Optimizing Your App for Multitasking on iPad in iOS 9

Session 212

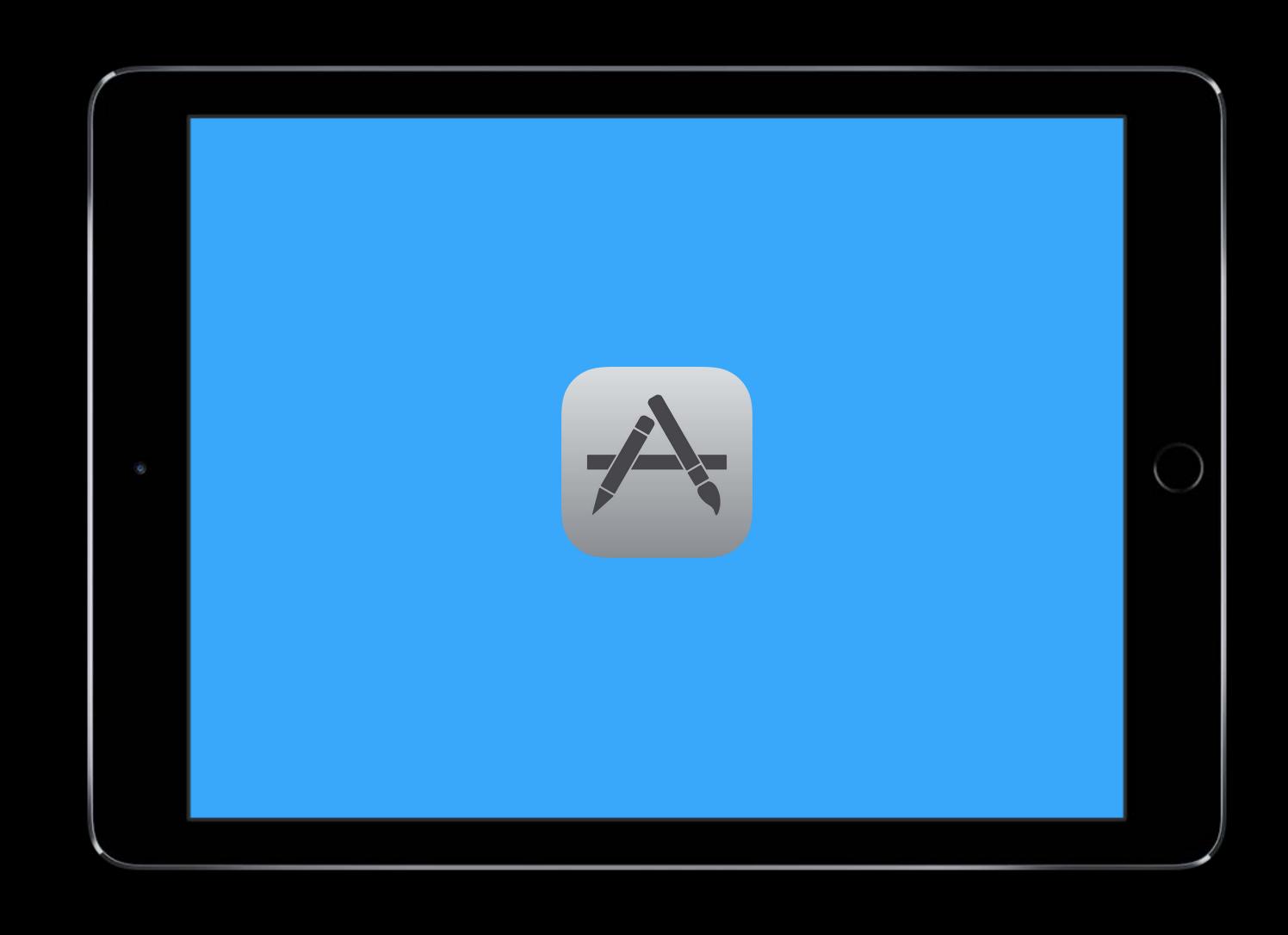
Brittany Paine iOS SpringBoard Engineer Jon Drummond iOS SpringBoard Engineer

Multitasking Sessions

Getting Started with Multitasking on iPad in iOS 9	Presidio	Tuesday 4:30PM
Multitasking Essentials for Media-Based Apps on iPad in iOS 9	Pacific Heights	Wednesday 2:30PM
Optimizing Your App for Multitasking in iOS	Presidio	Wednesday 3:30PM

Multitasking Sessions

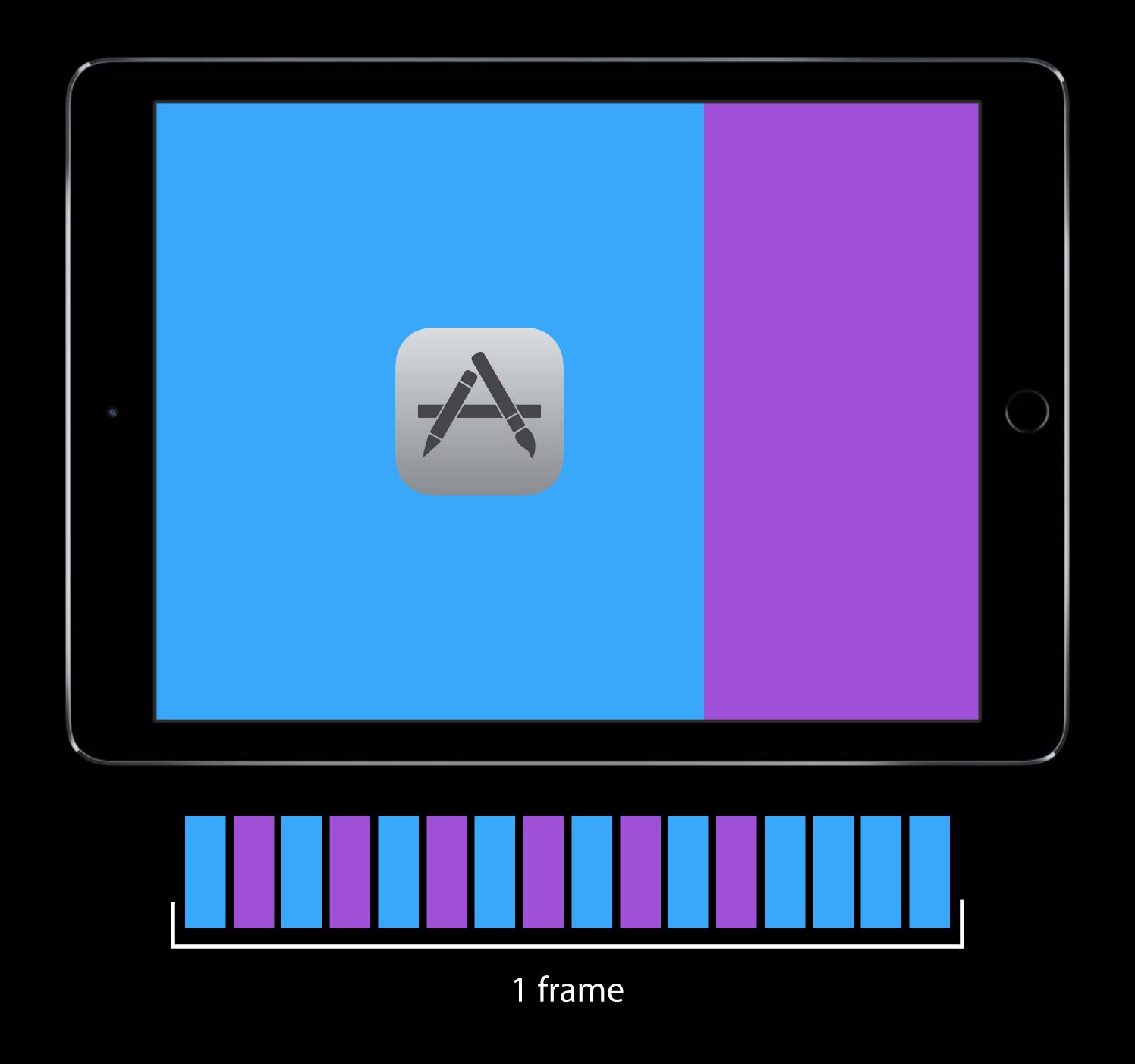
Getting Started with Multitasking on iPad in iOS 9	Presidio	Tuesday 4:30PM
Multitasking Essentials for Media-Based Apps on iPad in iOS 9	Pacific Heights	Wednesday 2:30PM
Optimizing Your App for Multitasking in iOS	Presidio	Wednesday 3:30PM



