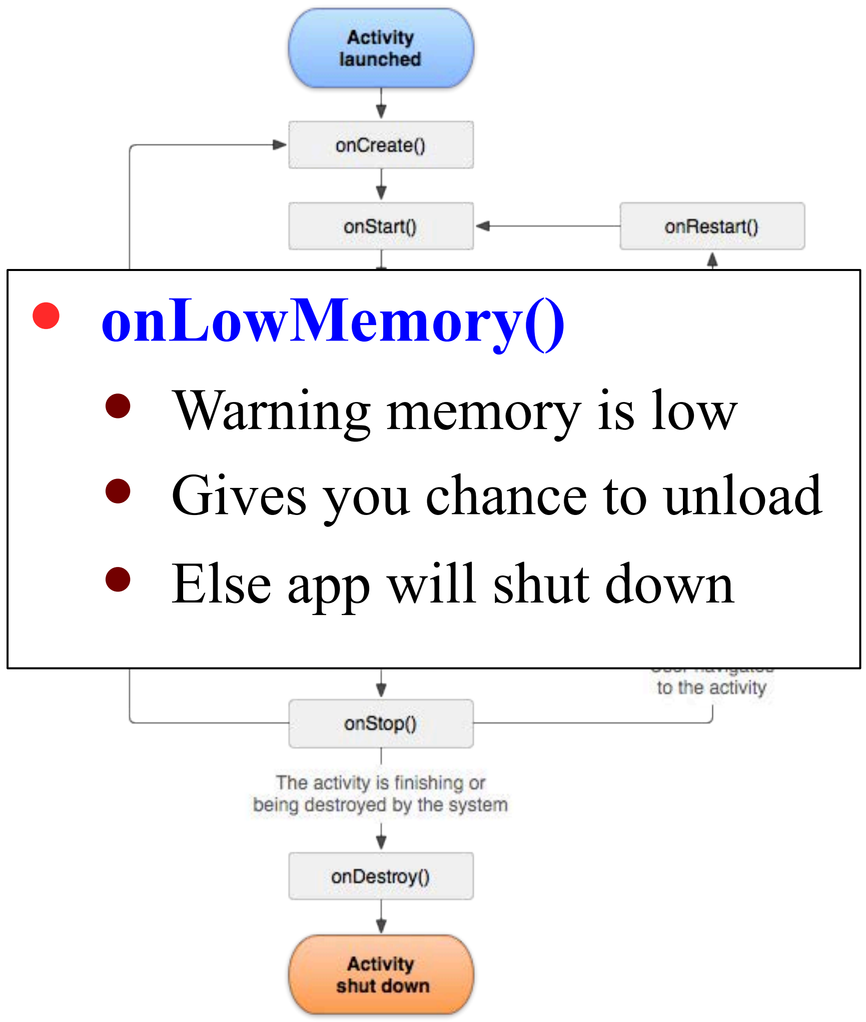
CUGL is Simplified Android Model



Activity launched
onCreate()
onStart()
onRestart()
User navigates to the activity
onResume()
App process killed
Activity running
Another activity comes into the foreground
User returns to the activity
Apps with higher priority need memory
onPause()
The activity is no longer visible
User navigates to the activity
onStop()
The activity is finishing or being destroyed by the system
onDestroy()
Activity shut down

- **onStartup()**
  - Initialized and now active
- **onSuspend()**
  - Sent to background
  - Gives you chance to save
  - Also time to pause music
- **onResume()**
  - Returns to app to active
  - Allows you to restore state
- **onShutdown()**
  - Stopped & memory freed

CUGL is Simplified Android Model



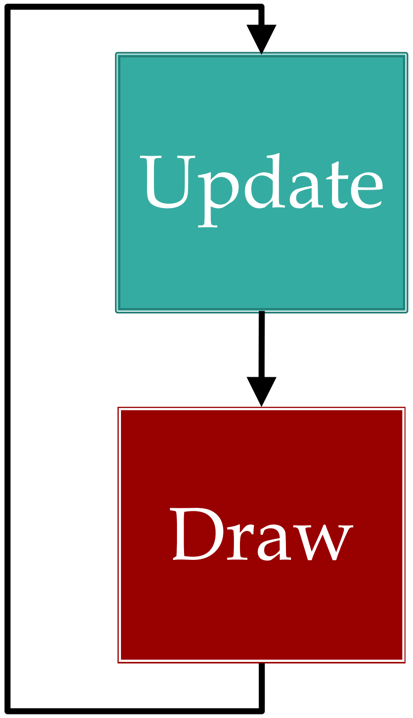
- **onLowMemory()**
  - Warning memory is low
  - Gives you chance to unload
  - Else app will shut down

- **onStartup()**
  - Initialized and now active
- **onSuspend()**
  - Sent to background
  - Gives you chance to save
  - Also time to pause music
- **onResume()**
  - Returns to app to active
  - Allows you to restore state
- **onShutdown()**
  - Stopped & memory freed

Memory Organization and Games

Inter-Frame Memory

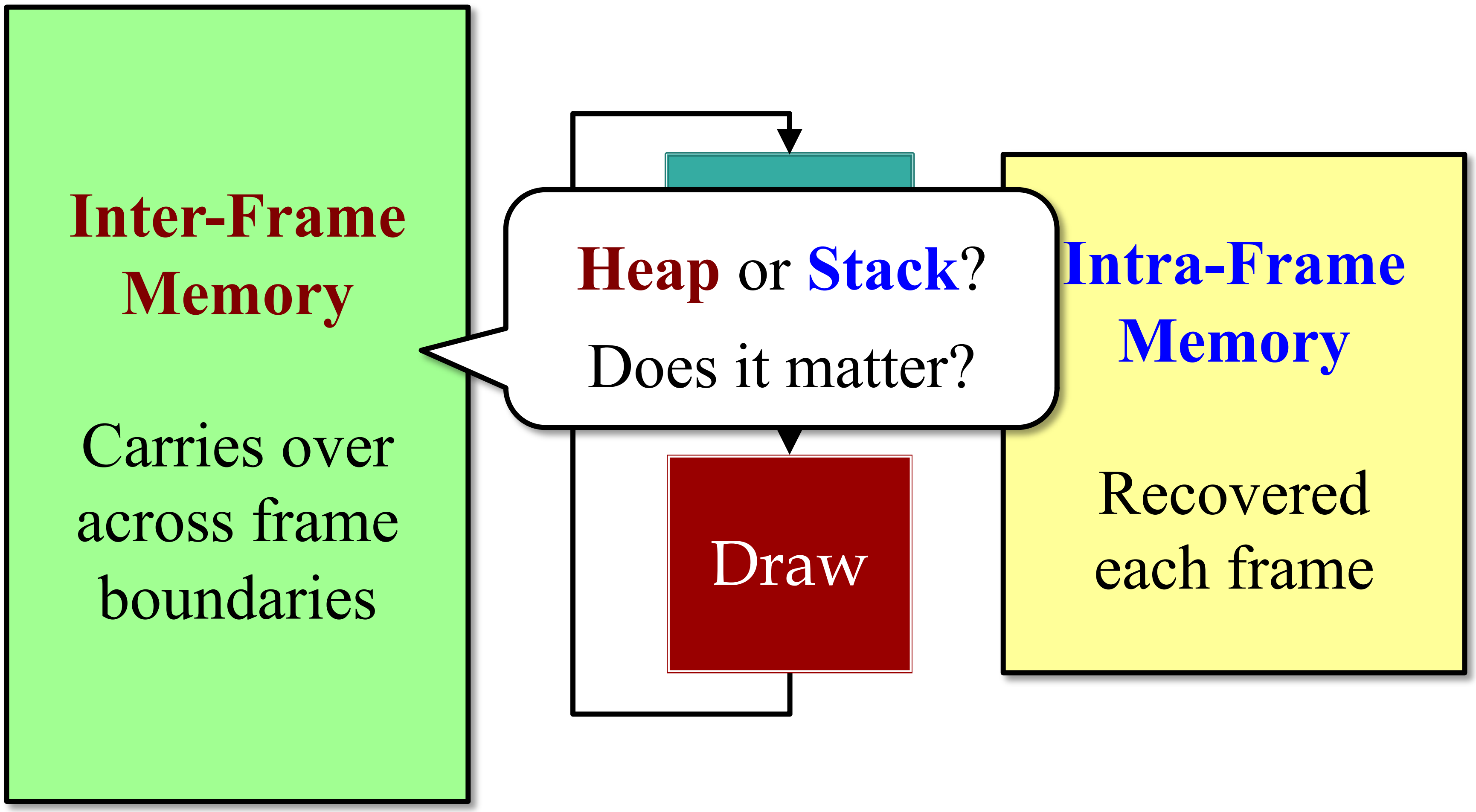
Carries over across frame boundaries



Intra-Frame Memory

Recovered each frame

Memory Organization and Games



Distinguishing Data Types

Intra-Frame

- **Local computation**
  - Local variables (managed by compiler)
  - Temporary objects (not necessarily managed)
- **Transient data structures**
  - Built at the start of update
  - Used to process update
  - Can be deleted at end