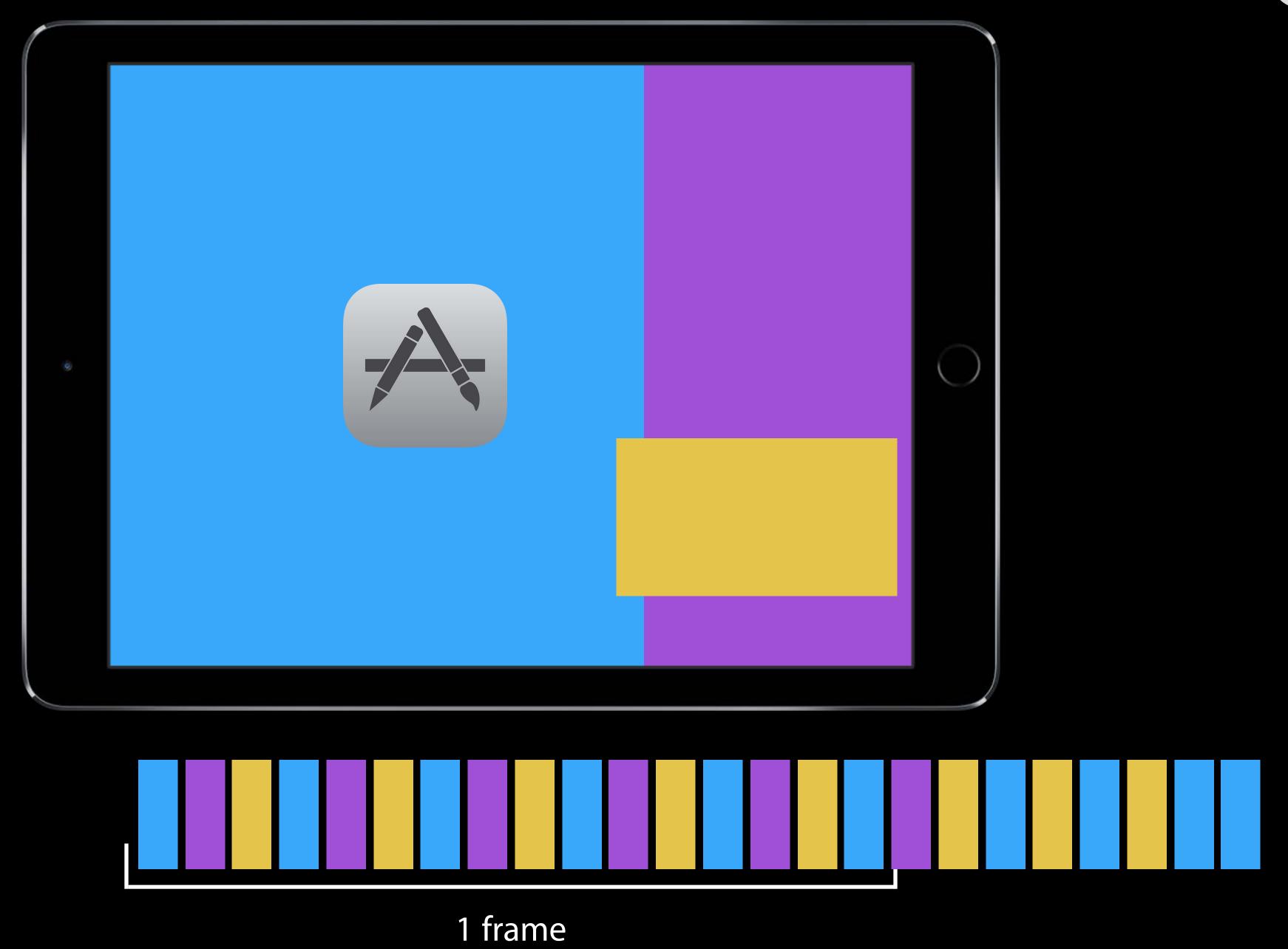








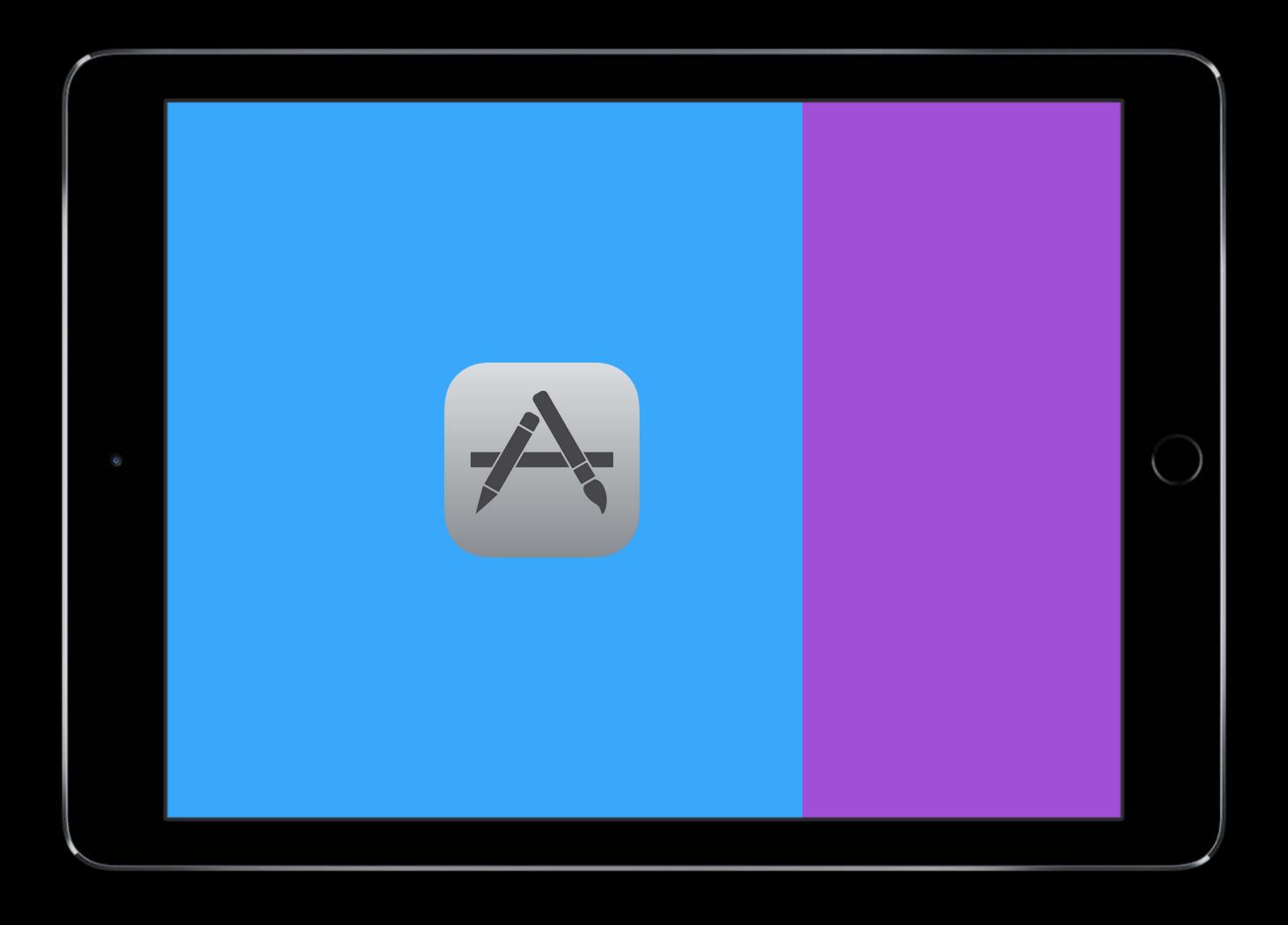






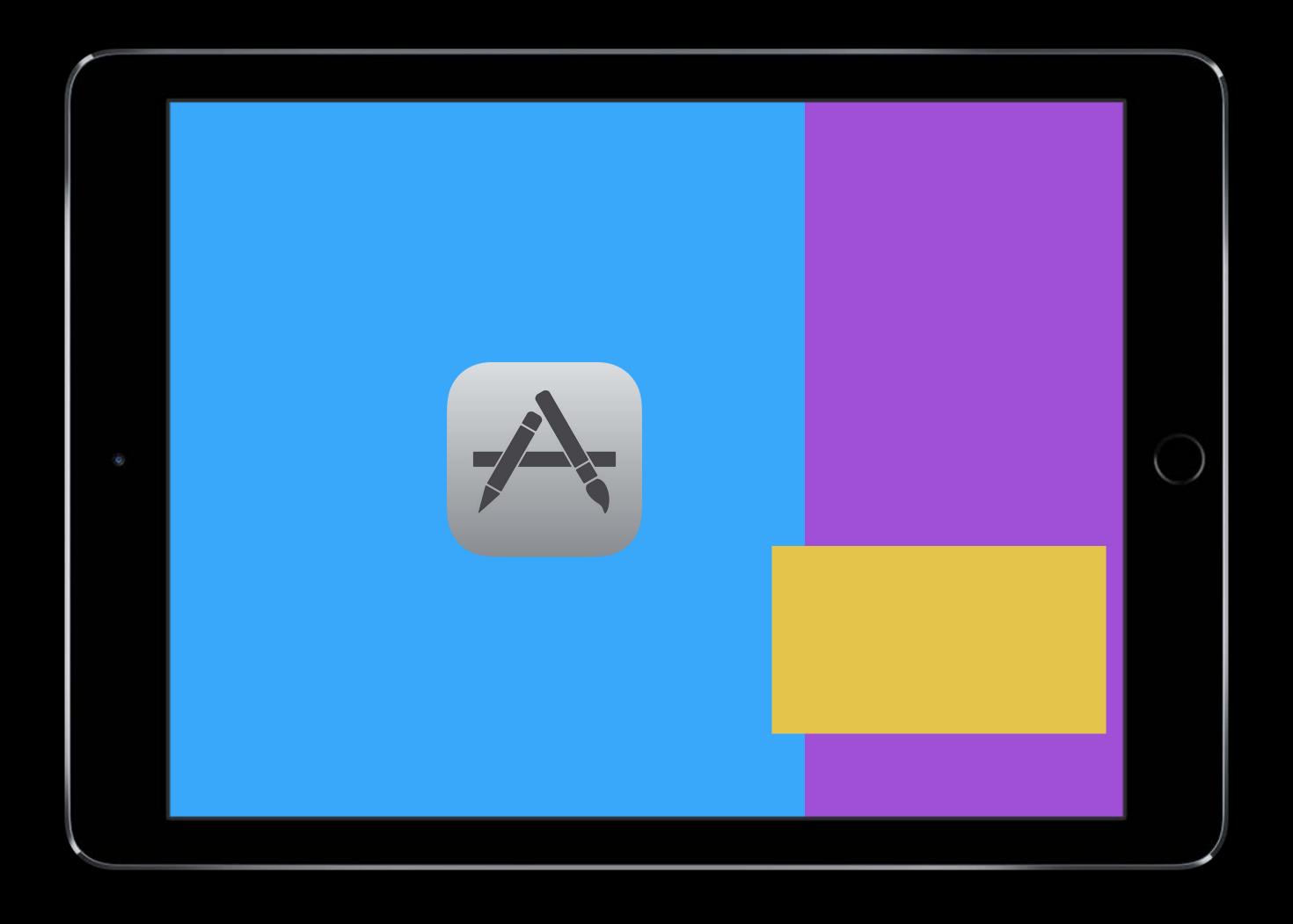
System Primary App Free





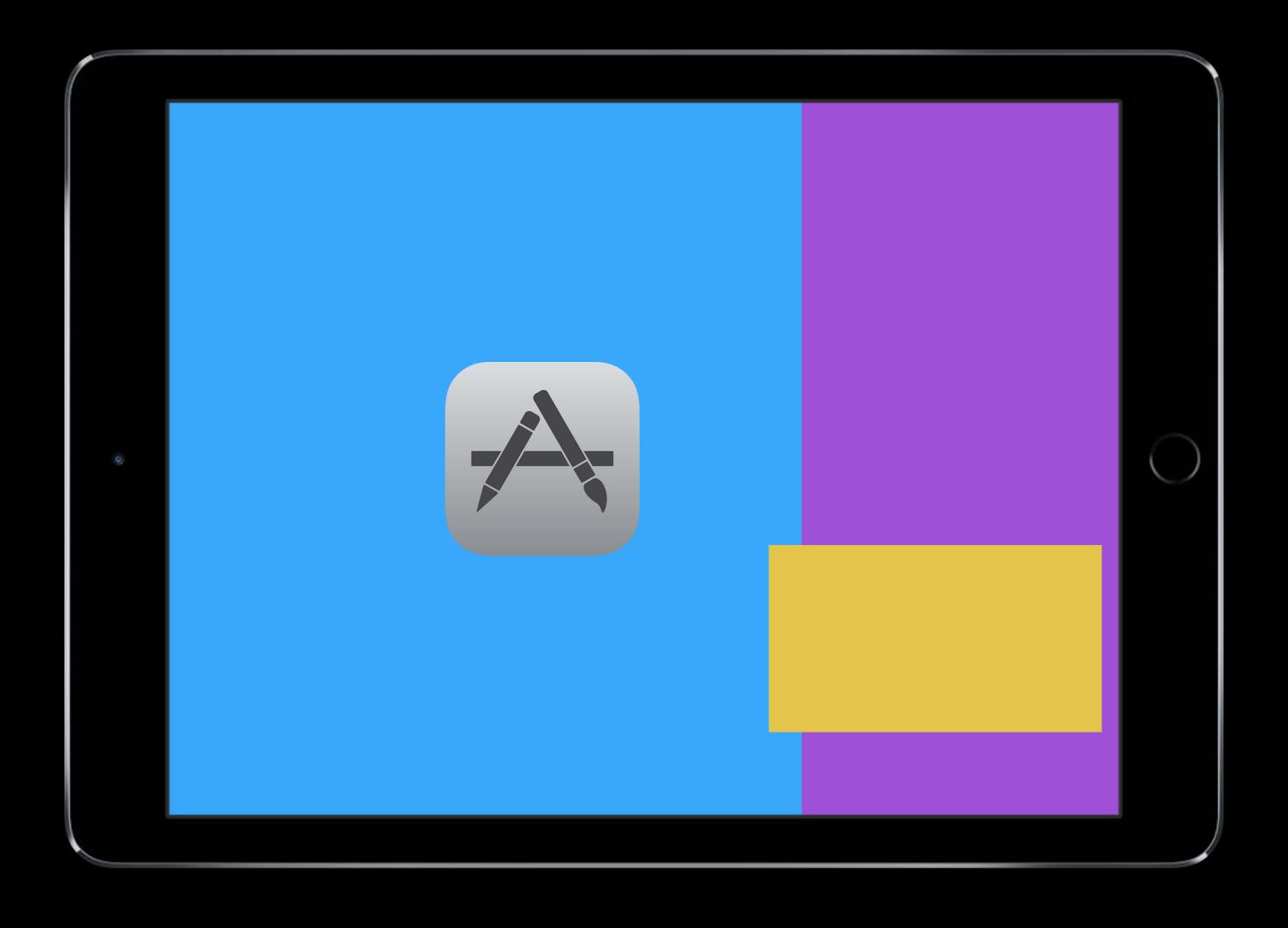
System Primary App Secondary App Free





System Primary App Secondary App PiP



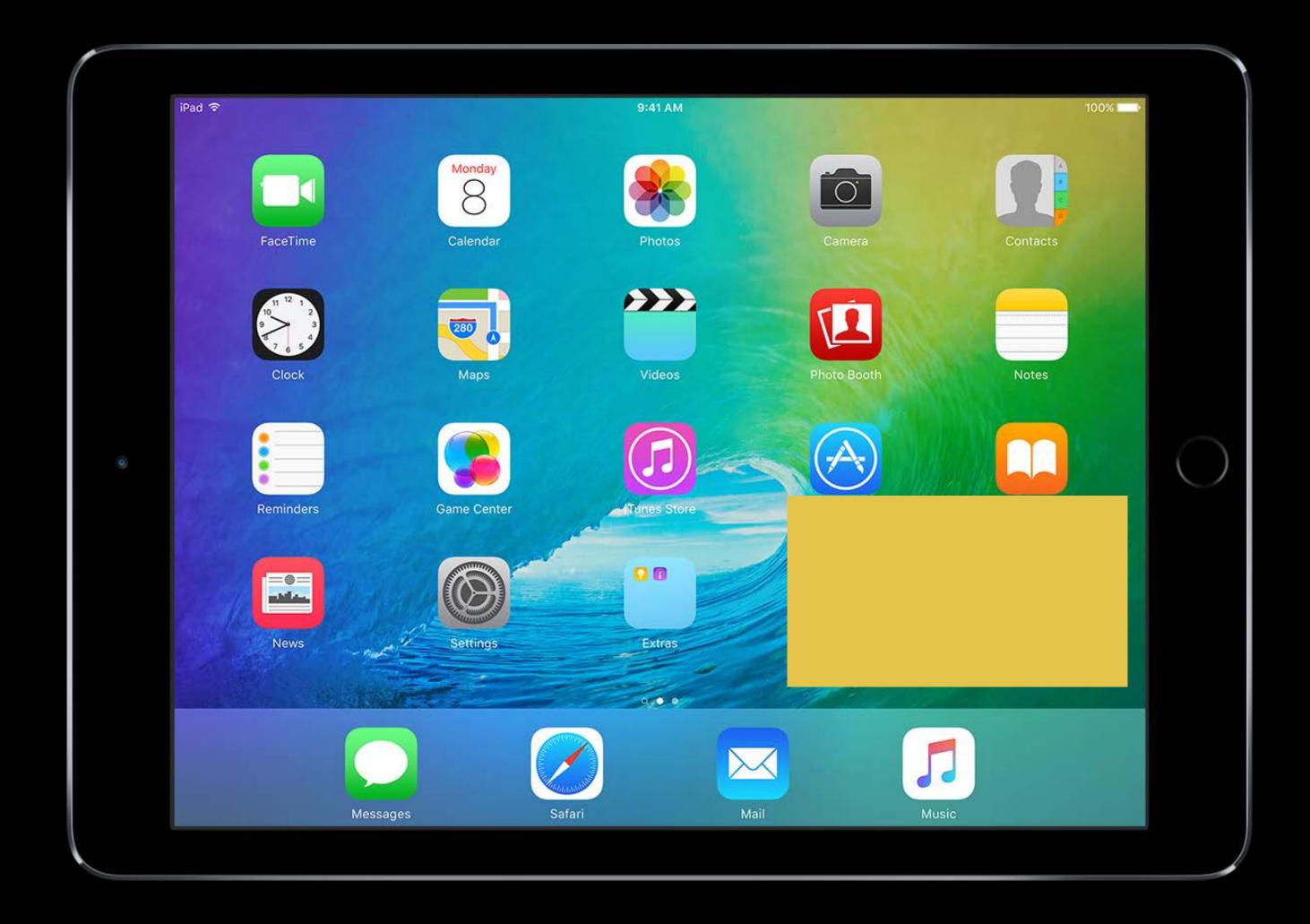


System

Secondary App

PiP





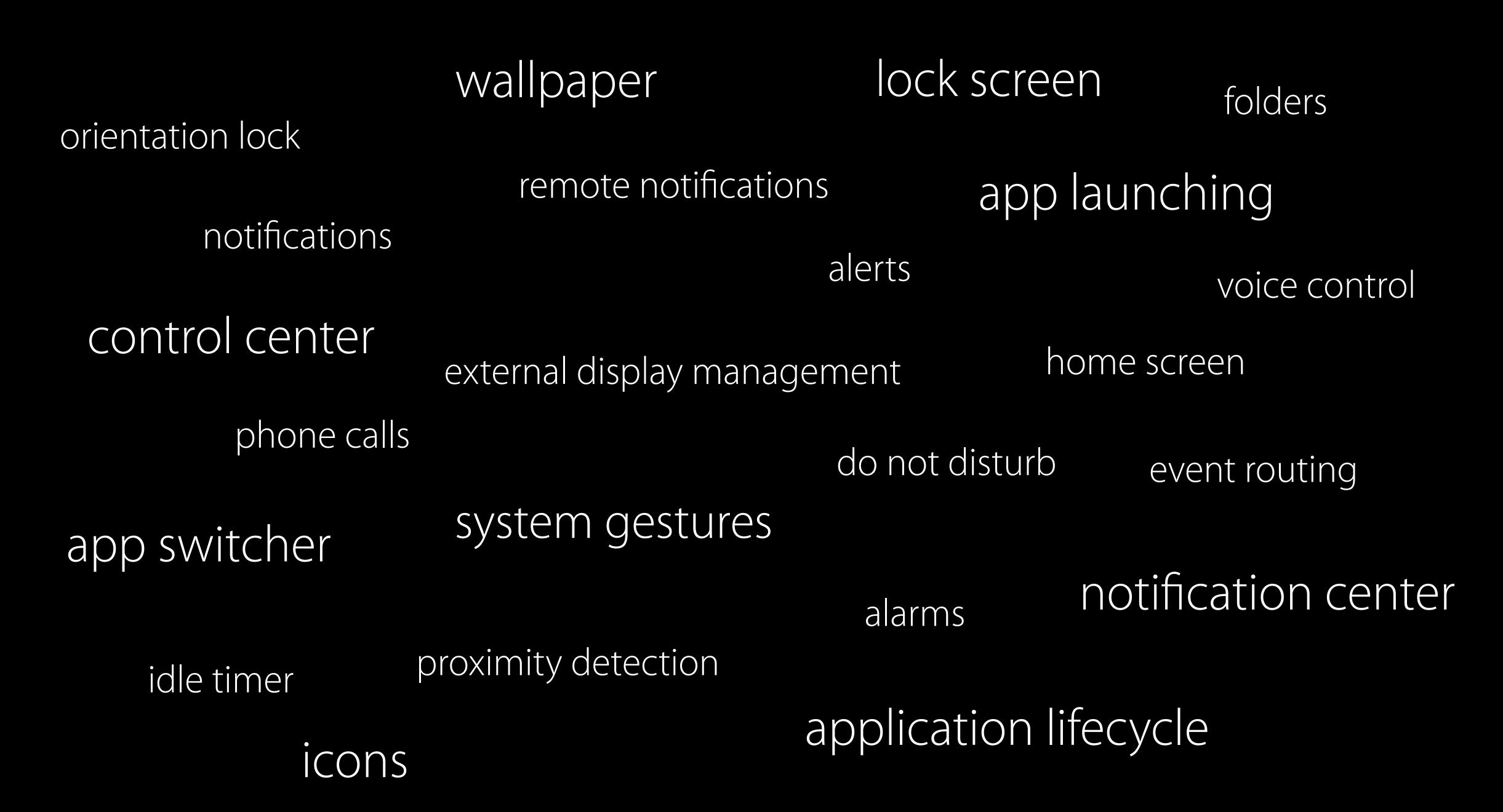
System

Secondary App

PiP

What Is SpringBoard?

What Is SpringBoard?

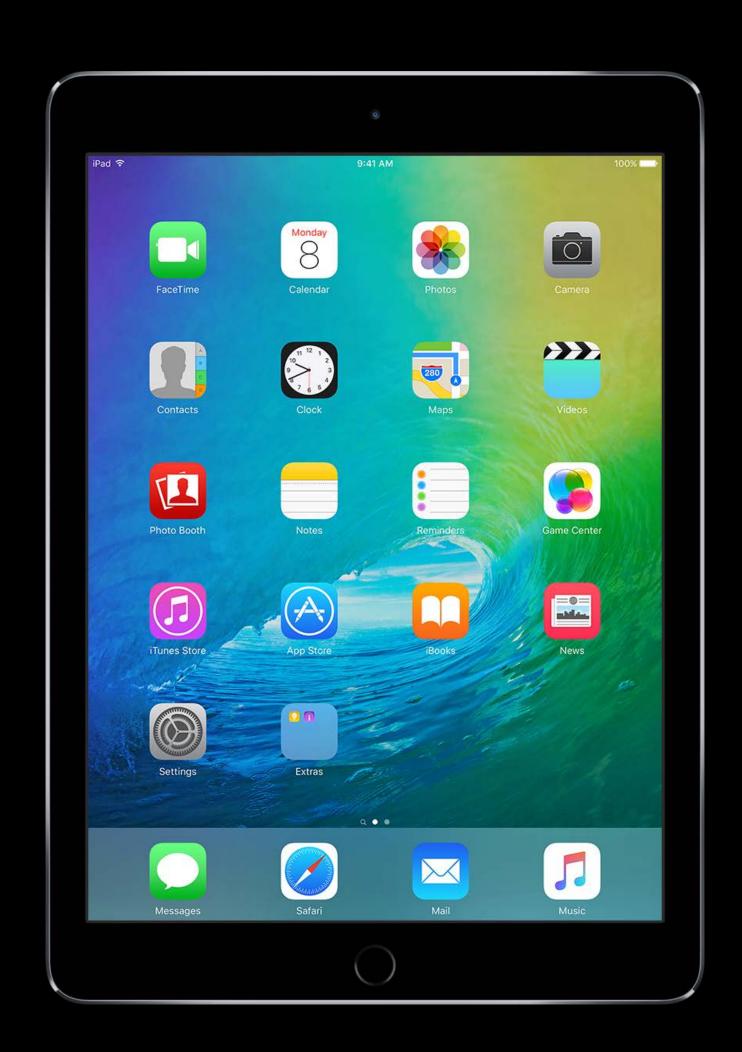


What Is SpringBoard?

UlApplication



The original multitasking app



The original multitasking app

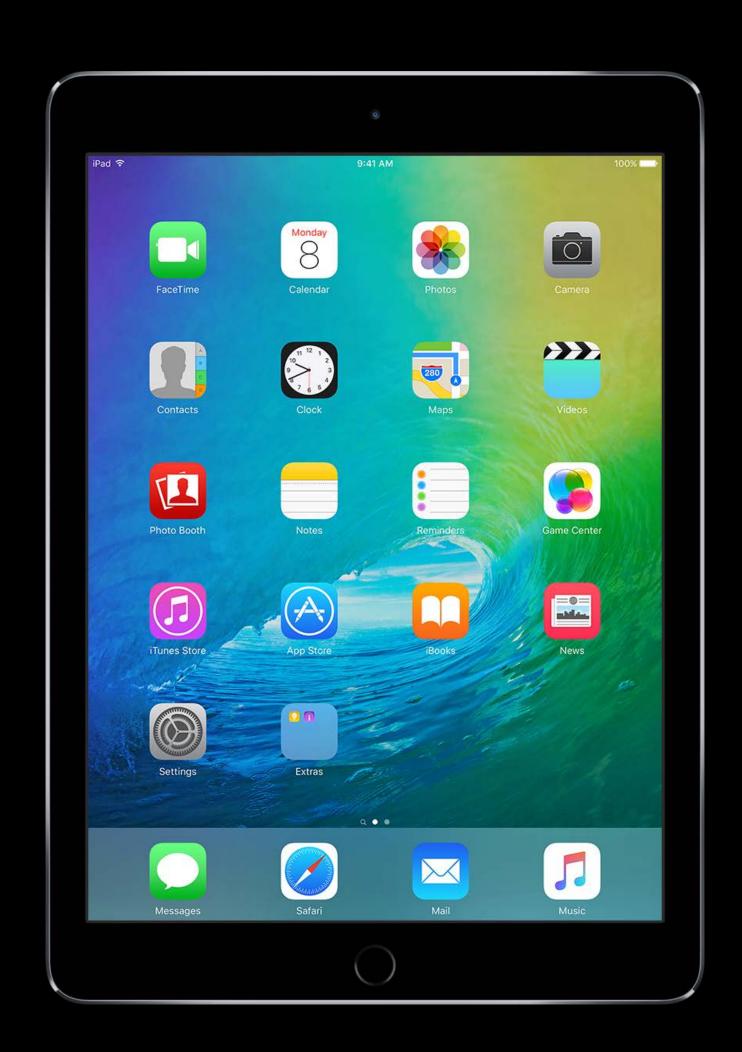
Faced the same multitasking challenges as you do now



The original multitasking app

Faced the same multitasking challenges as you do now

Learned a lot of lessons along the way



The easy stuff

The easy stuff

Fix memory leaks

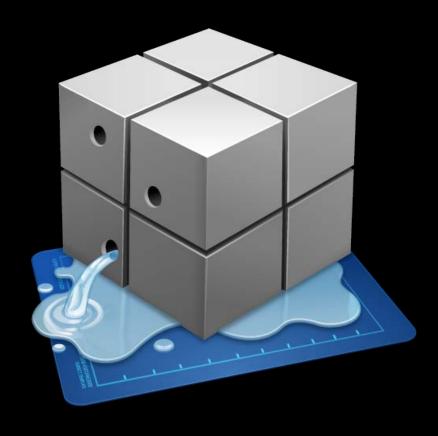


The easy stuff

Fix memory leaks

Fix retain cycles and unbounded memory growth





The easy stuff

Fix memory leaks

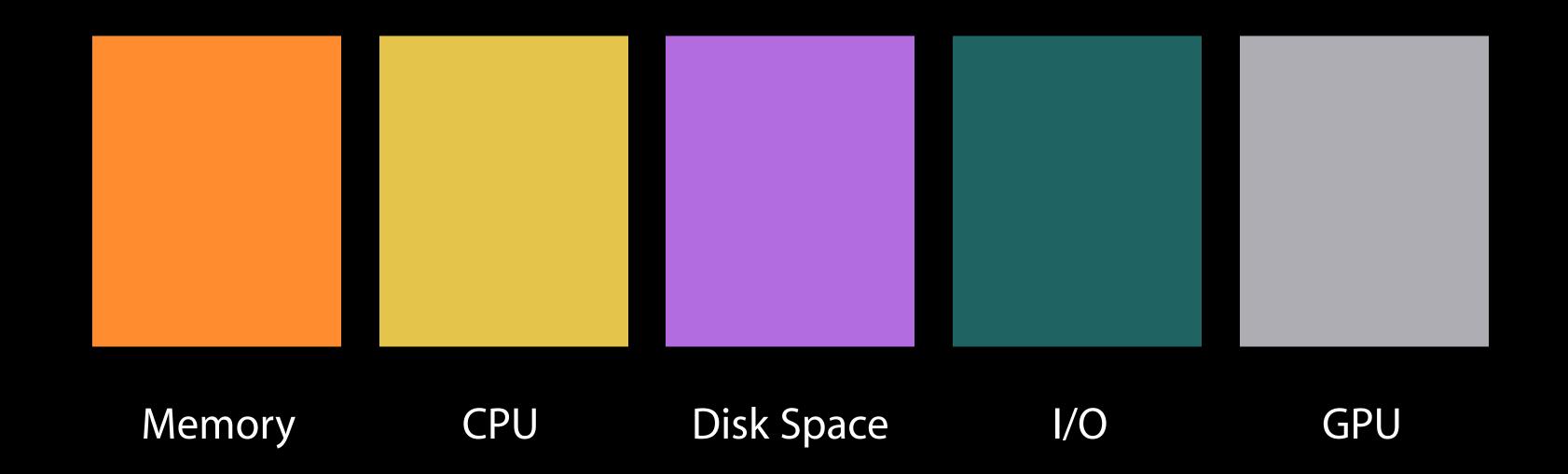
Fix retain cycles and unbounded memory growth

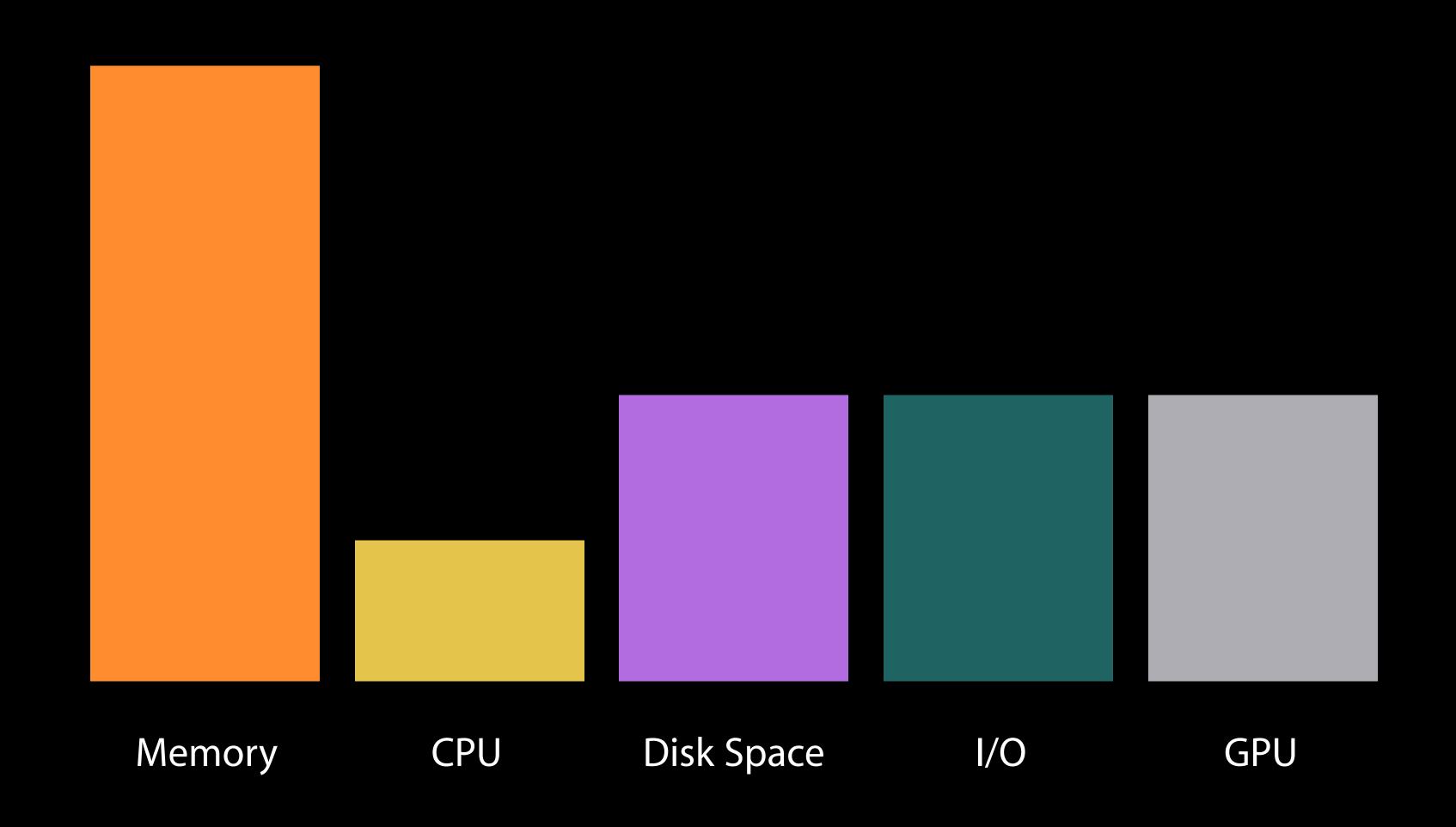
Fix inefficient algorithms

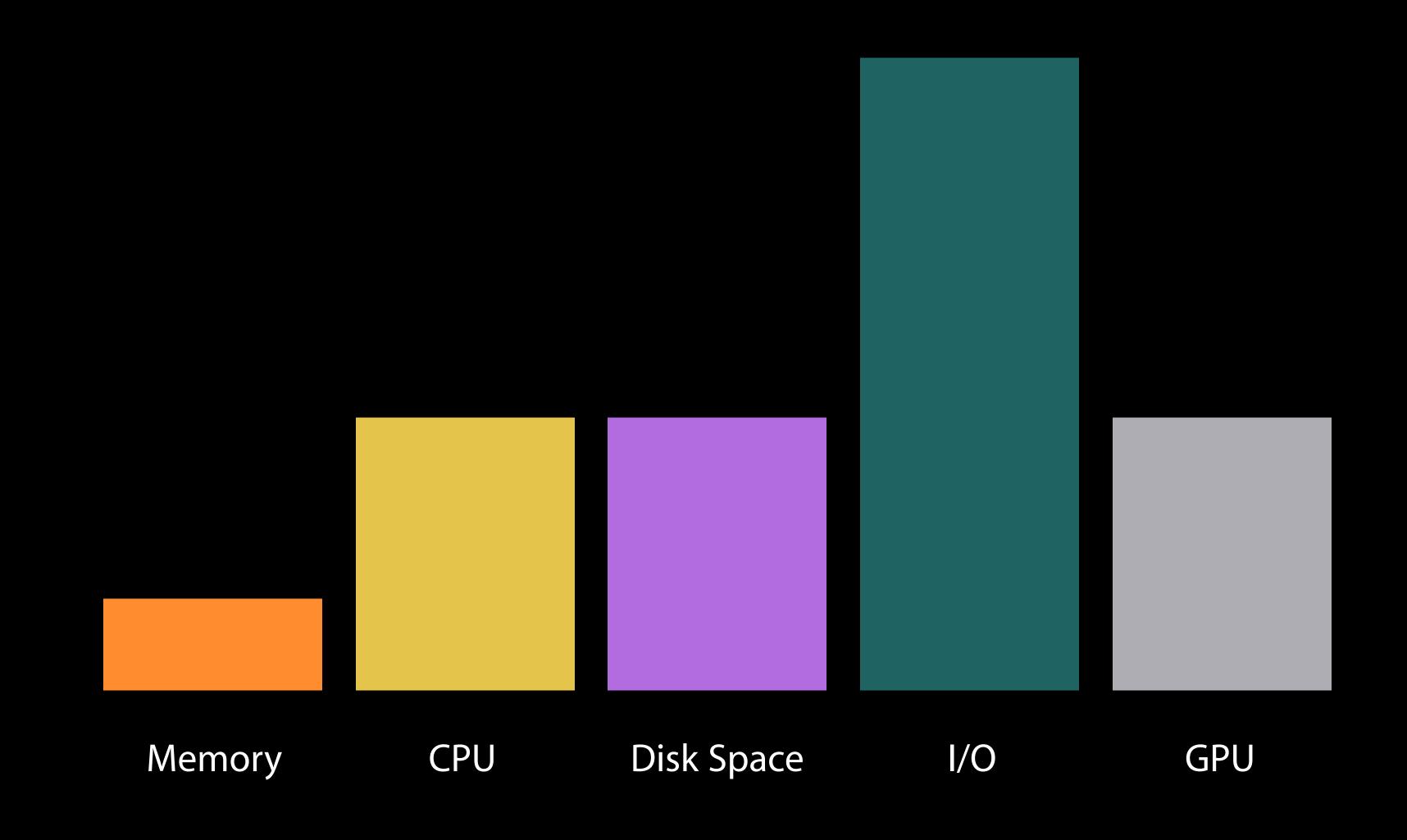


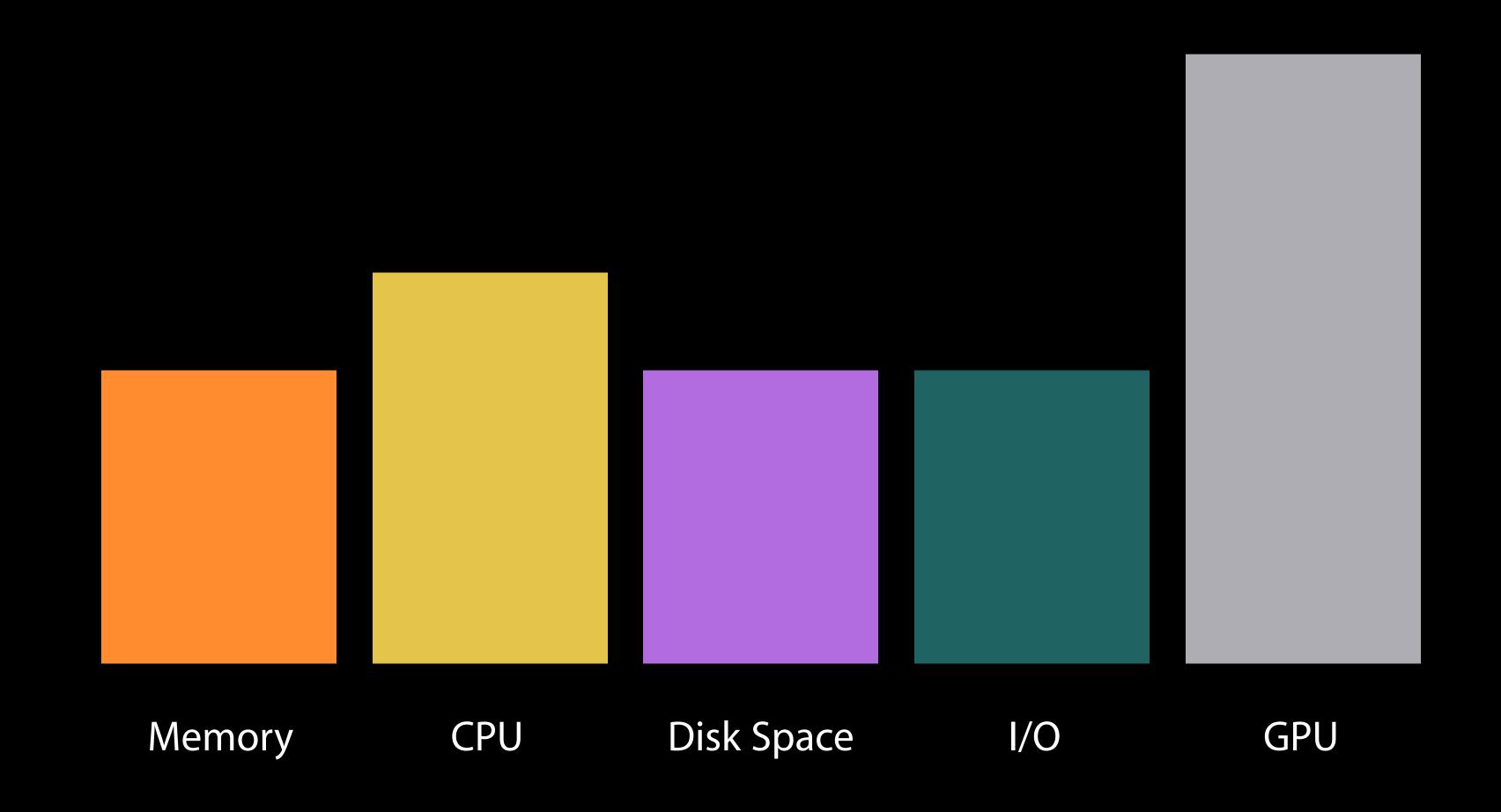


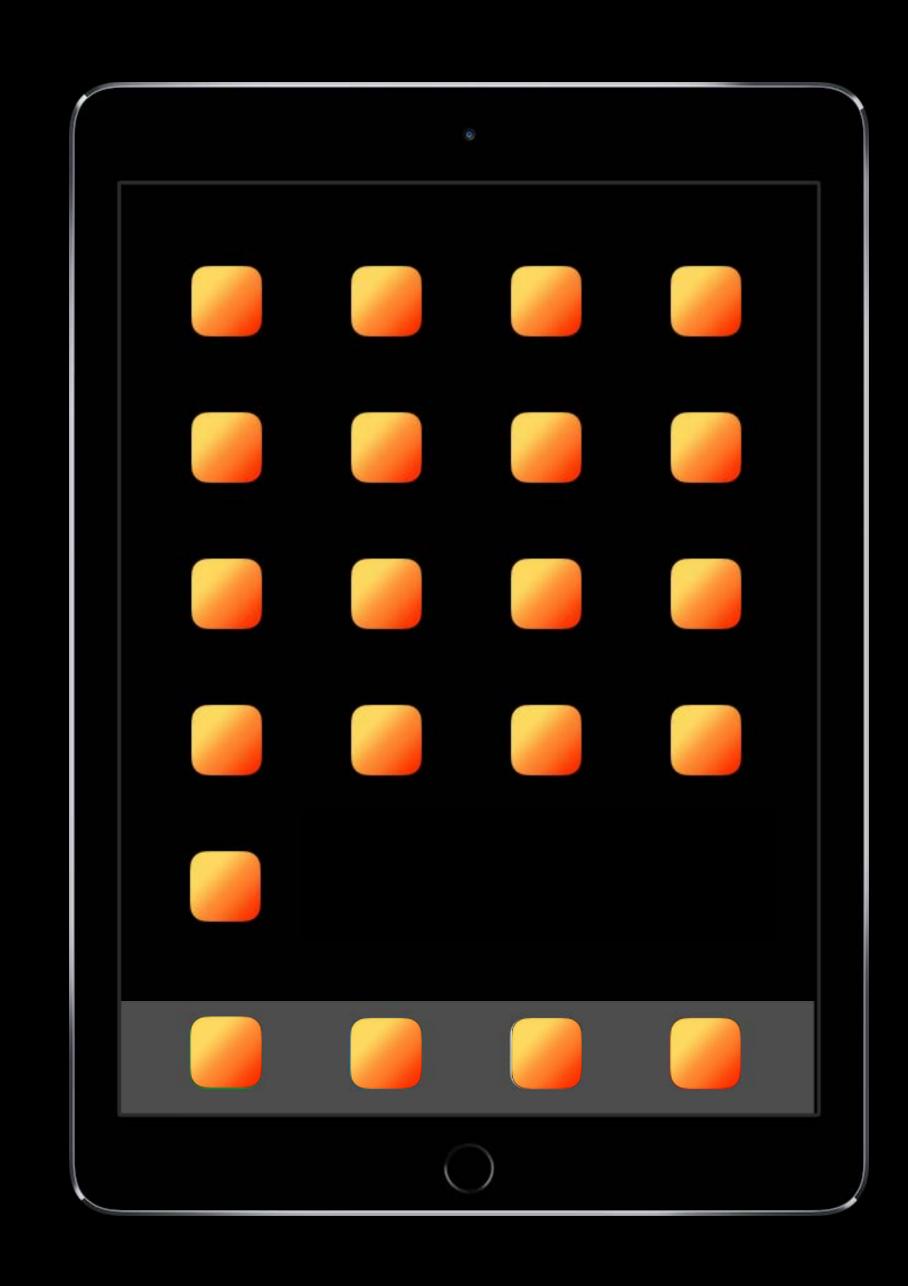


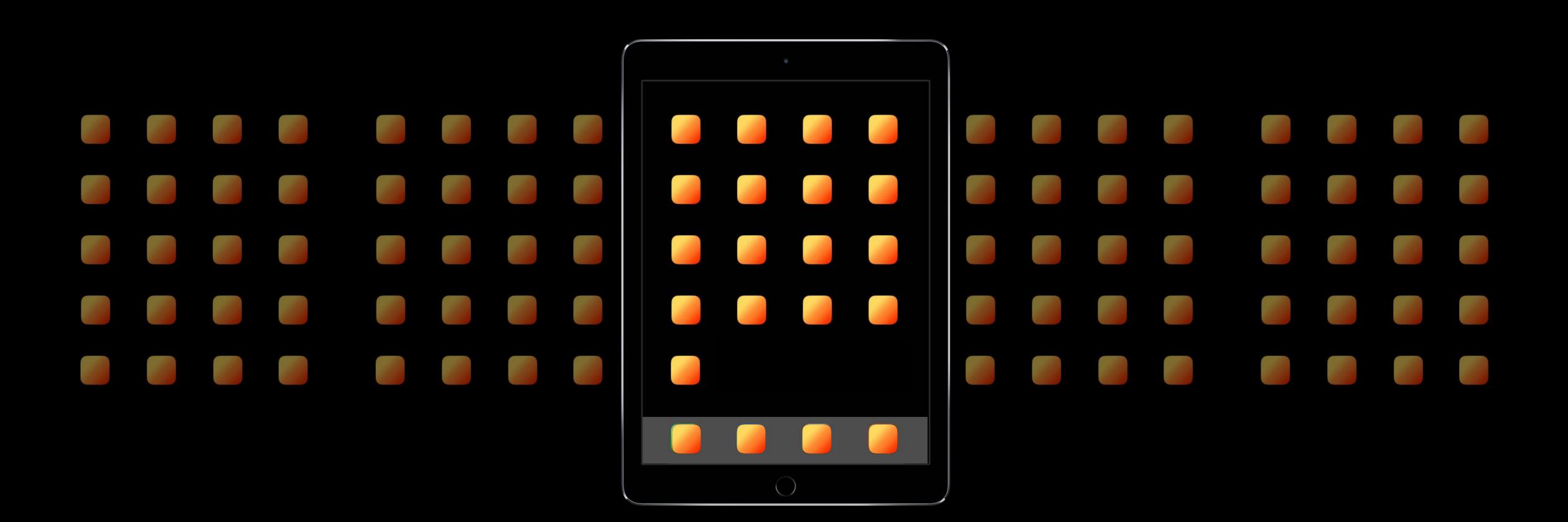


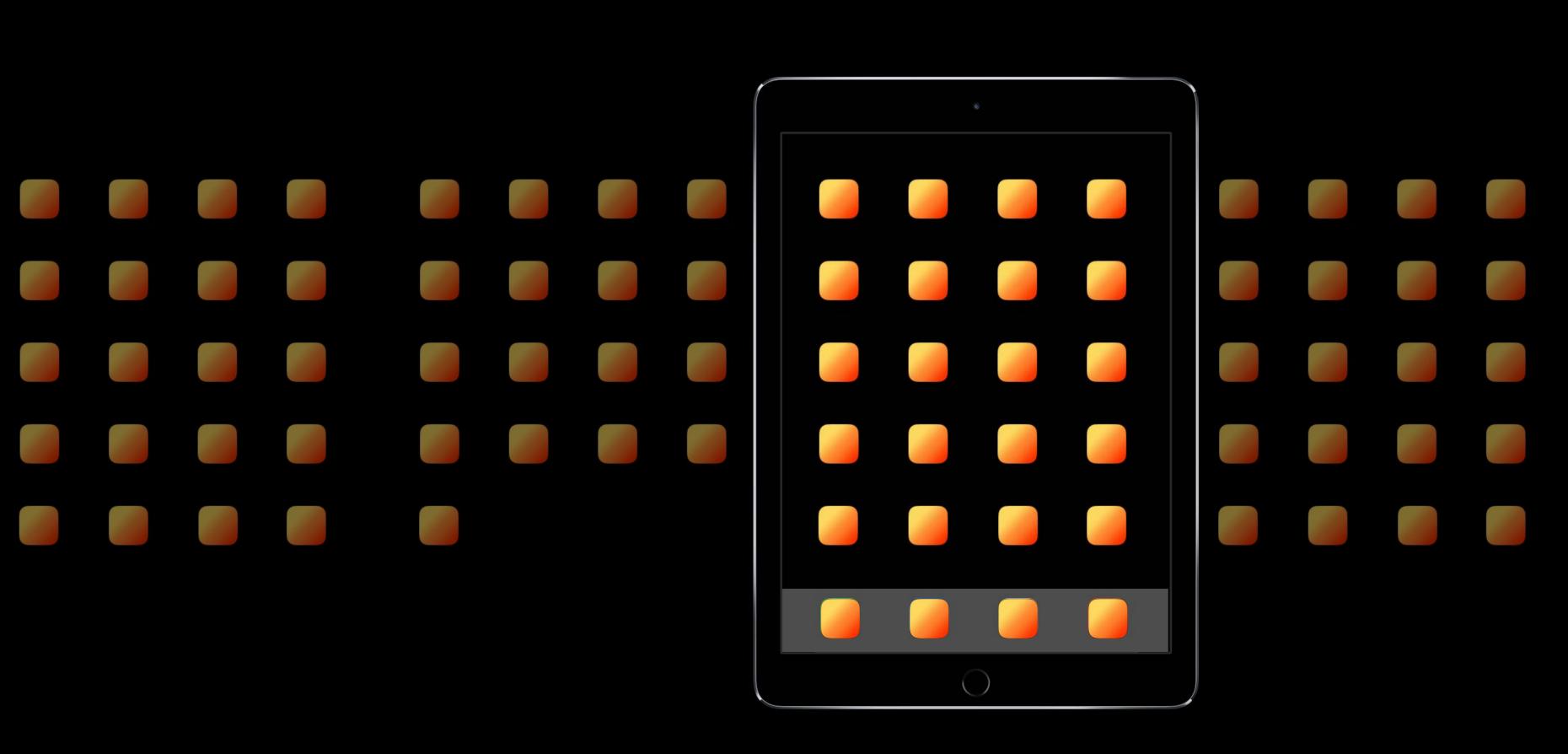


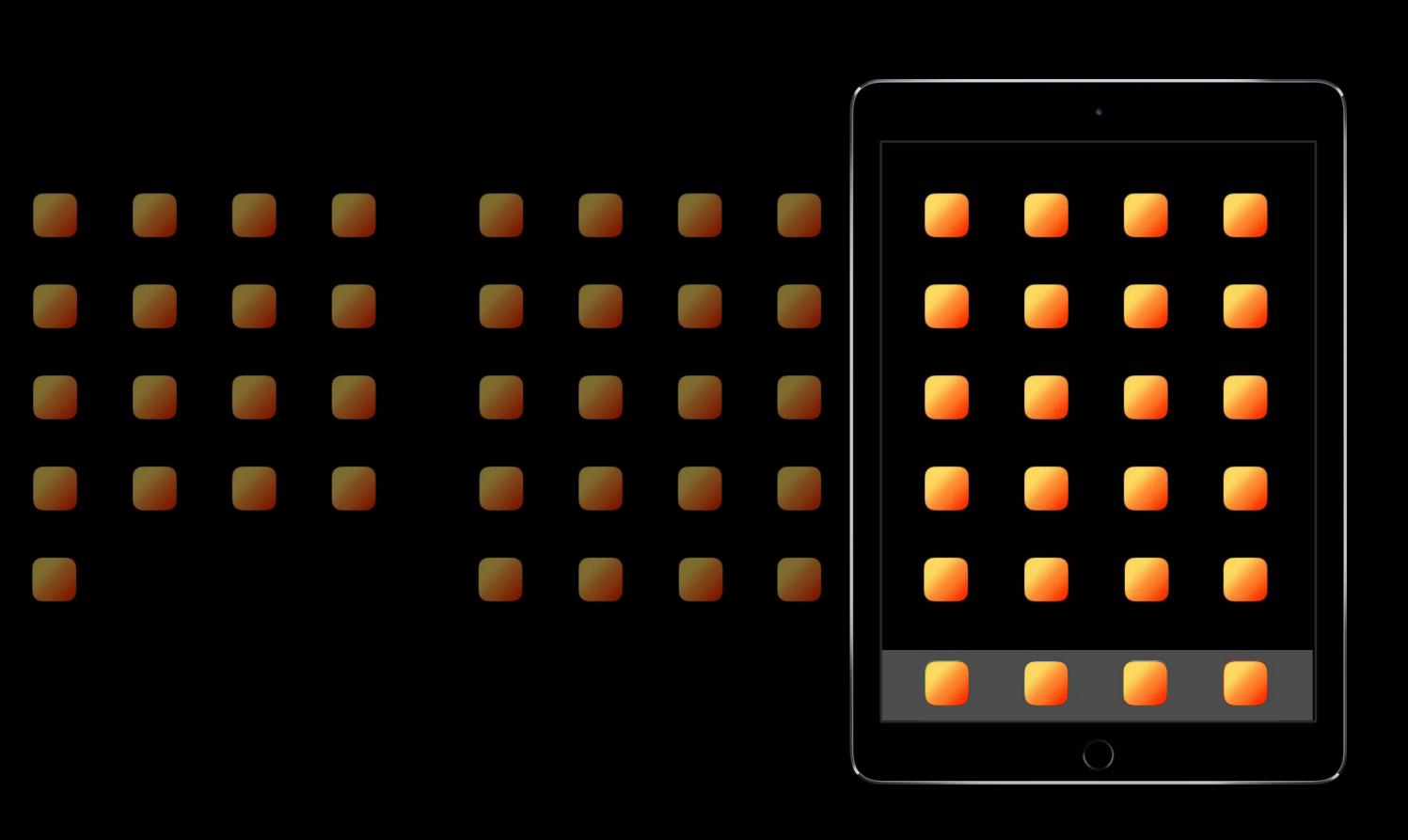


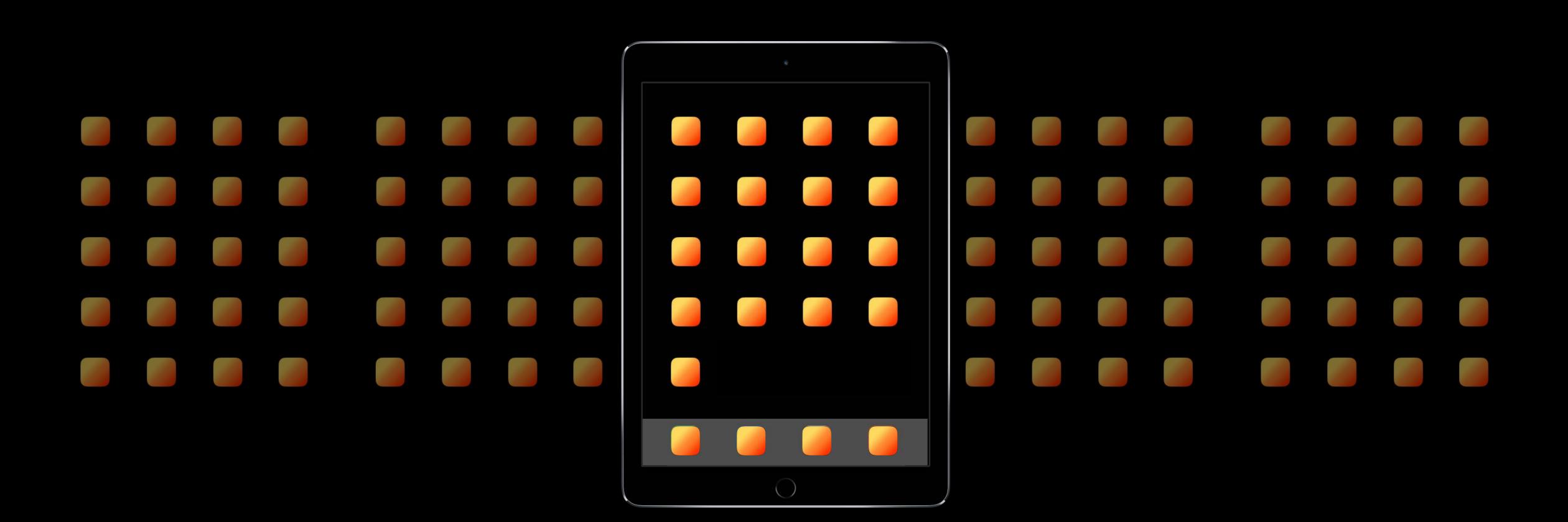


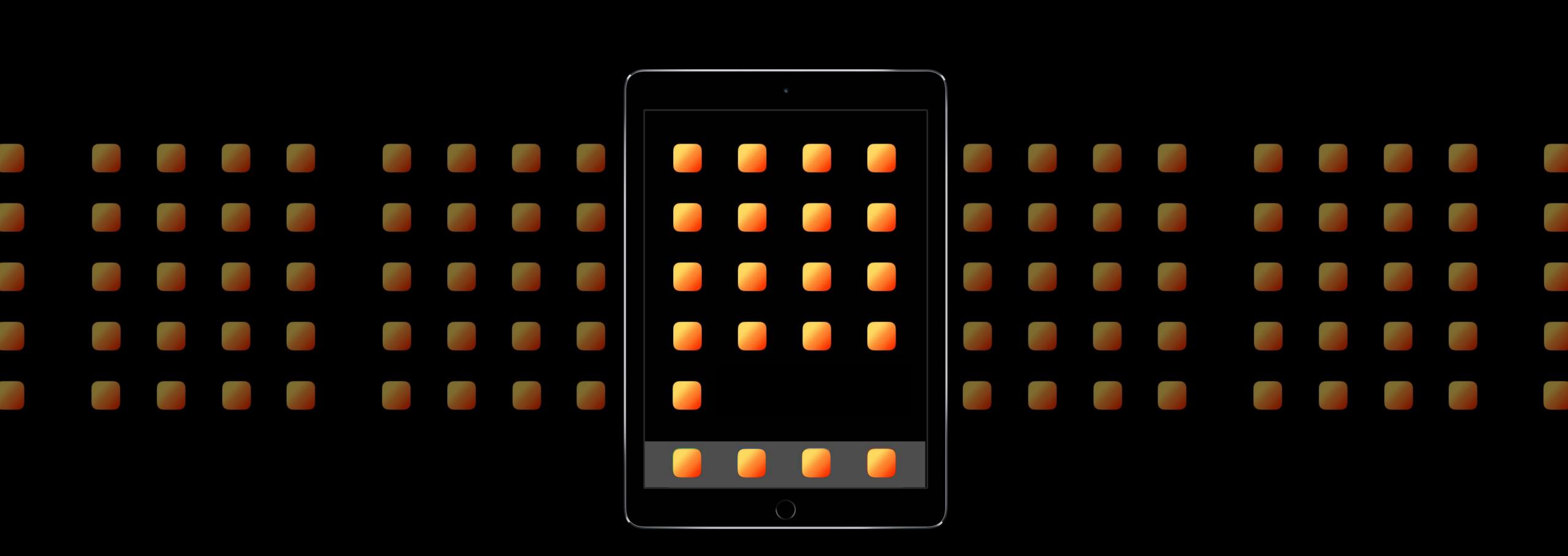


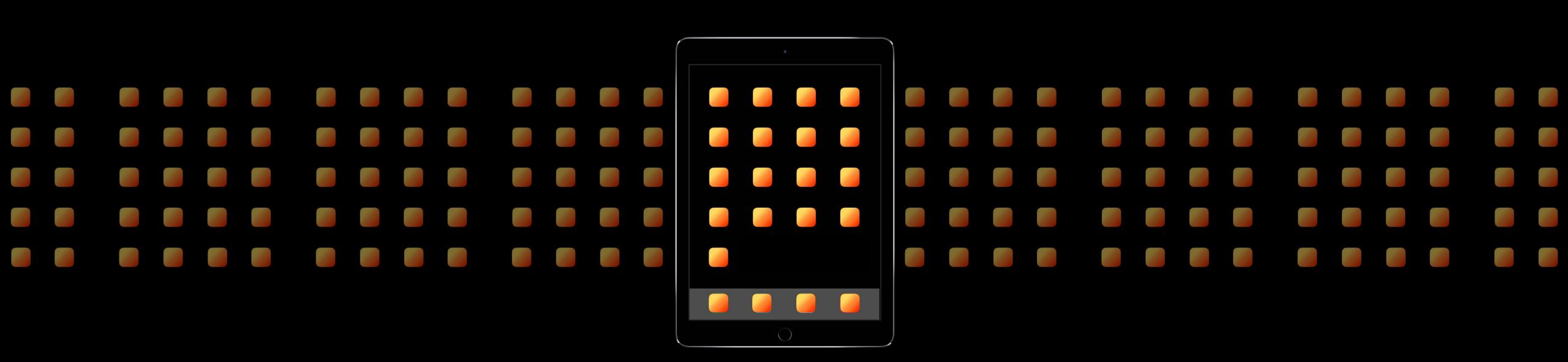


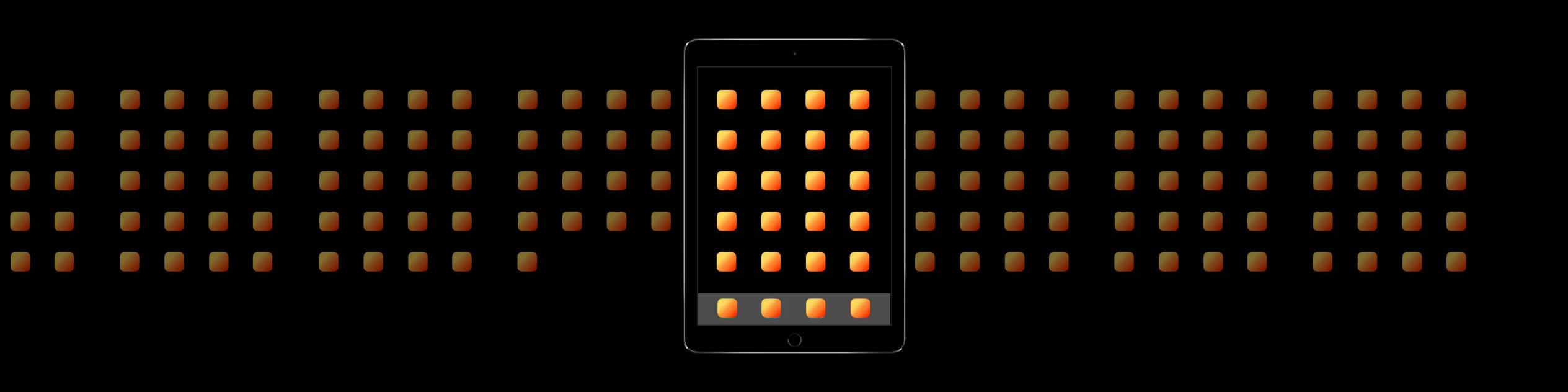


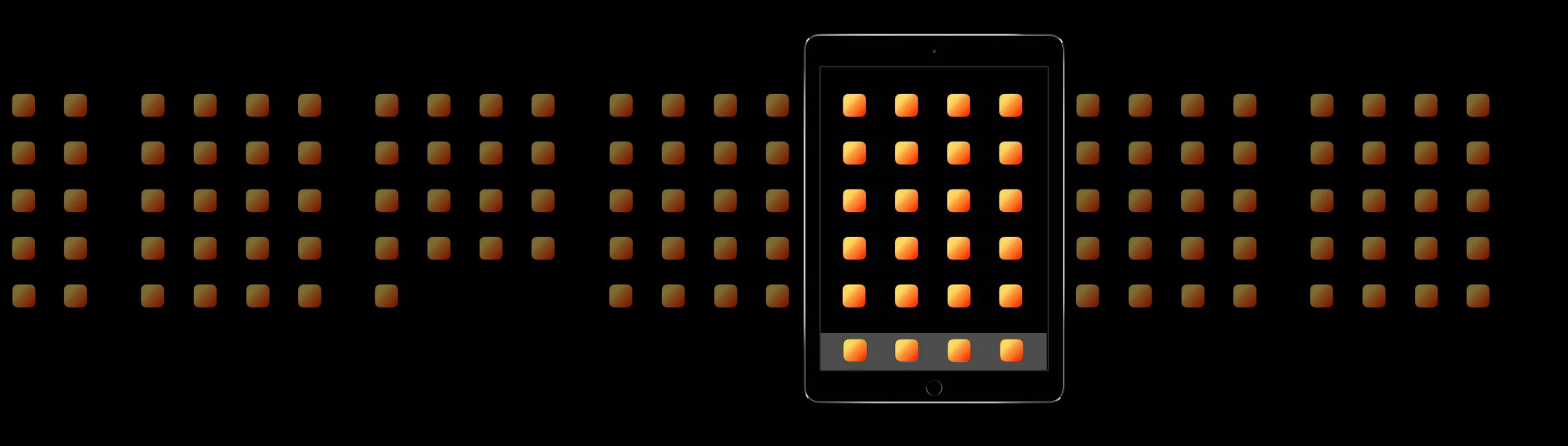


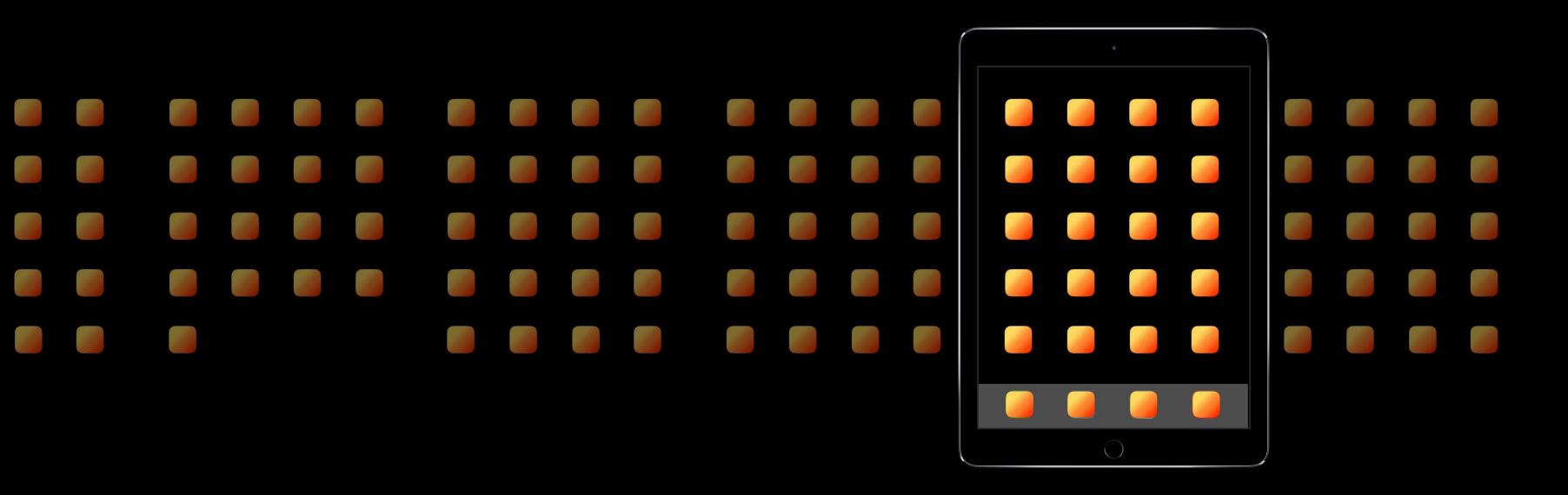


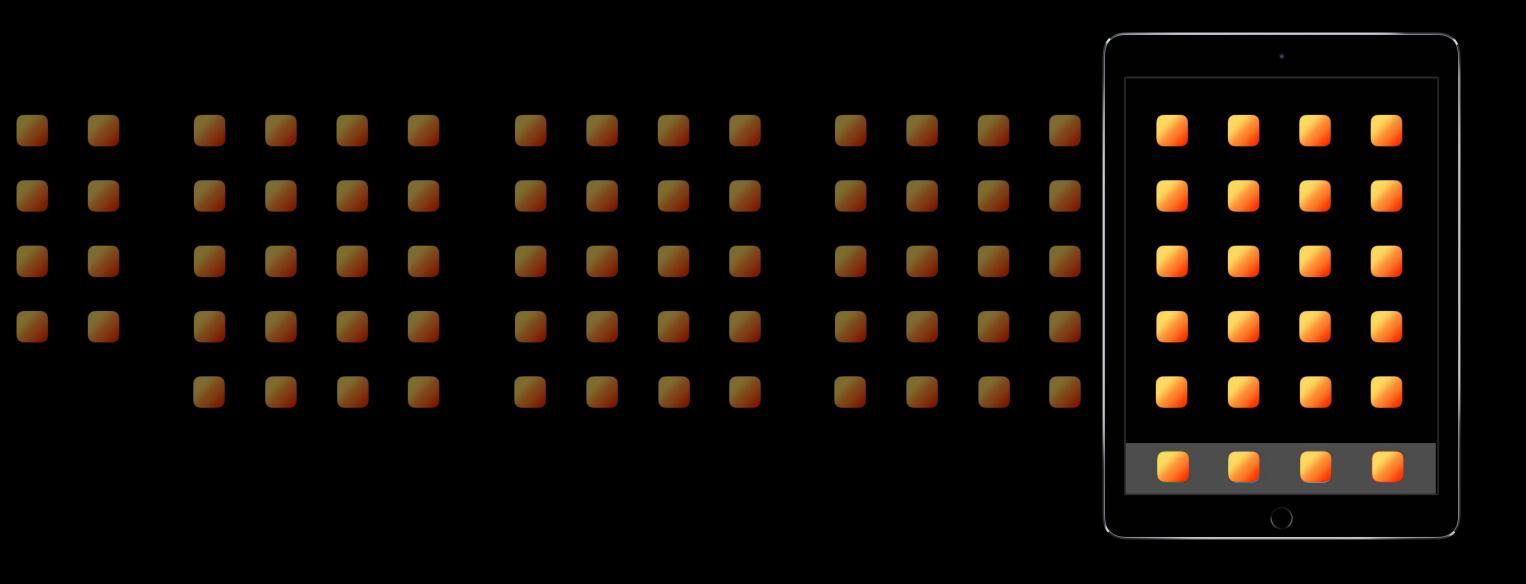


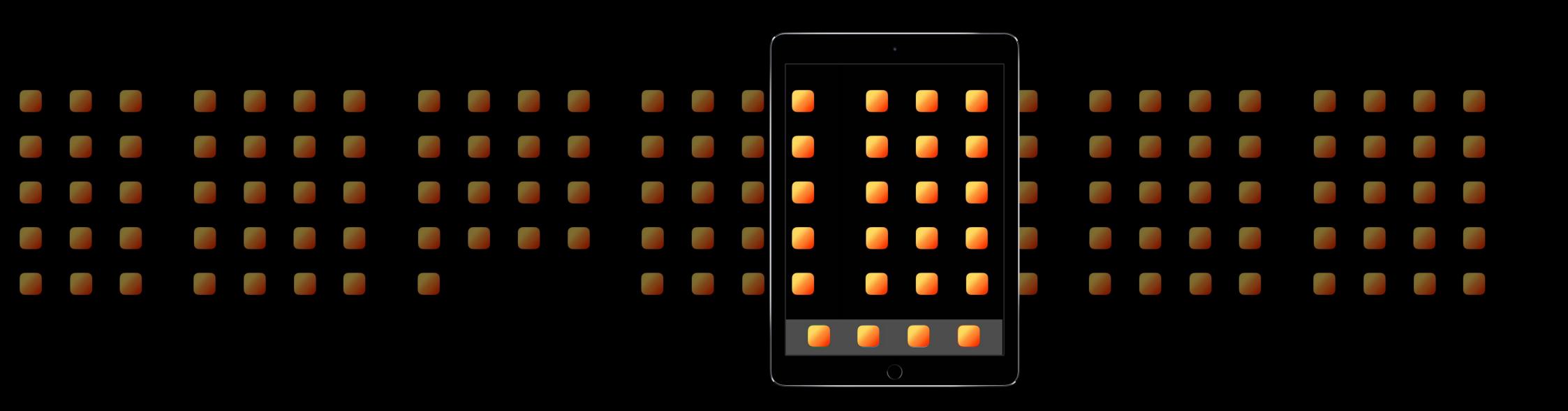




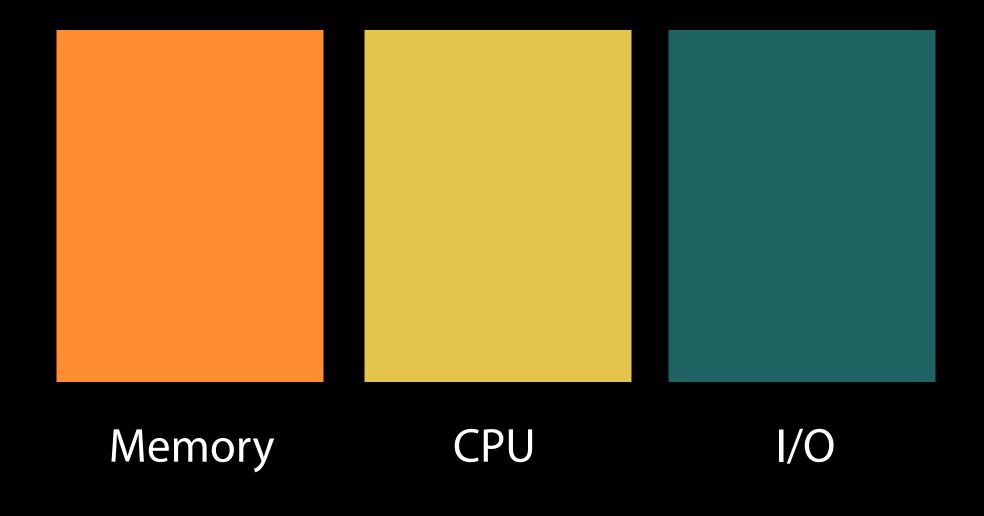






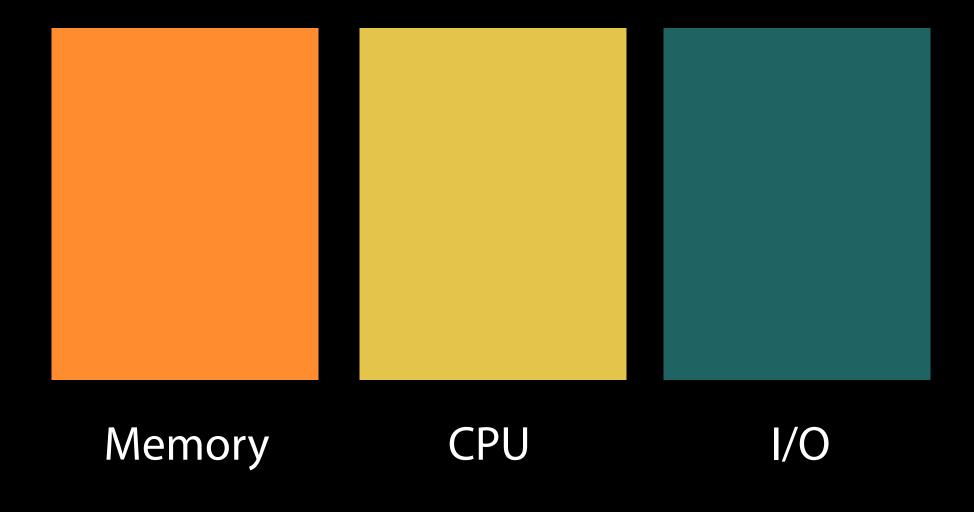






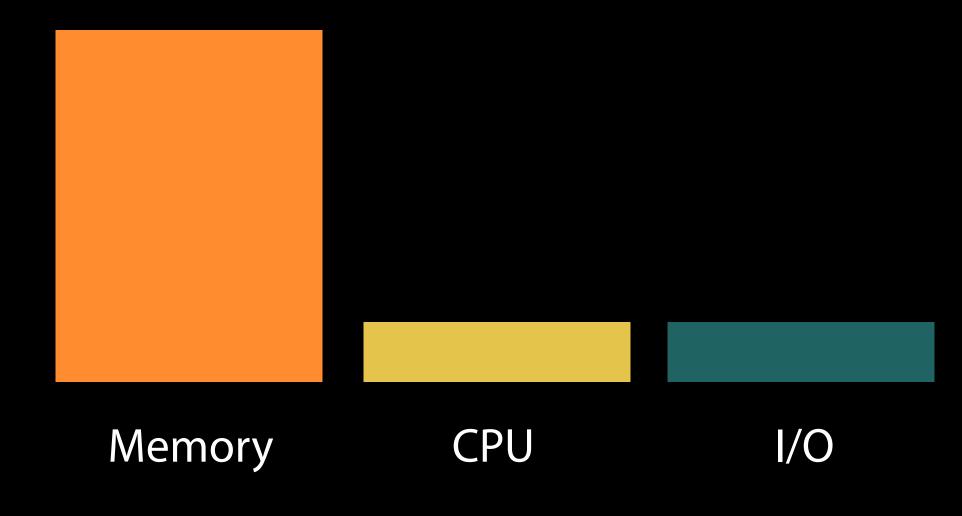