Inter-Frame

- **Game state**
  - Model instances
  - Controller state
  - View state and caches
- **Long-term data structures**
  - Built at start/during frame
  - Lasts for multiple frames
  - May adjust to data changes

Distinguishing Data Types

Intra-Frame

- **Local computation**
  - Local variables (ma… er)
  - …jects (not necessarily managed)
- **Transient data structures**
  - Built at the start of update
  - Used to process update
  - Can be deleted at end

Local Variables

Inter-Frame

- **Game state**
  - Model instances
  - C…
  - … and caches
- **Long-term data structures**
  - Built at start/during frame
  - Lasts for multiple frames
  - May adjust to data changes

Object Fields

Distinguishing Data Types

Intra-Frame

- **Local computation**
  - Local variables (ma er)
  - jects (not necessarily managed)
- **Transient data structures**
  - Built at the sta pdate
  - te
  - deleted at end

Local Variables

e.g. Collisions

Inter-Frame

- **Game state**
  - Model instances
  - C
  - nd caches
- **Long-term data structures**
  - Built at start/d frame
  - mes
  - just to data changes

Object Fields

e.g. Pathfinding

Handling Game Memory

Intra-Frame

- Does not need to be paged
  - Drop the latest frame
  - Restart on frame boundary
- Want size reasonably **fixed**
  - Local variables always are
  - Limited # of allocations
  - Limit `new` inside loops
- Often use **custom allocator**
  - GC at frame boundaries

Inter-Frame

- Potential to be paged
  - Defines current game state
  - May just want level start
- Size is more **flexible**
  - No. of objects is variable
  - Subsystems may turn on/off
  - User settings may affect
- **OS allocator** okay, but…
  - Recycle with **free lists**

Handling Game Memory

Intra-Frame

- Does not need to be paged
  - Drop the latest frame
  - Restart on frame boundary
- Want size rea
  - Loca
  - Limit allocations
  - Limit `new` inside loops
- Often use **custom allocator**
  - GC at frame boundaries

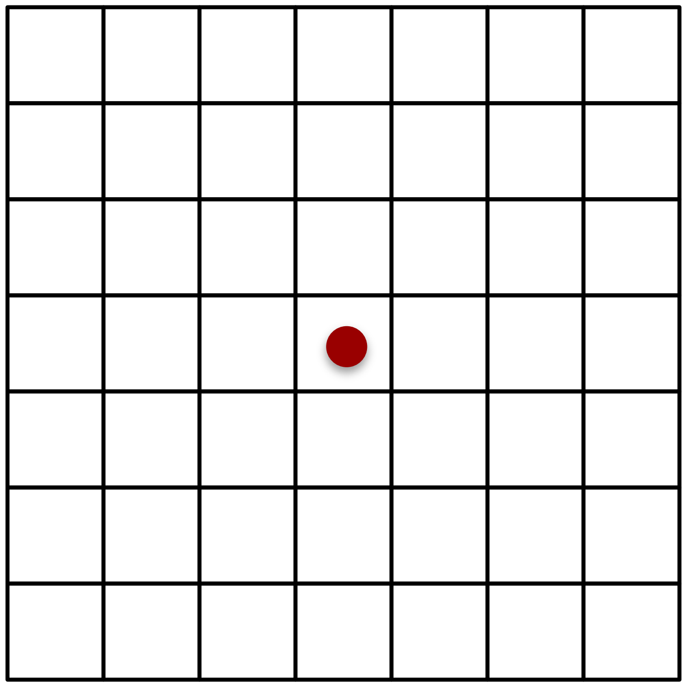
Inter-Frame

- Potential to be paged
  - Defines current game state
  - l start
- **flexible**
  - No. of objects is variable
  - Subsystems may turn on/off
  - User settings may affect
- **OS allocator** okay, but…
  - Recycle with **free lists**