CPU and disk I/O are minimal while scrolling





CPU and disk I/O are minimal while scrolling

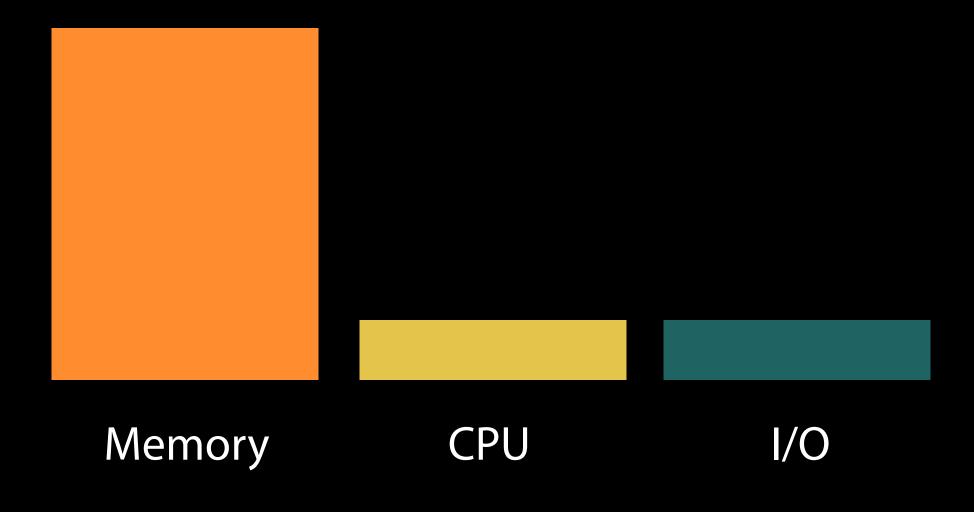




CPU and disk I/O are minimal while scrolling



All icons kept in memory

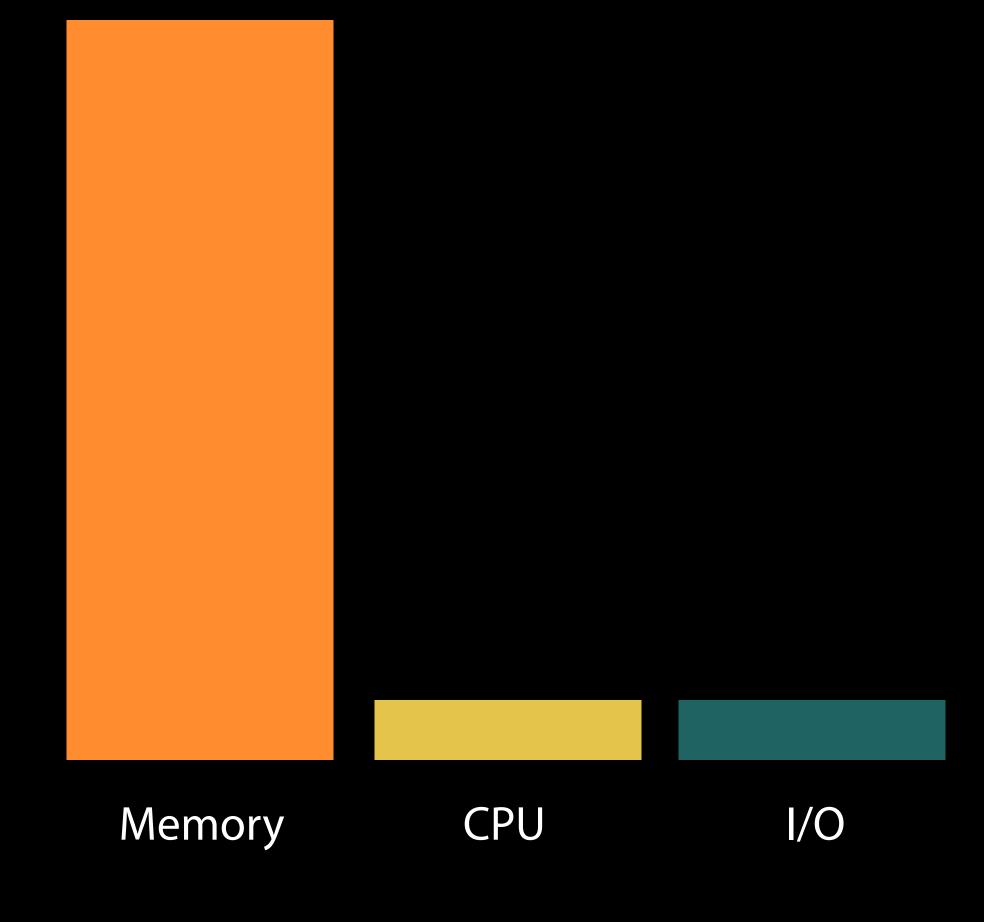




CPU and disk I/O are minimal while scrolling



All icons kept in memory



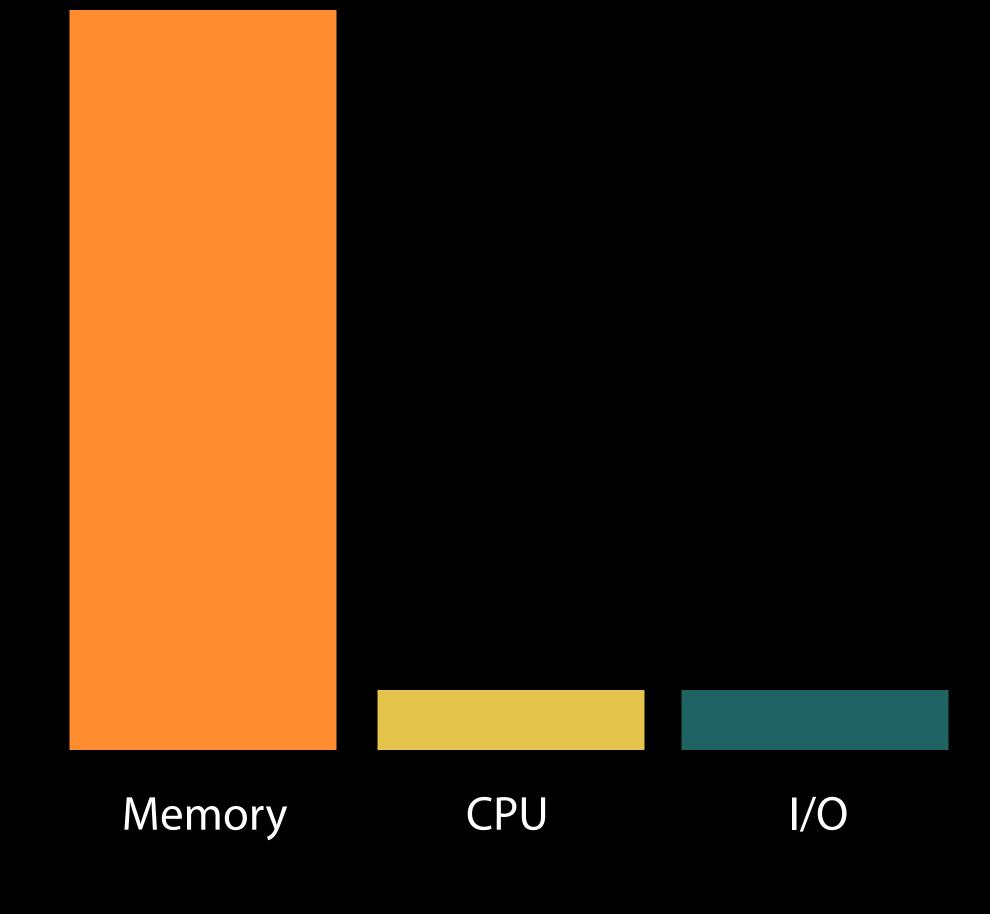


CPU and disk I/O are minimal while scrolling



All icons kept in memory

System reclaims memory by terminating processes



Memory Management

Working set

The critical objects and resources you need right now

The critical objects and resources you need right now

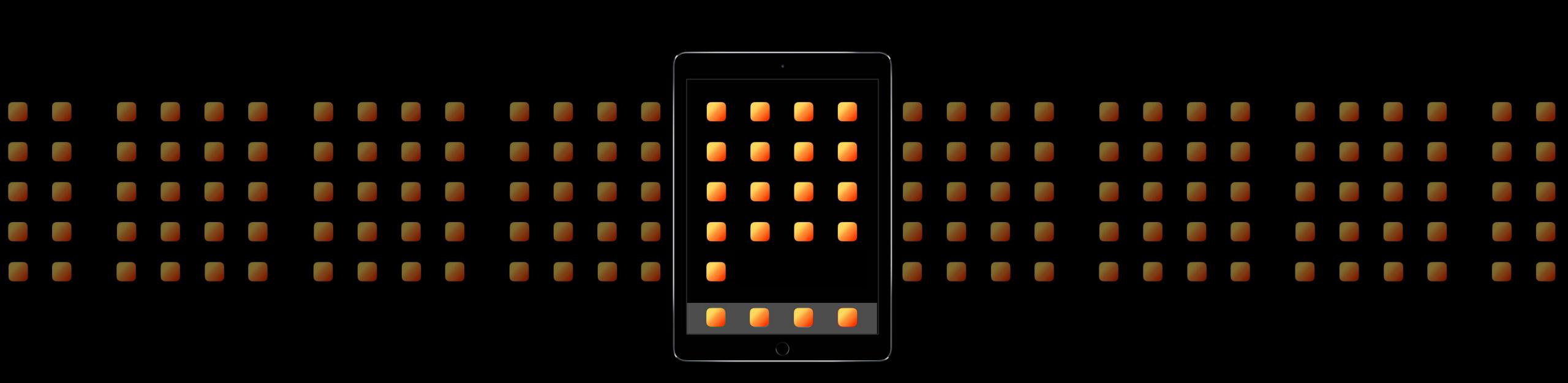
Keep it small

The critical objects and resources you need right now

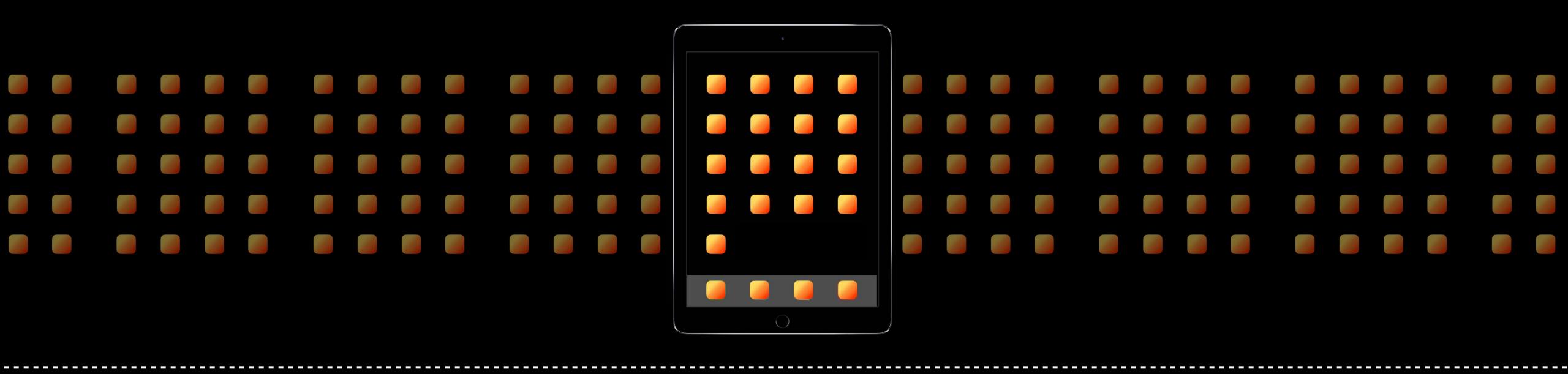
- Keep it small
- It might change based on context

The critical objects and resources you need right now

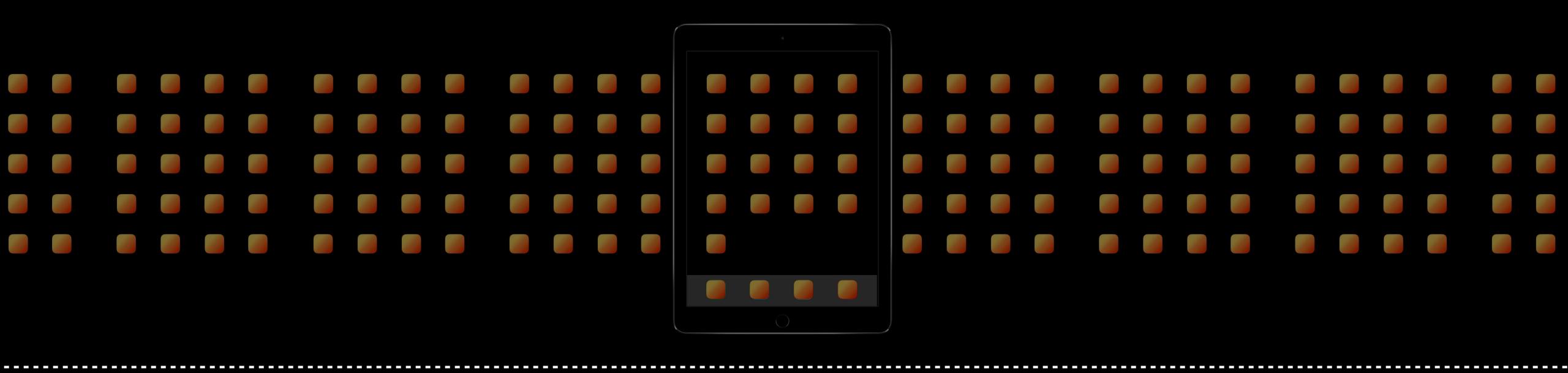
- Keep it small
- It might change based on context
- Don't let it grow unbounded