Talked About this in C++ Lesson

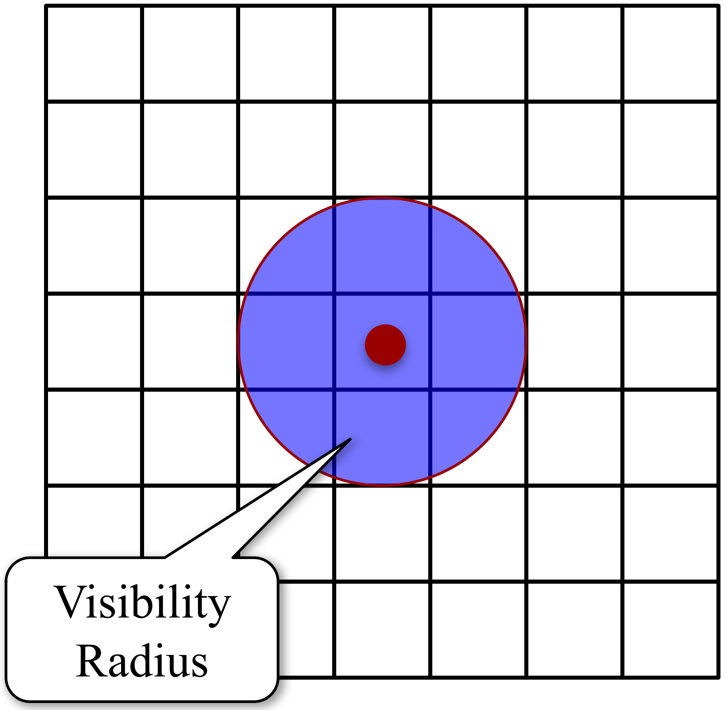
Advanced: Spatial Loading

- Most game data is ***spatial***
  - Only load if player nearby
  - Unload as player moves away
  - Minimizes memory used
- Arrange memory in ***cells***
  - Different from a memory pool
  - Track player visibility radius
  - Load/unload via outer radius
- **Alternative**: loading zones
  - Elevators and "tight spaces"



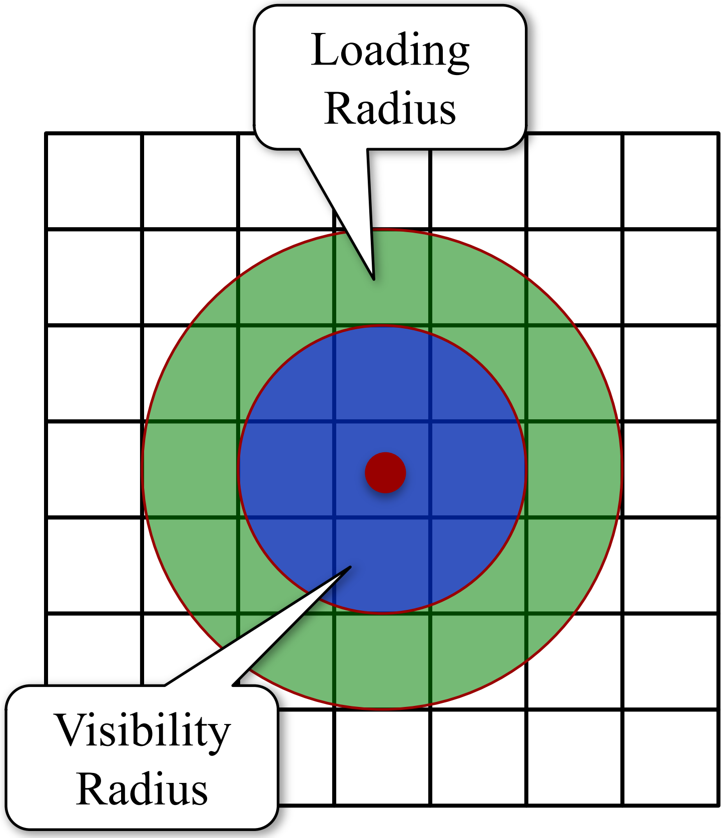
Advanced: Spatial Loading

- Most game data is ***spatial***
  - Only load if player nearby
  - Unload as player moves away
  - Minimizes memory used
- Arrange memory in ***cells***
  - Different from a memory pool
  - Track player visibility radius
  - Load/unload via outer radius
- **Alternative**: loading zones
  - Elevators and "tight spaces"