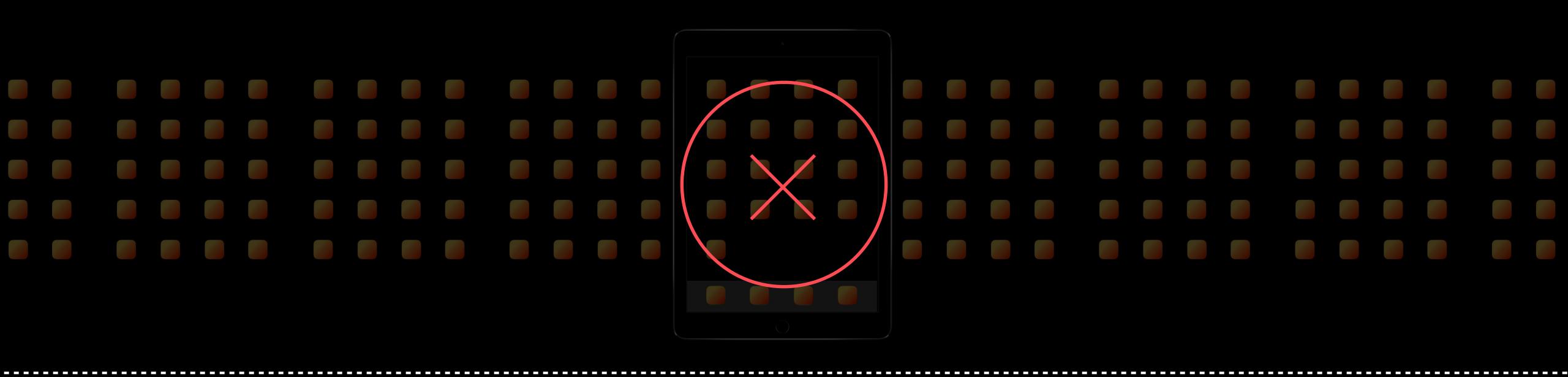
Working Set = Every Page

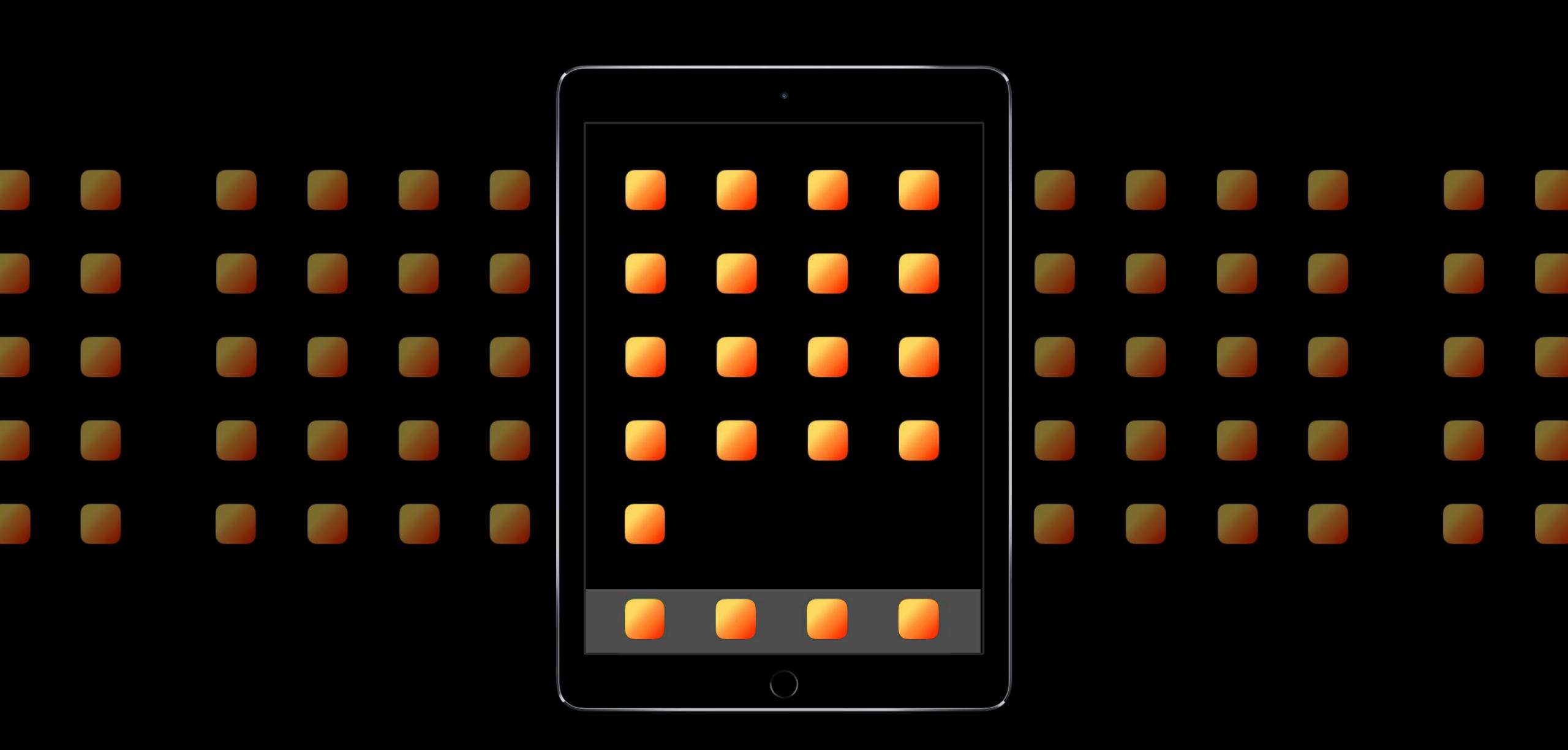


Working Set = Every Page

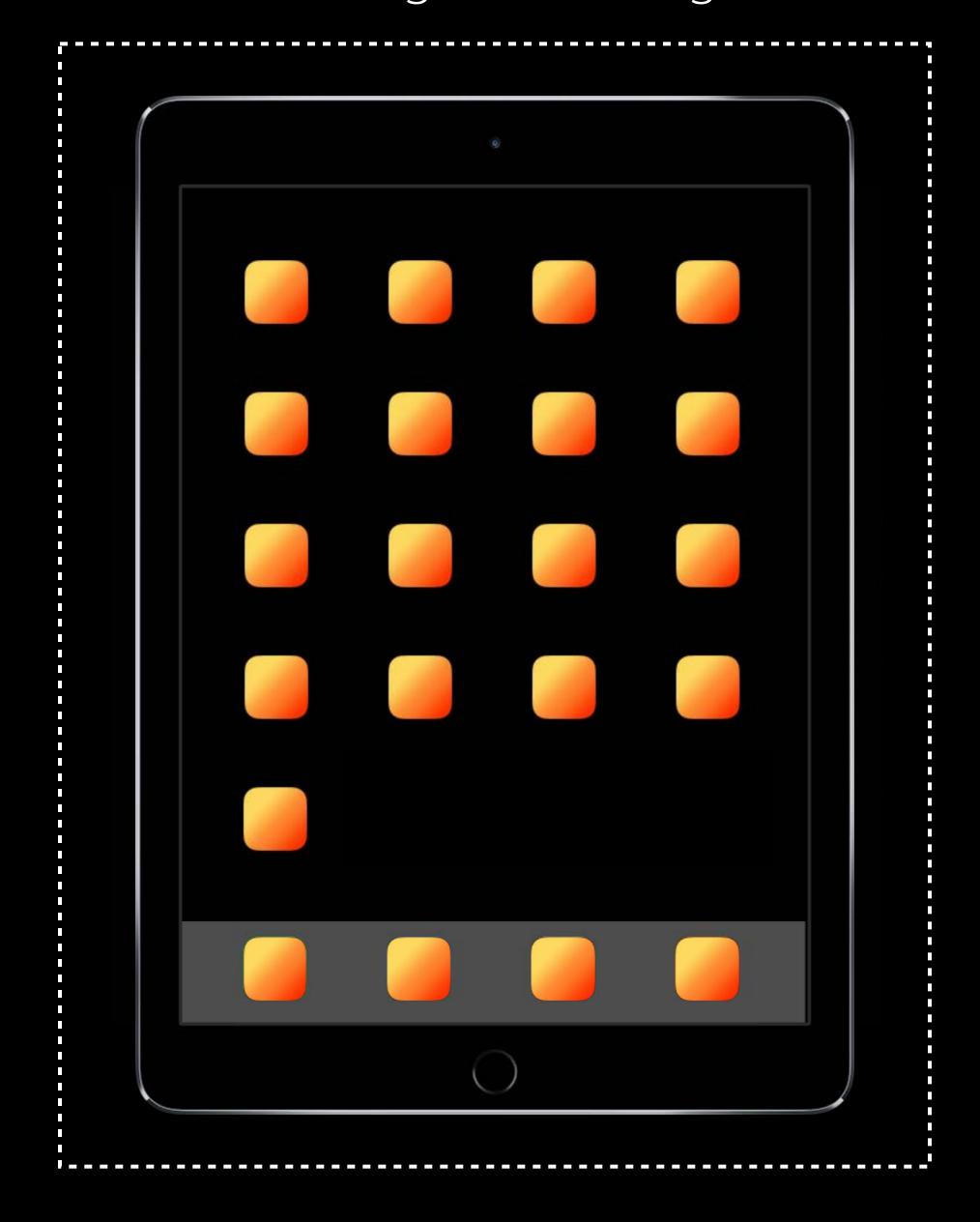


Working Set = Every Page

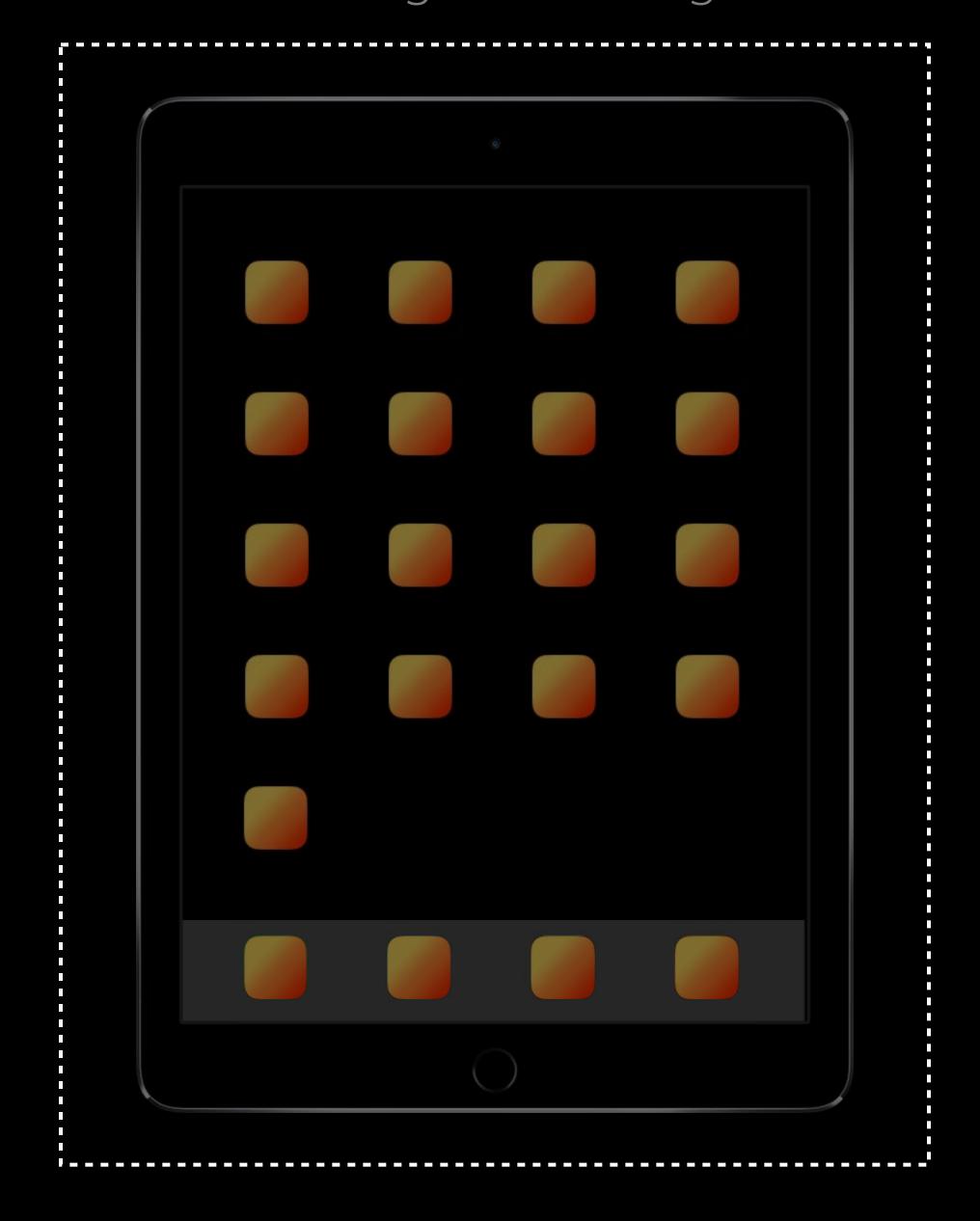




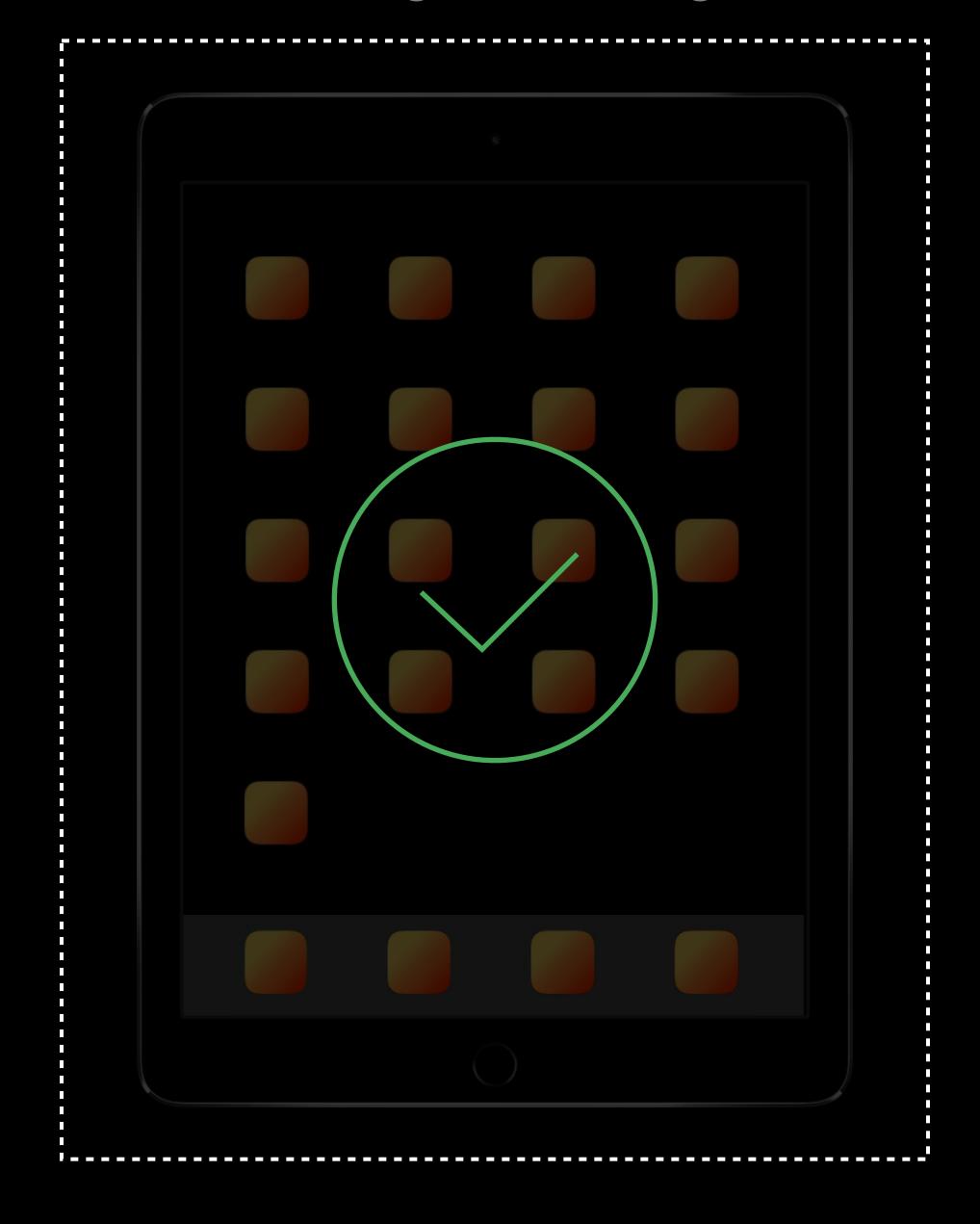
Working Set = 1 Page

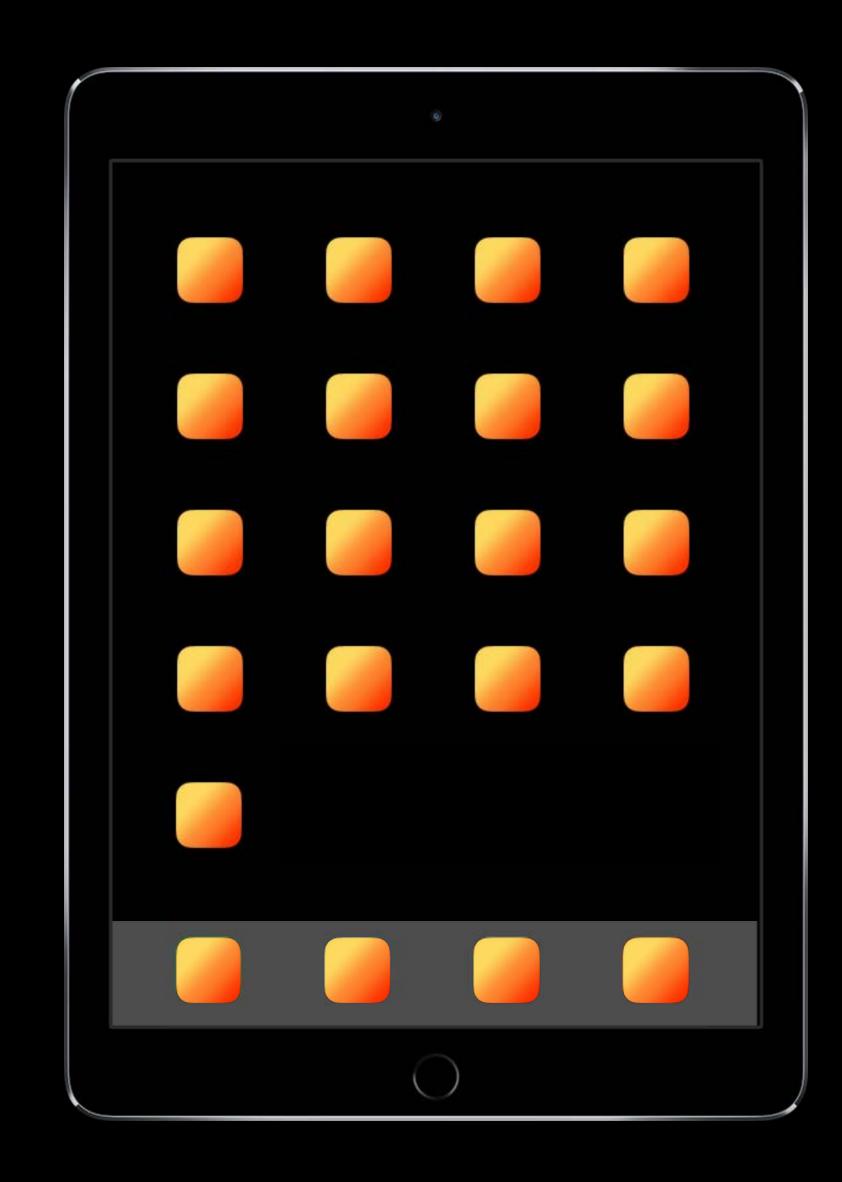


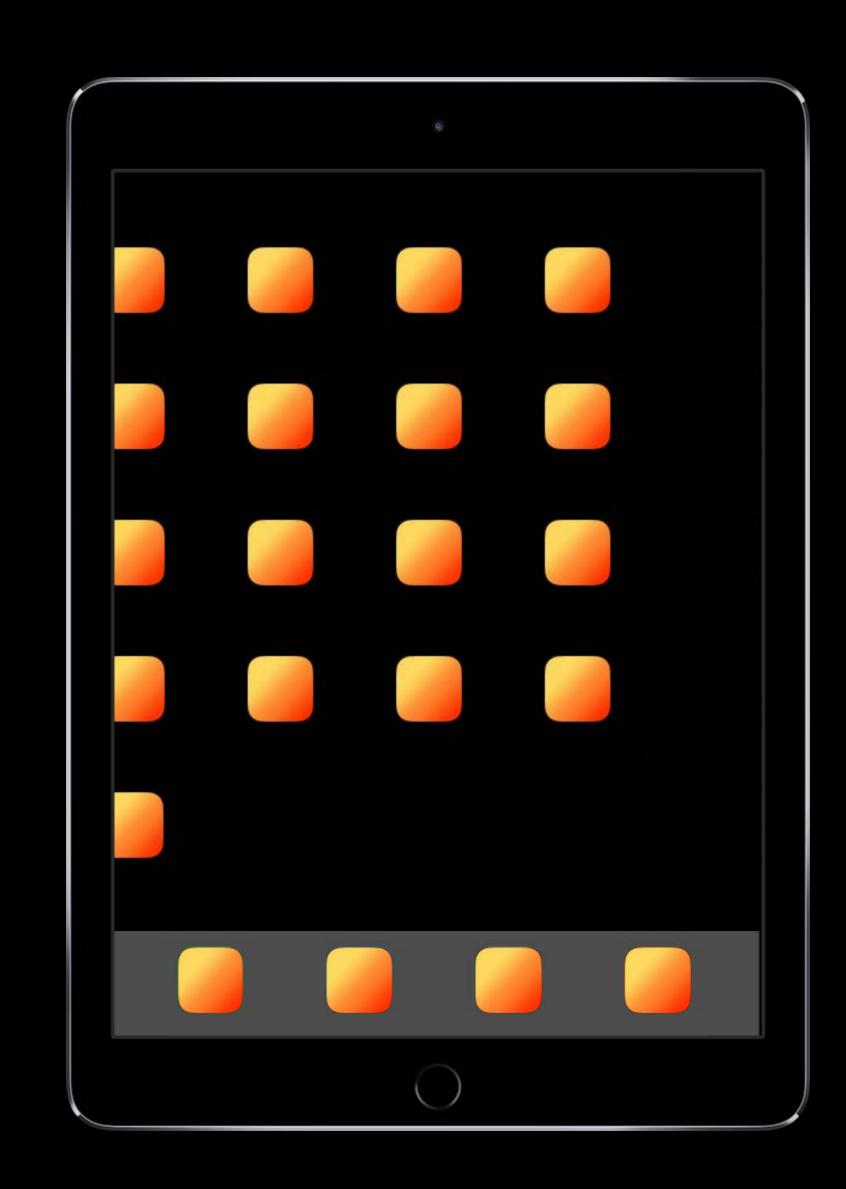
Working Set = 1 Page

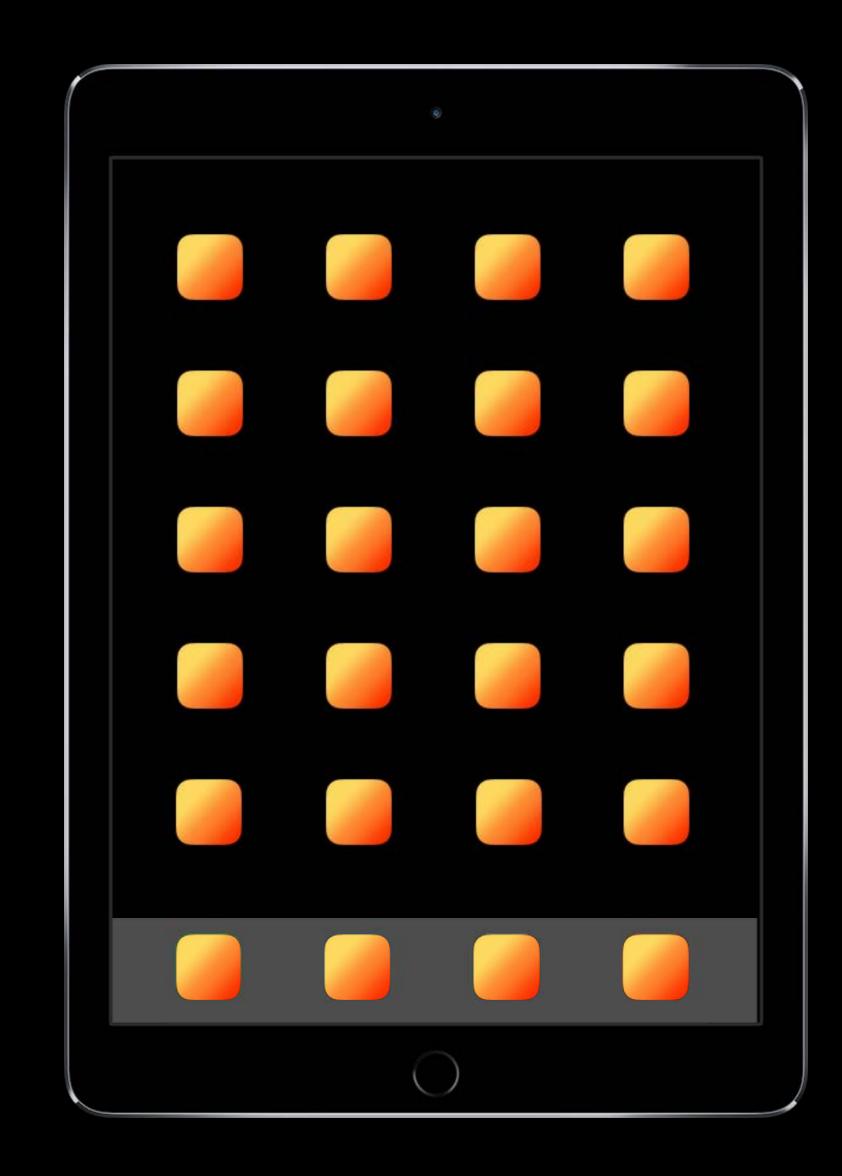


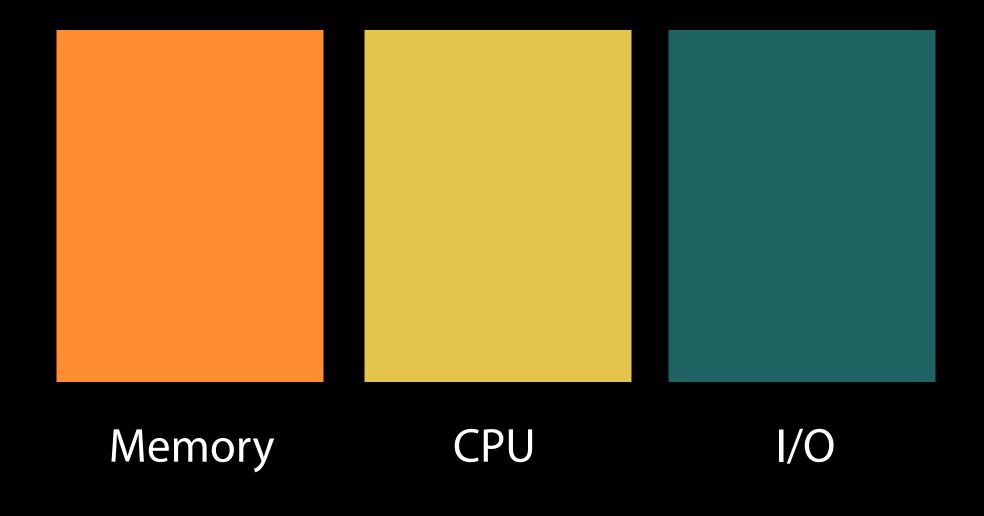
Working Set = 1 Page





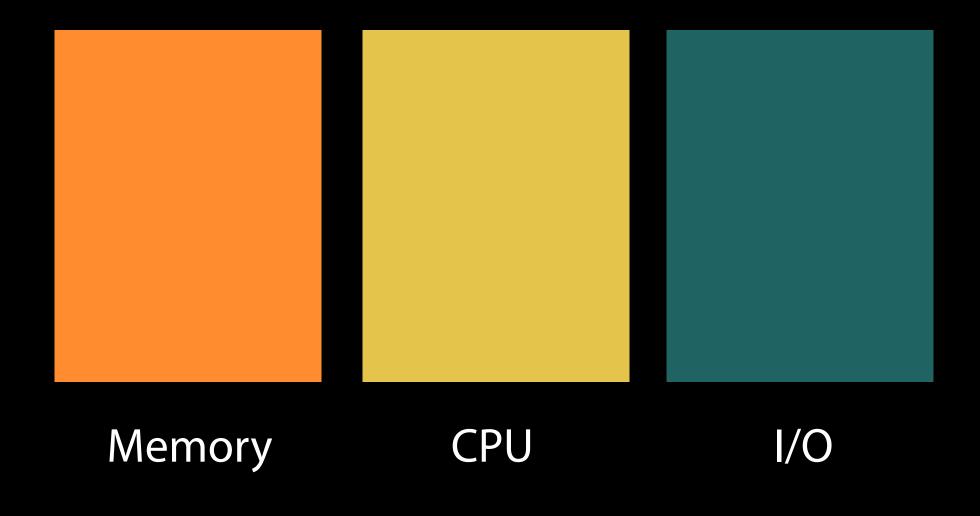






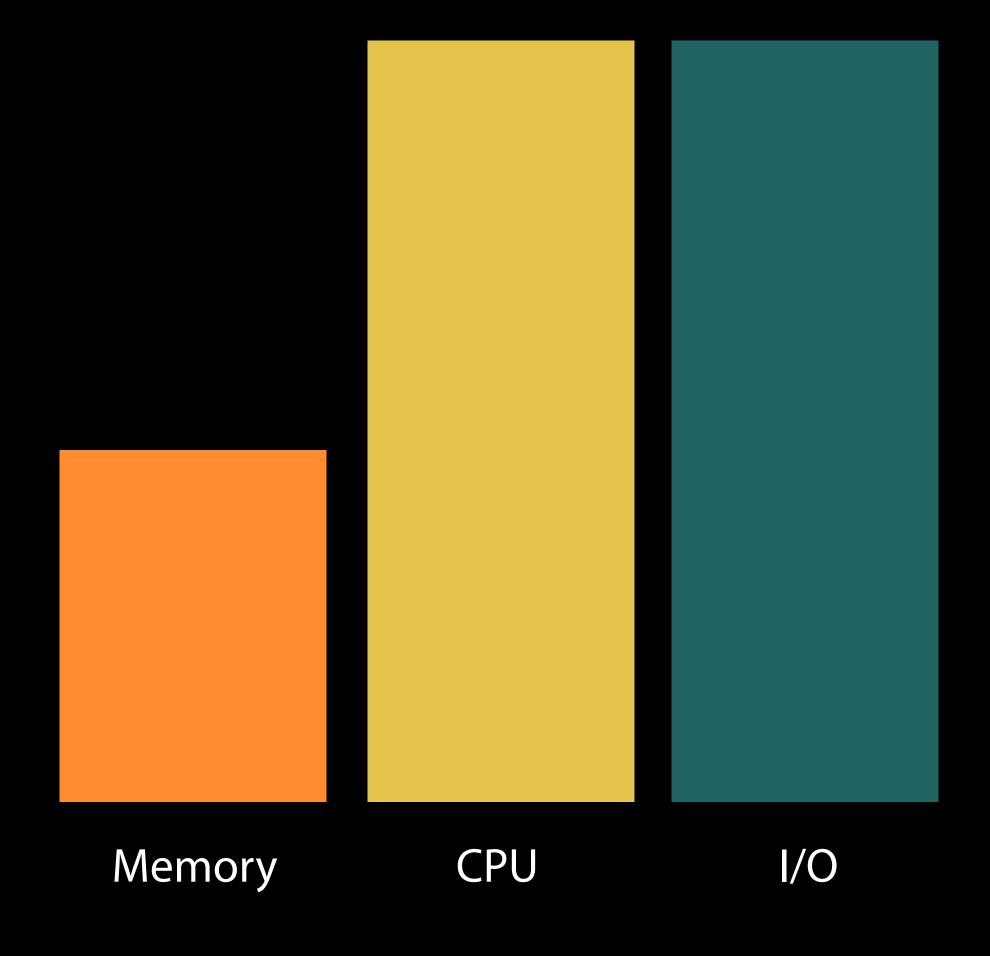


CPU and disk I/O are high while scrolling





CPU and disk I/O are high while scrolling

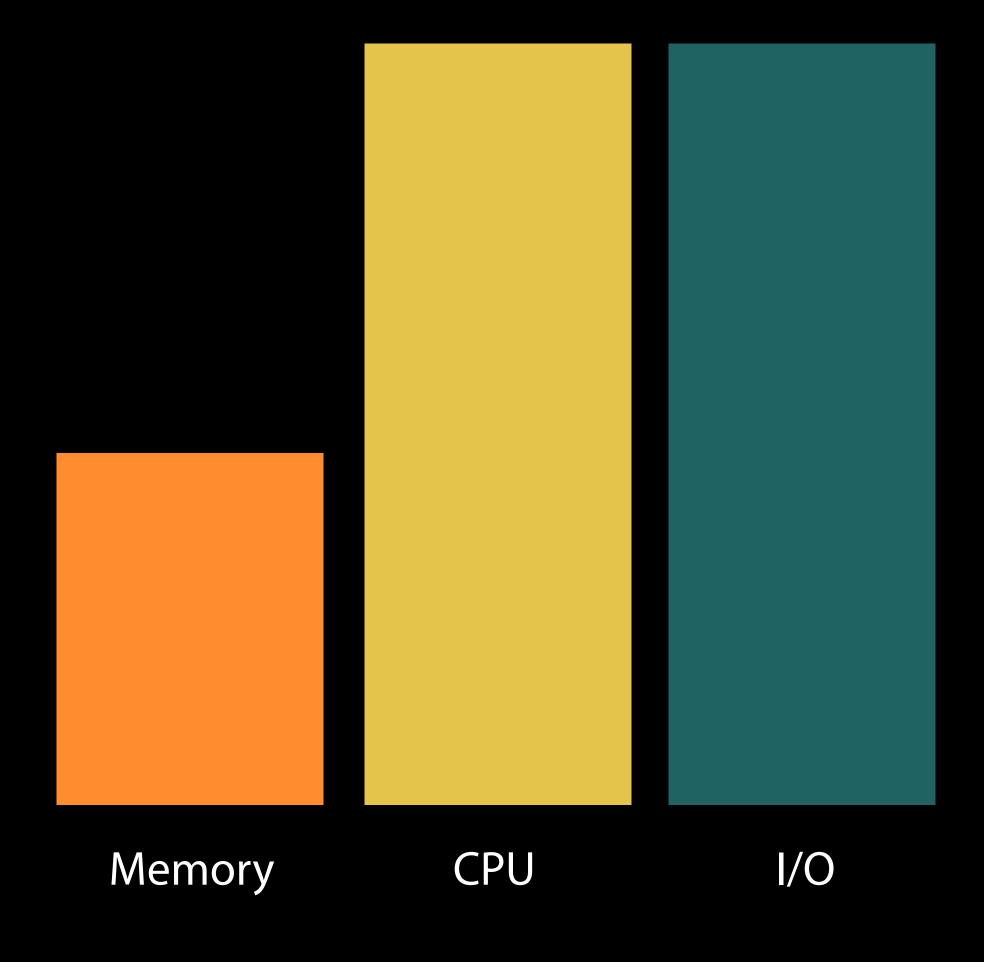




CPU and disk I/O are high while scrolling



Very few icons in memory

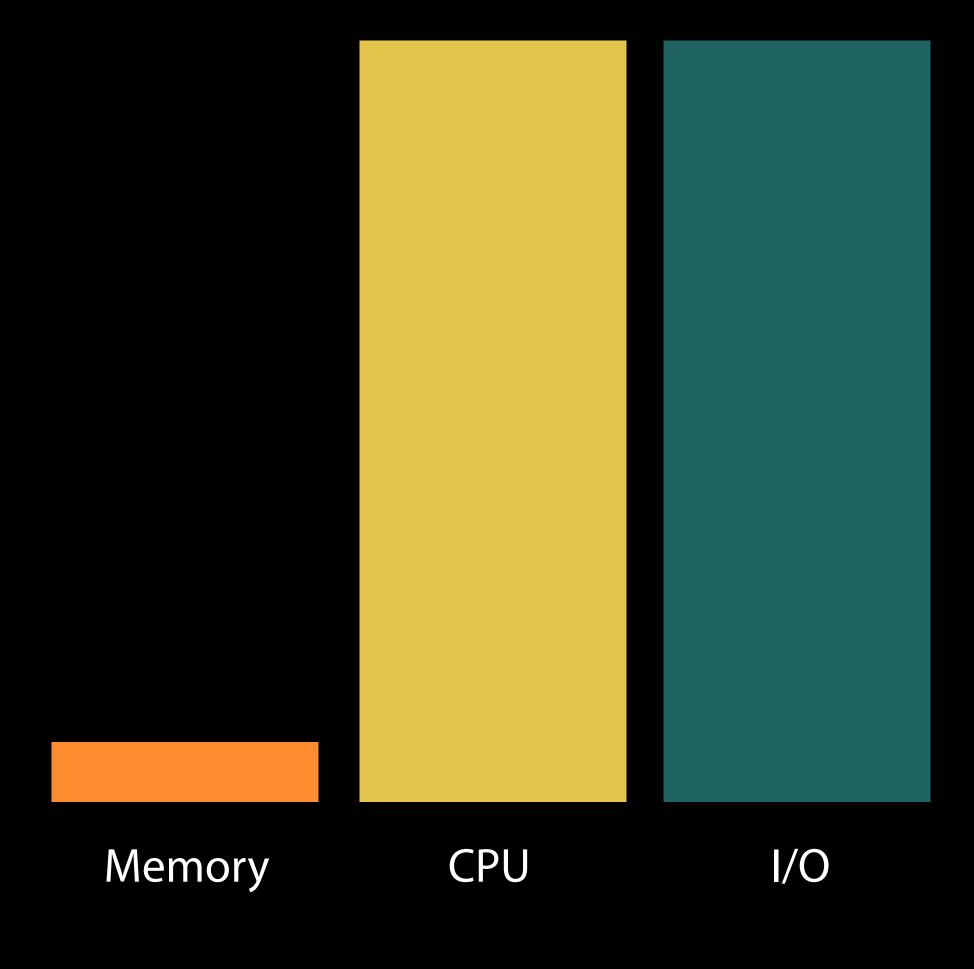




CPU and disk I/O are high while scrolling



Very few icons in memory



CPU Management

Main thread

Main thread's top priority is responding to user events

Don't do unnecessary work

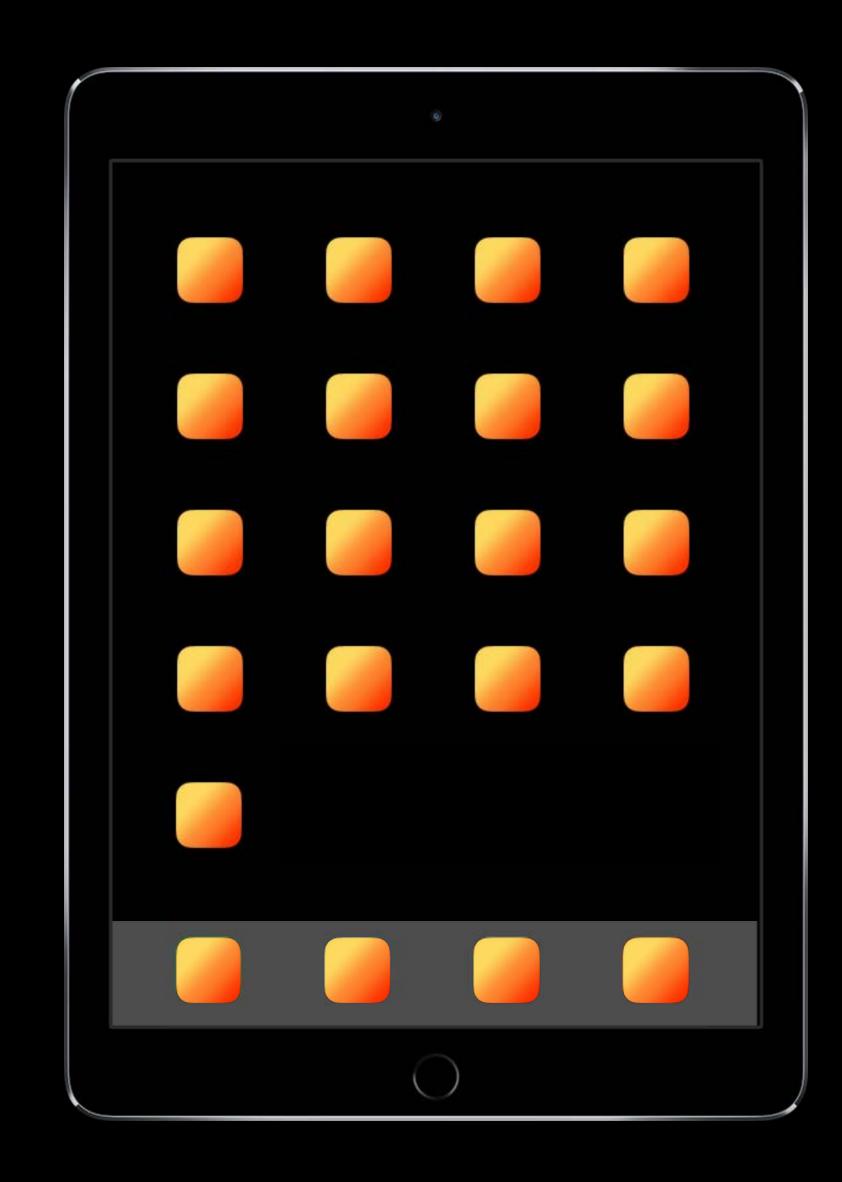
CPU Management

Quality of service

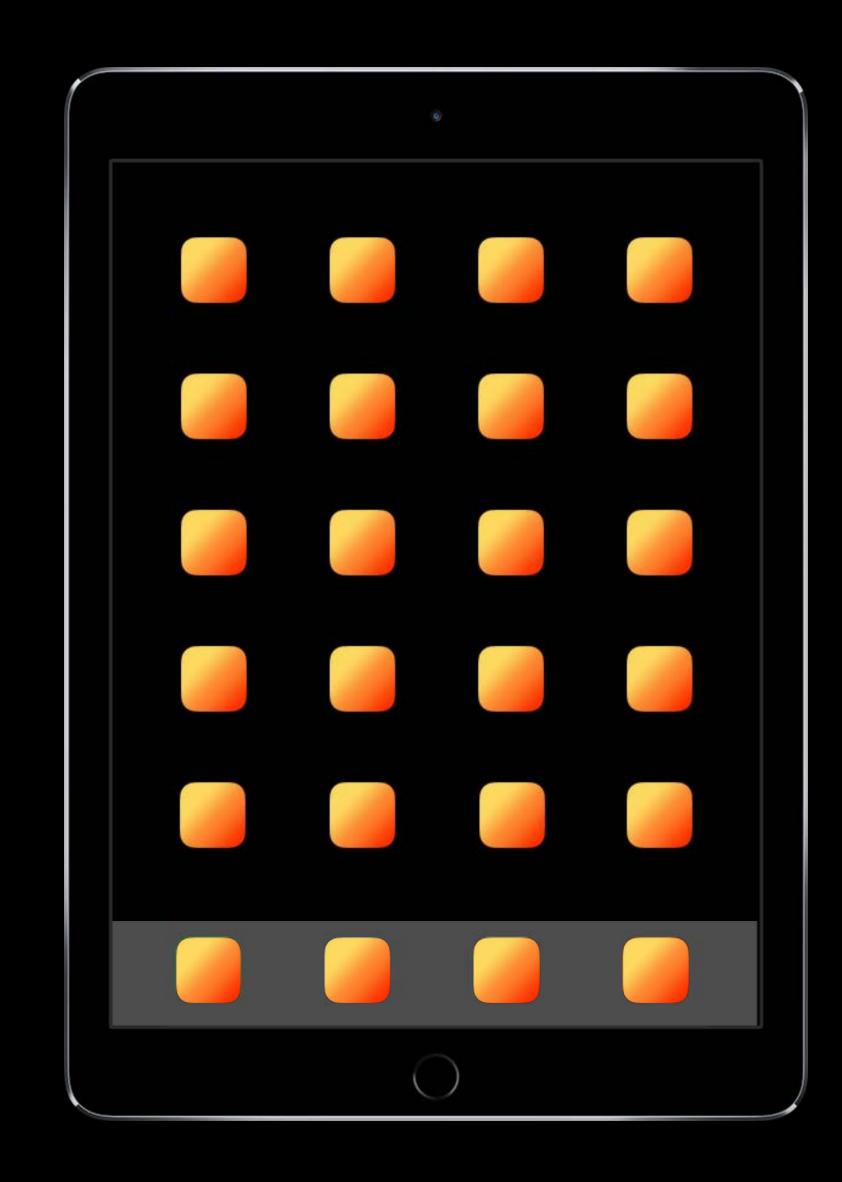
UI	User Interactive*	Is this work actively involved in updating the UI?
IN	User Initiated	Is this work required to continue user interaction?
UT	Utility	Is the user aware of the progress of this work?
BG	Background	Can this work be deferred to start at a better time?
		* Main thread

Building Responsive and Efficient Apps with GCD	Nob Hill	Friday 10:00AM
Power, Performance, and Diagnostics: What's New in GCE	Dand XPC	WWDC14

```
for (IRIcon *icon in [iconPage icons]) {
    dispatch_async(iconQueue, ^{
        [self _iconQueue_generateImageForIcon:icon];
    });
}];
```







CPU Management

QoS overrides

Lower priority queues can be temporarily boosted when a higher priority queue is waiting for it

The system needs a hint that you want a QoS override to occur

QoS Overrides

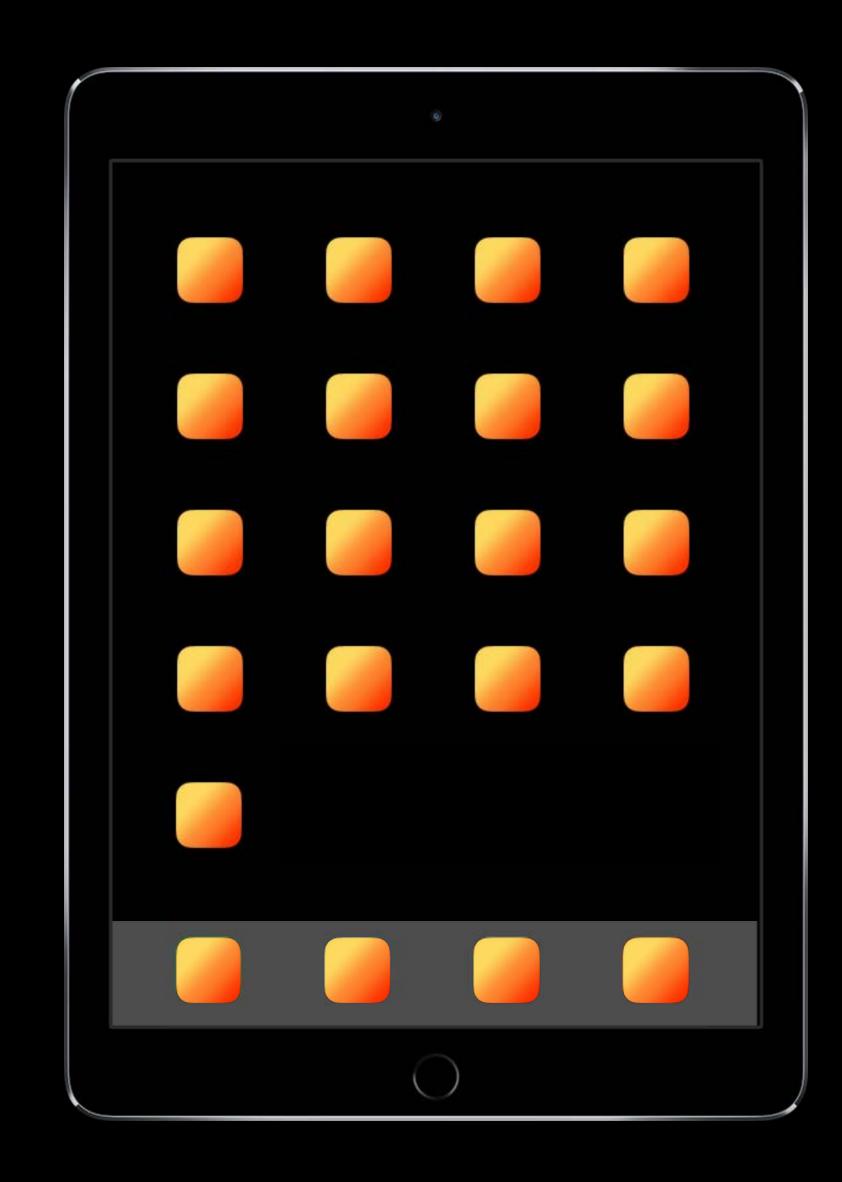
Dispatch

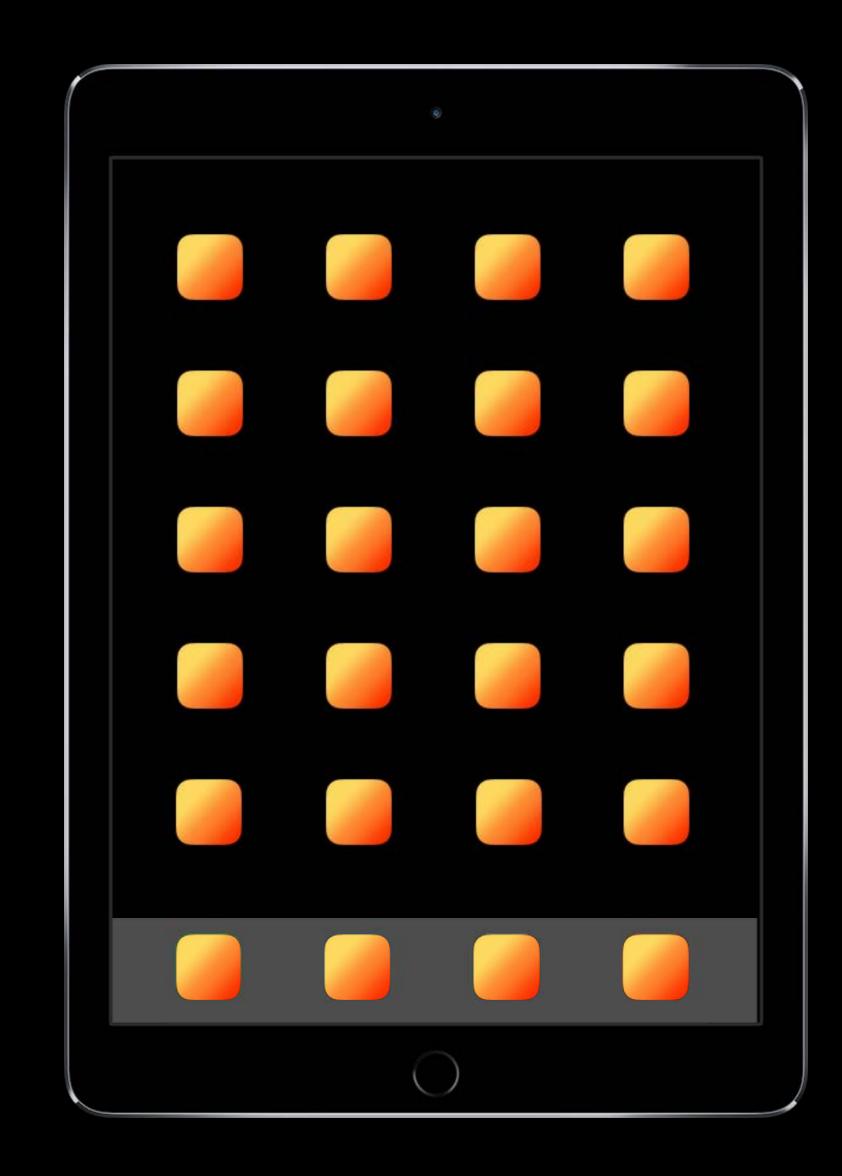
Overrides?
X
X

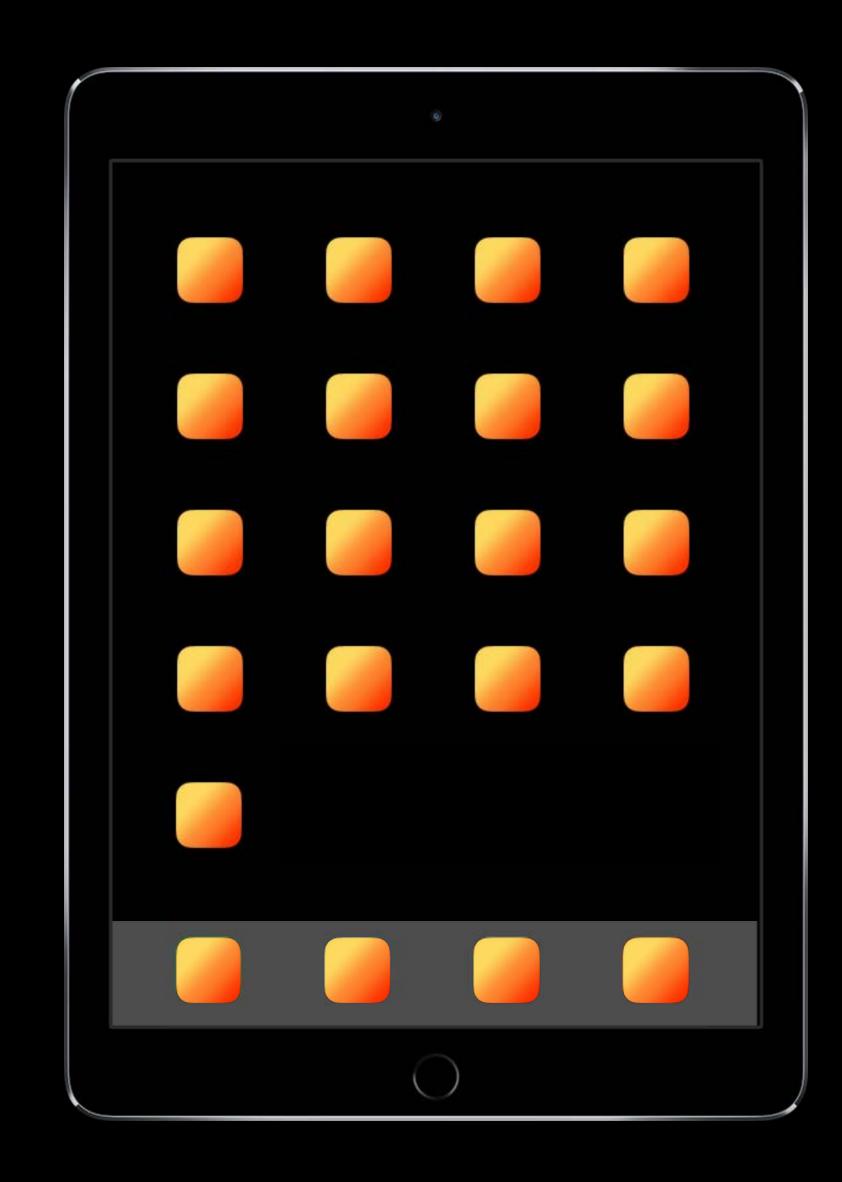
Building Responsive and Efficient Apps with GCD	Nob Hill	Friday 10:00AM
Power, Performance, and Diagnostics: What's New in GCI	D and XPC	WWDC14

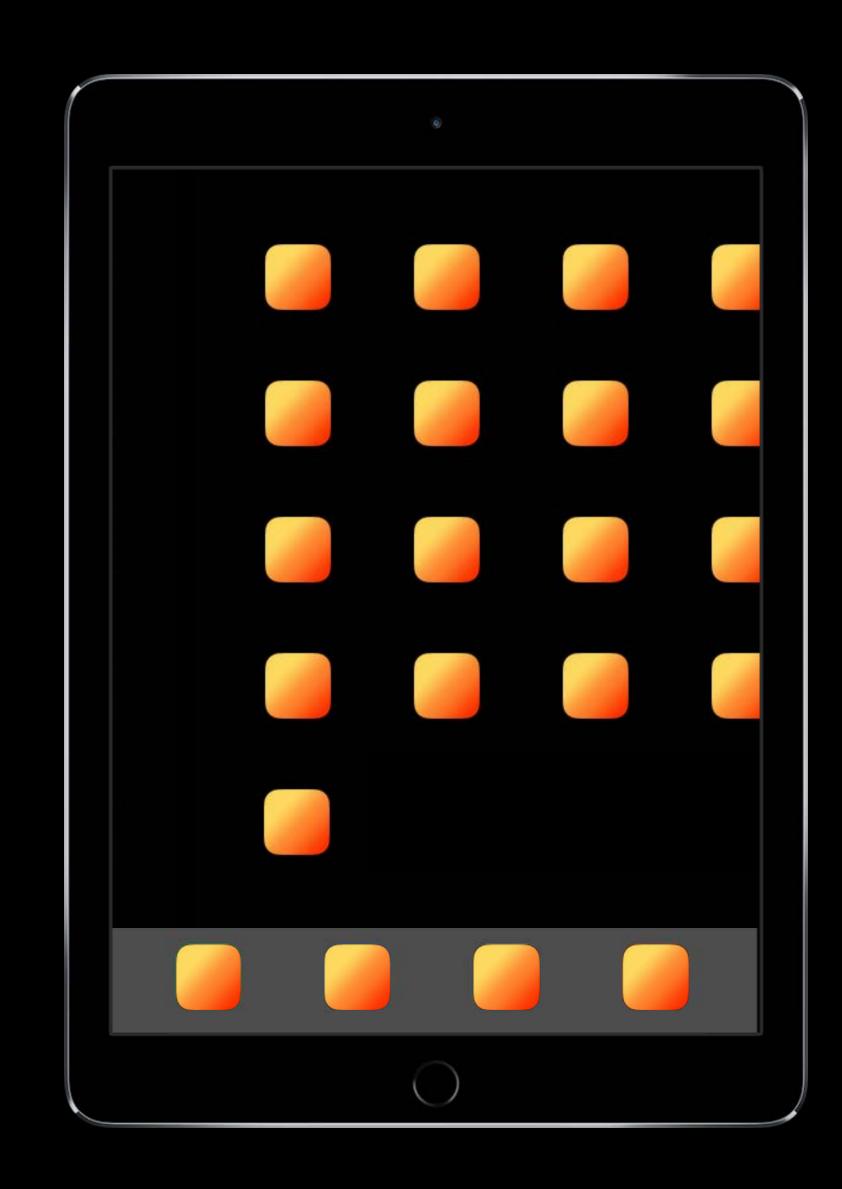
```
dispatch_sync(iconQueue, ^{
      // ensure icon generation completes at caller QoS
});
```

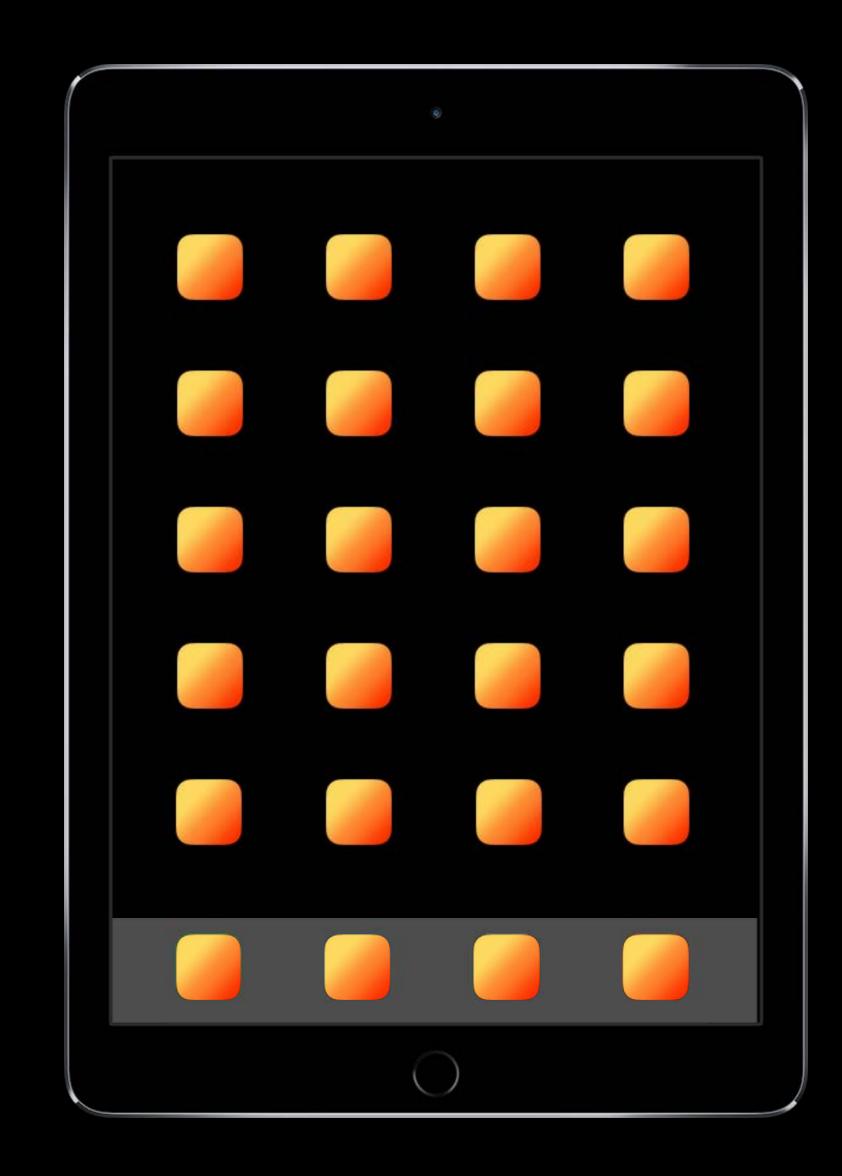
UIImage *image = [icon image];