Advanced: Spatial Loading

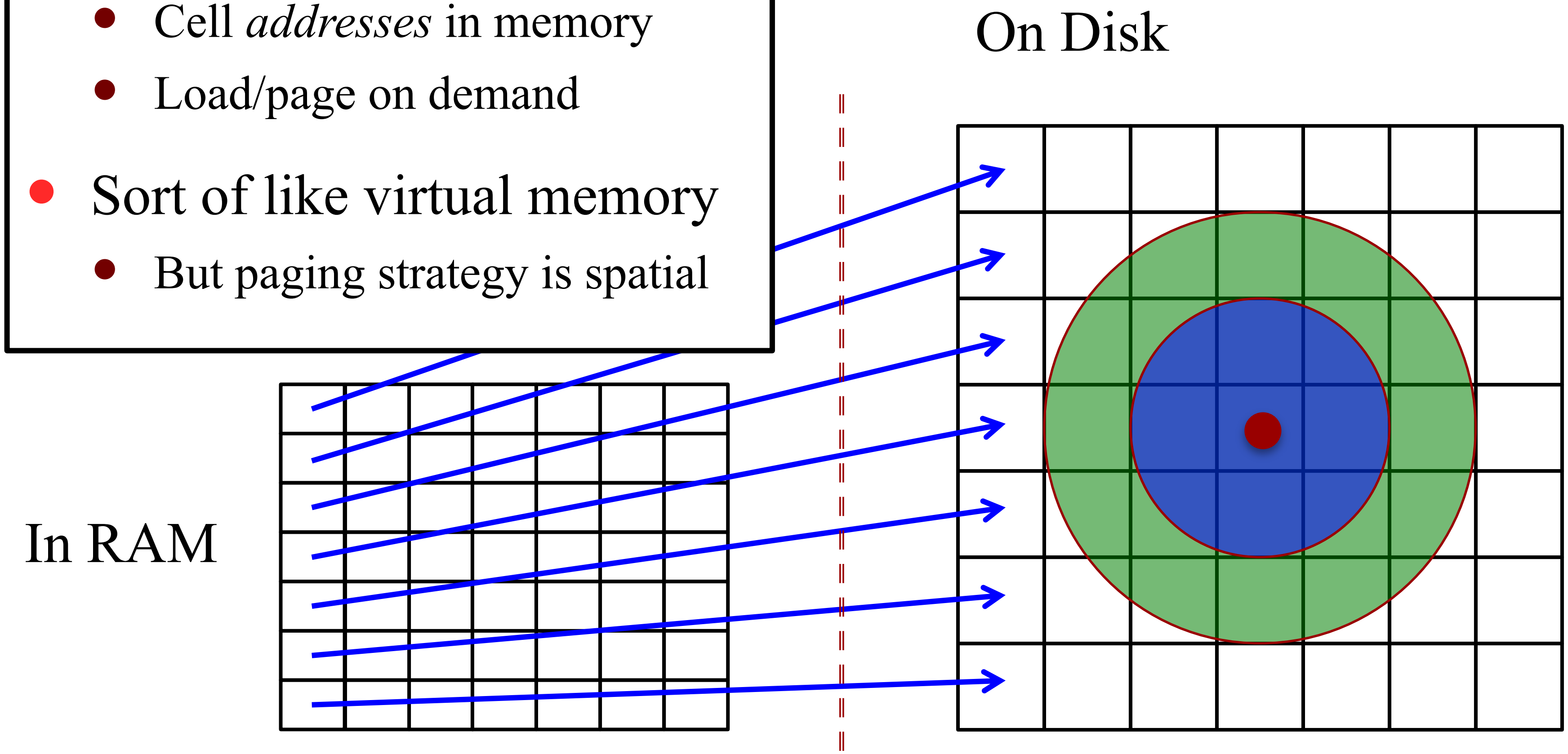
- Most game data is ***spatial***
  - Only load if player nearby
  - Unload as player moves away
  - Minimizes memory used
- Arrange memory in ***cells***
  - Different from a memory pool
  - Track player visibility radius
  - Load/unload via outer radius
- **Alternative**: loading zones
  - Elevators and "tight spaces"



Spatial Loading in *Assassin's Creed*

Implementing Spatial Loading

- Part of serialization model
  - Level/save file has the cells
  - Cell *addresses* in memory
  - Load/page on demand
- Sort of like virtual memory
  - But paging strategy is spatial



Spatial Loading Challenges

- **Not same** as virtual memory
  - Objects unloaded do not exist
  - Do not save state when unload
  - Objects loaded are new created
- Can lead to *unexpected states*
  - "Forgetful" NPCs
  - Creative *Assassin's Creed* kills
- **Workaround**: Global State
  - Track major game conditions
  - **Example**: Guards Alerted
  - Use to load objects in standard, but appropriate, configurations

Summary

- Memory usage is always an issue in games
  - Uncompressed images are quite large
  - Particularly a problem on mobile devices
- CUGL supports modular asset loading
  - Define a custom loader for your asset class
  - Loader has external/main thread components
- Mobile devices must be *monitored*
  - Page out large data when suspended
  - Shut down app when memory is low