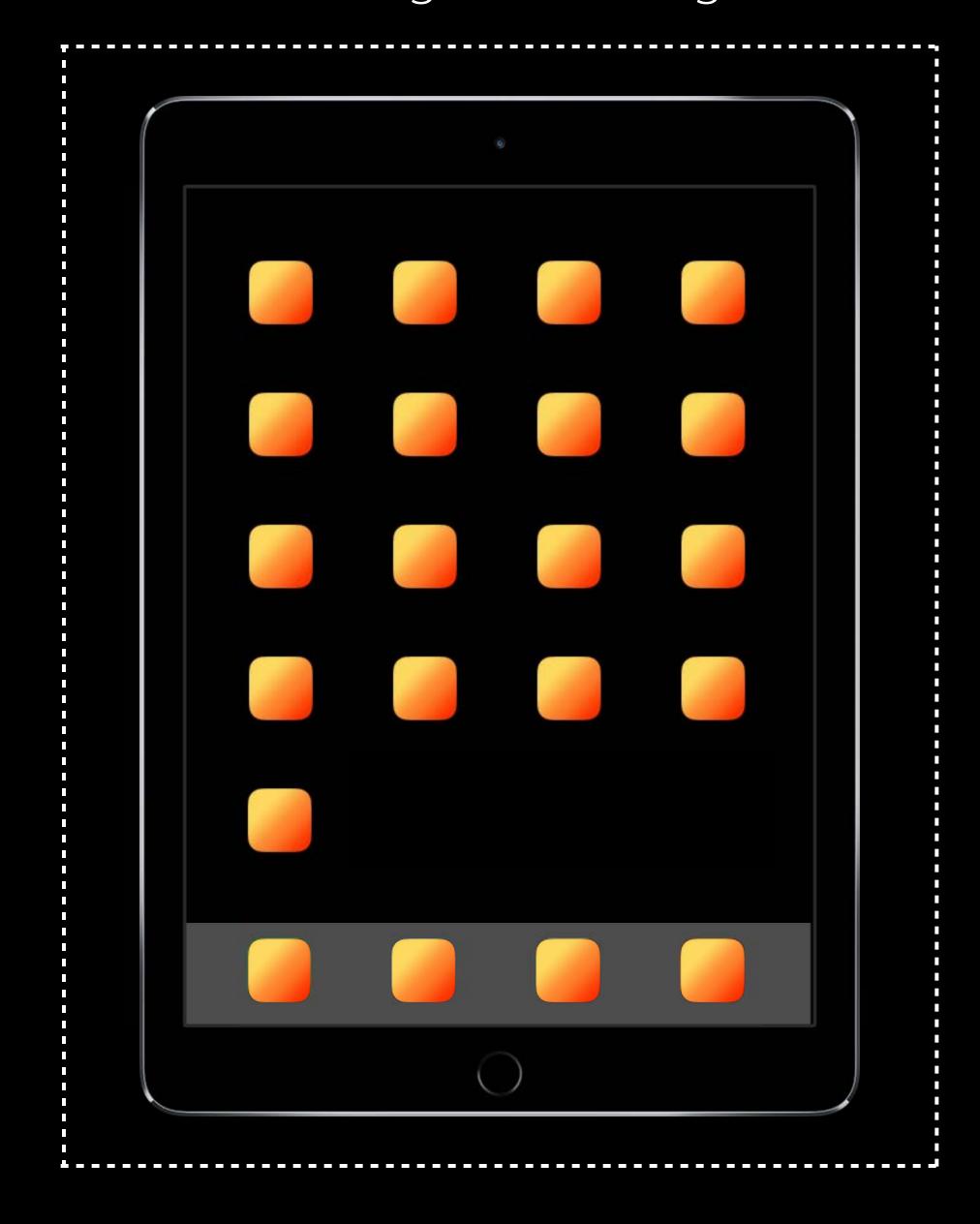




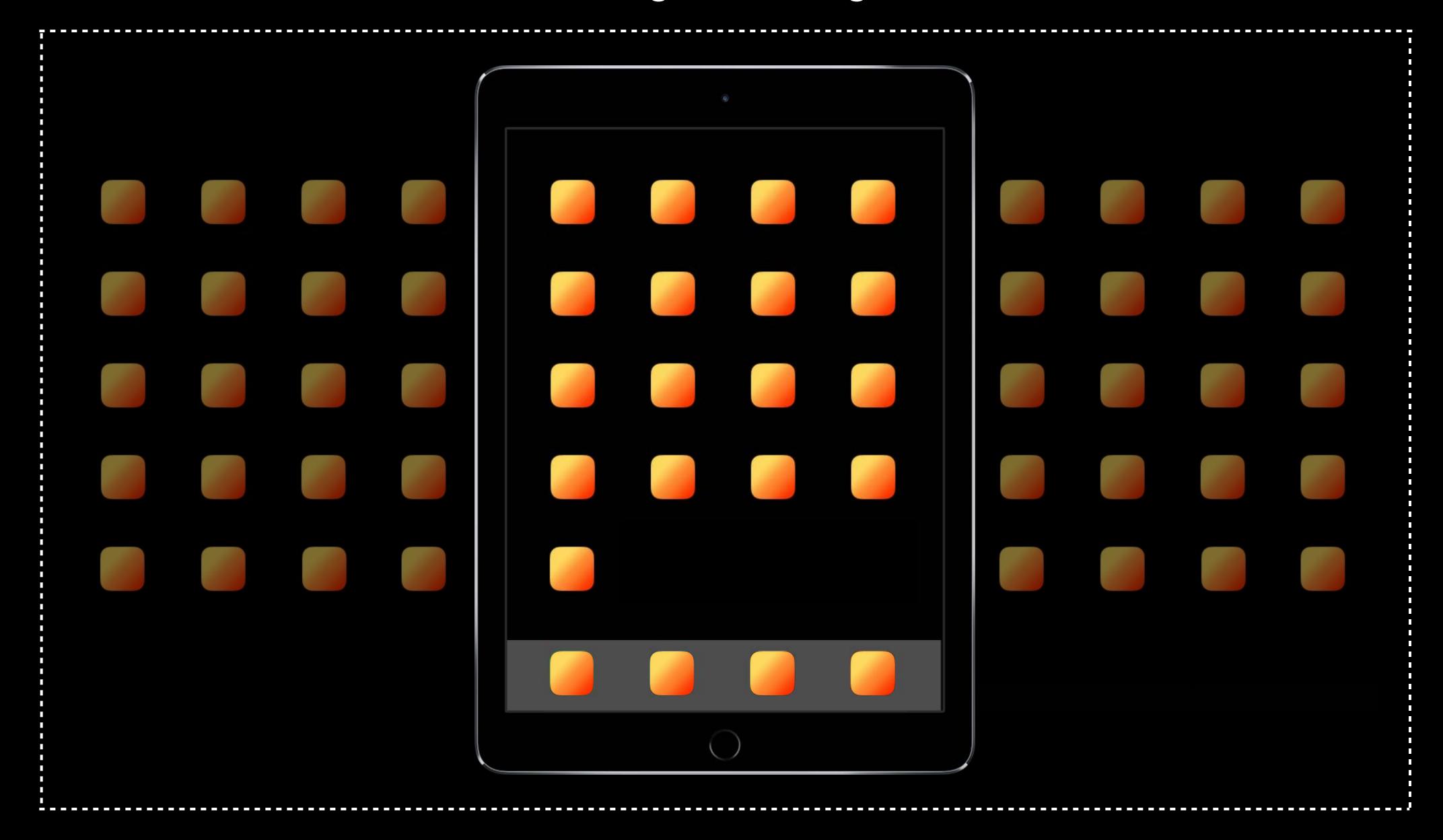




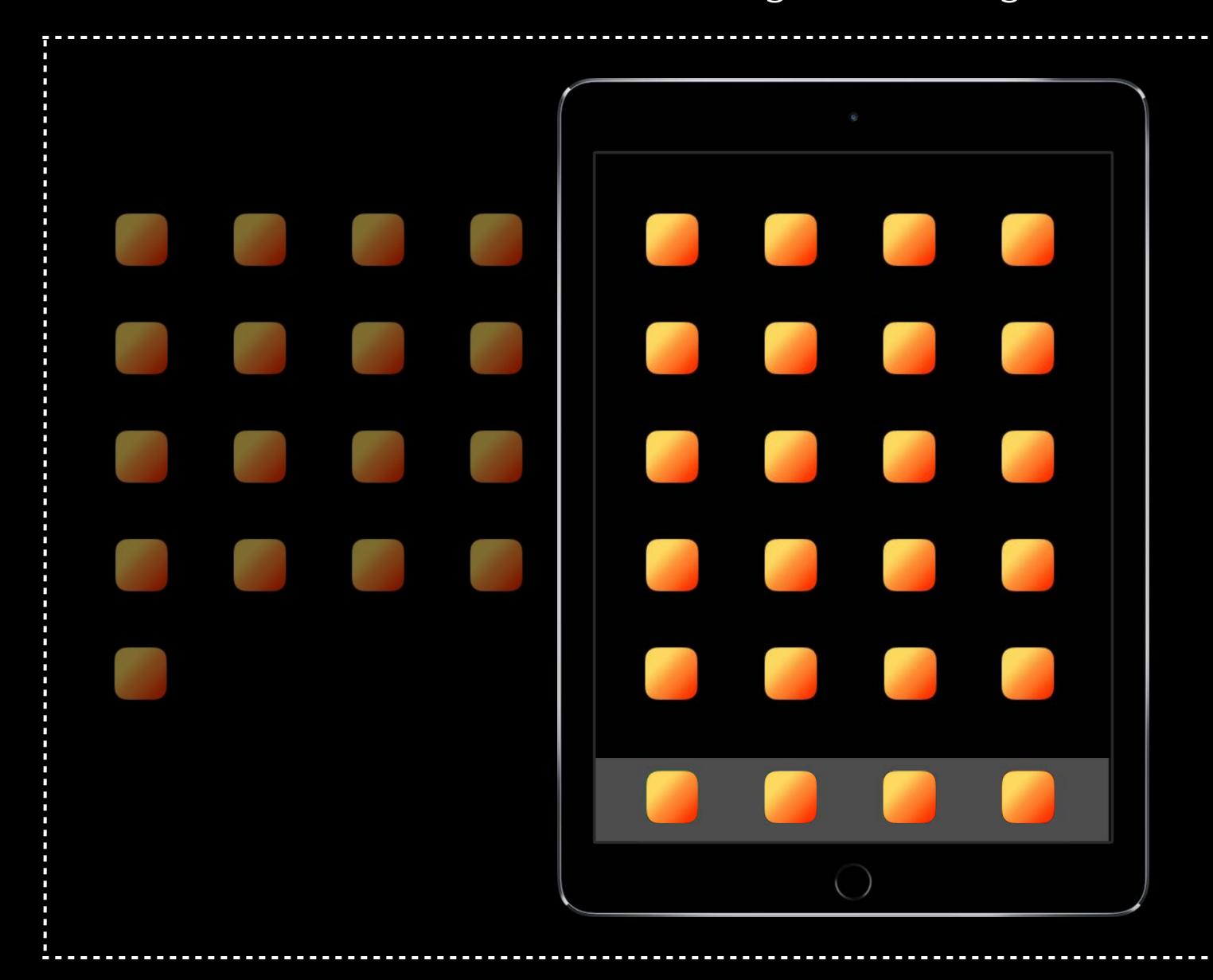
Working Set = 1 Page



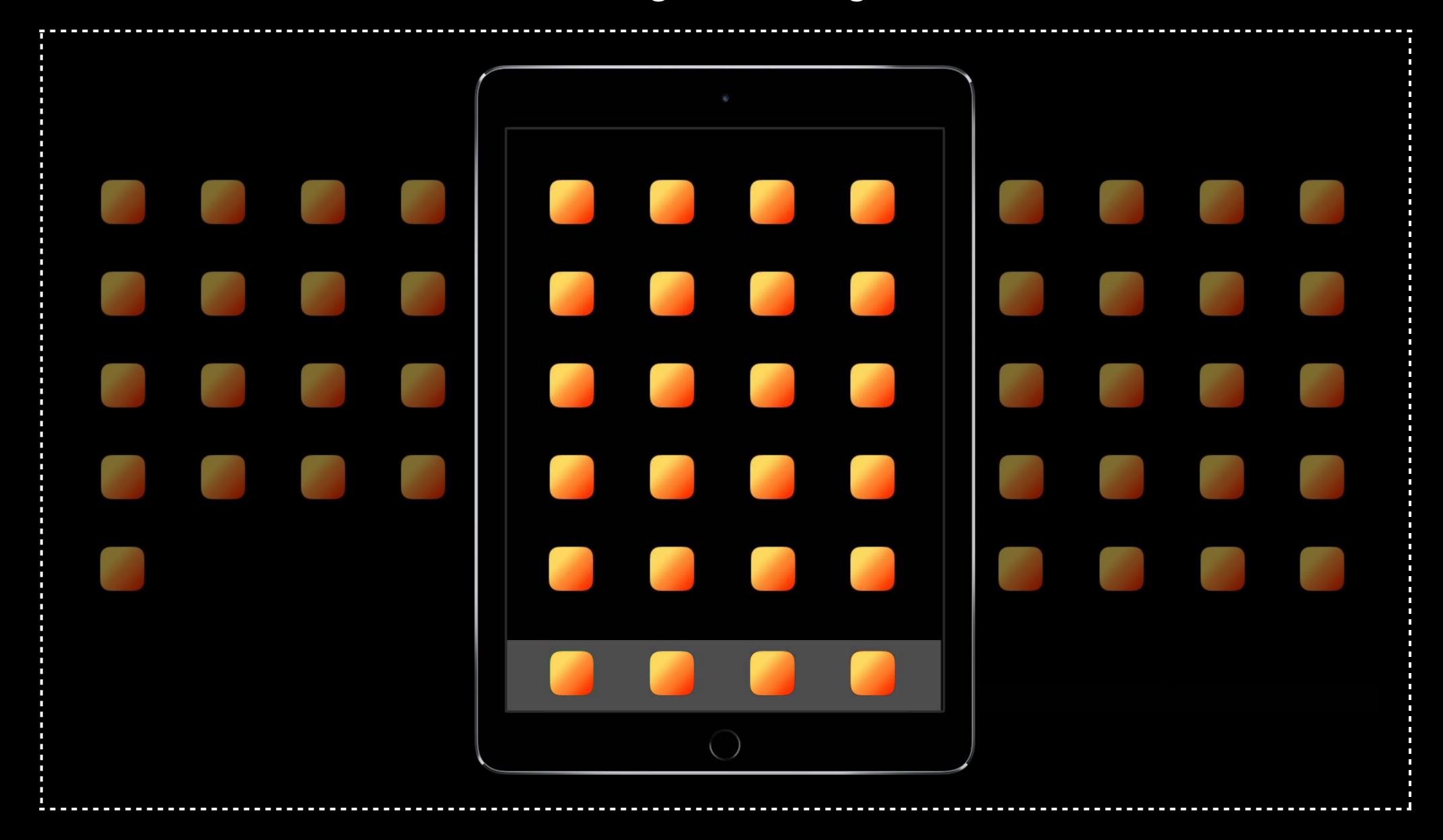
Working Set = 3 Pages



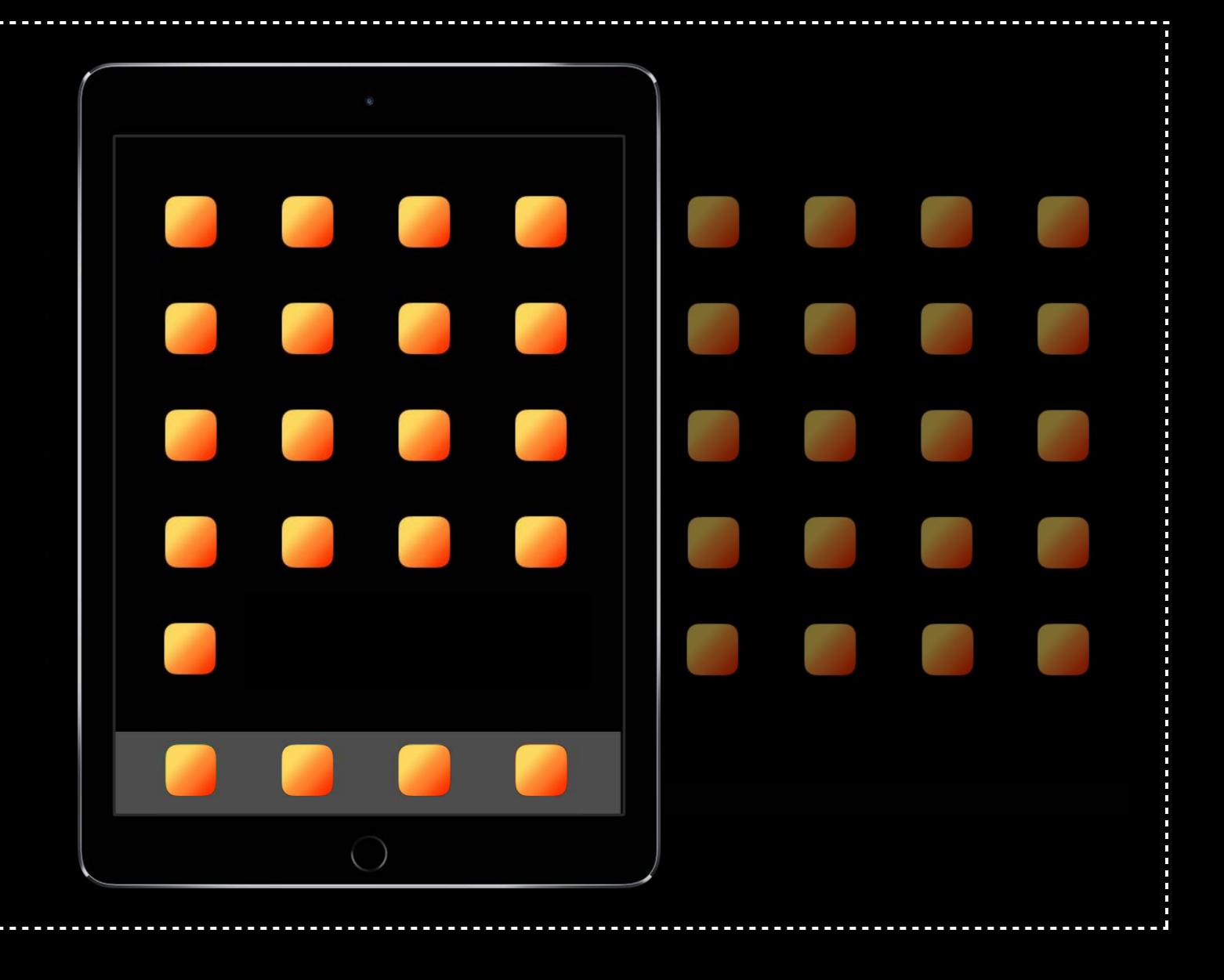
Working Set = 3 Pages



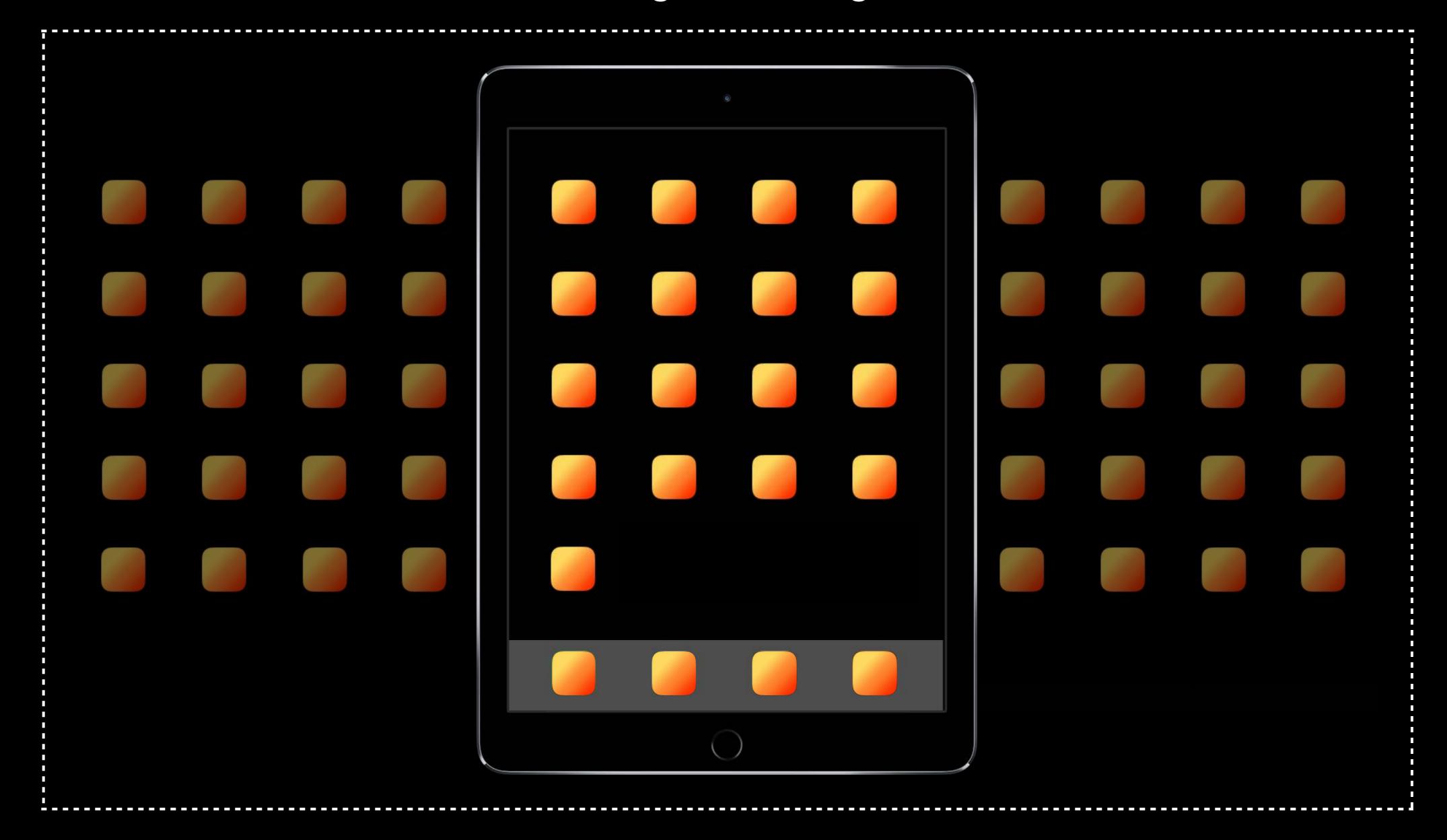
Working Set = 3 Pages



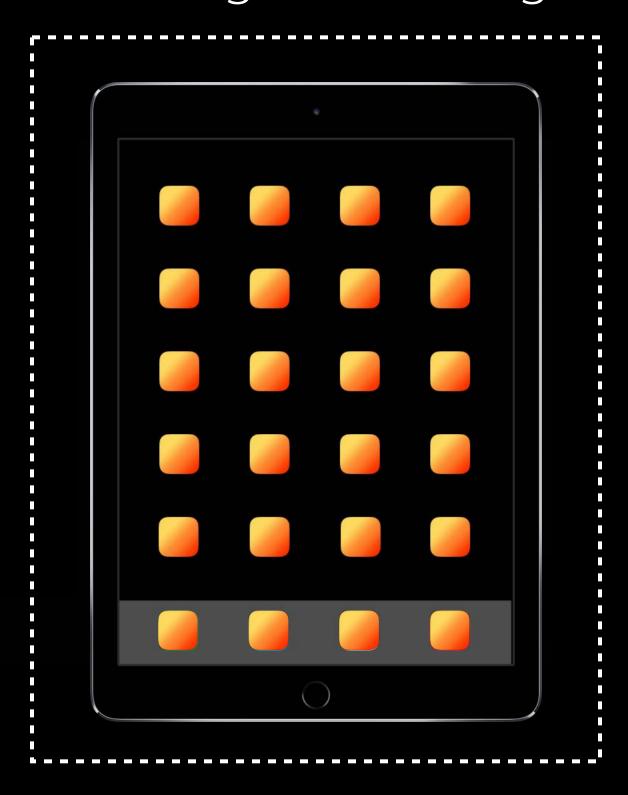
Working Set = 3 Pages



Working Set = 3 Pages

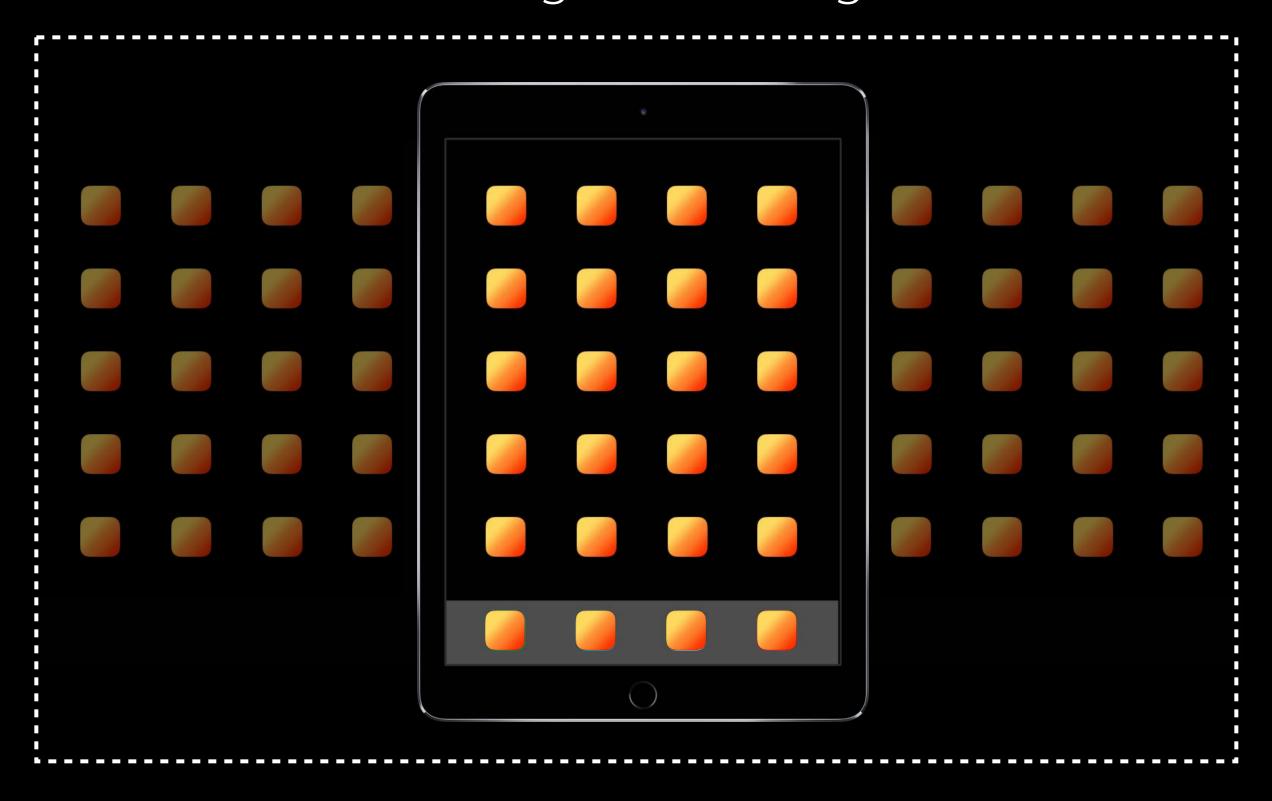


Working Set = 1 Page



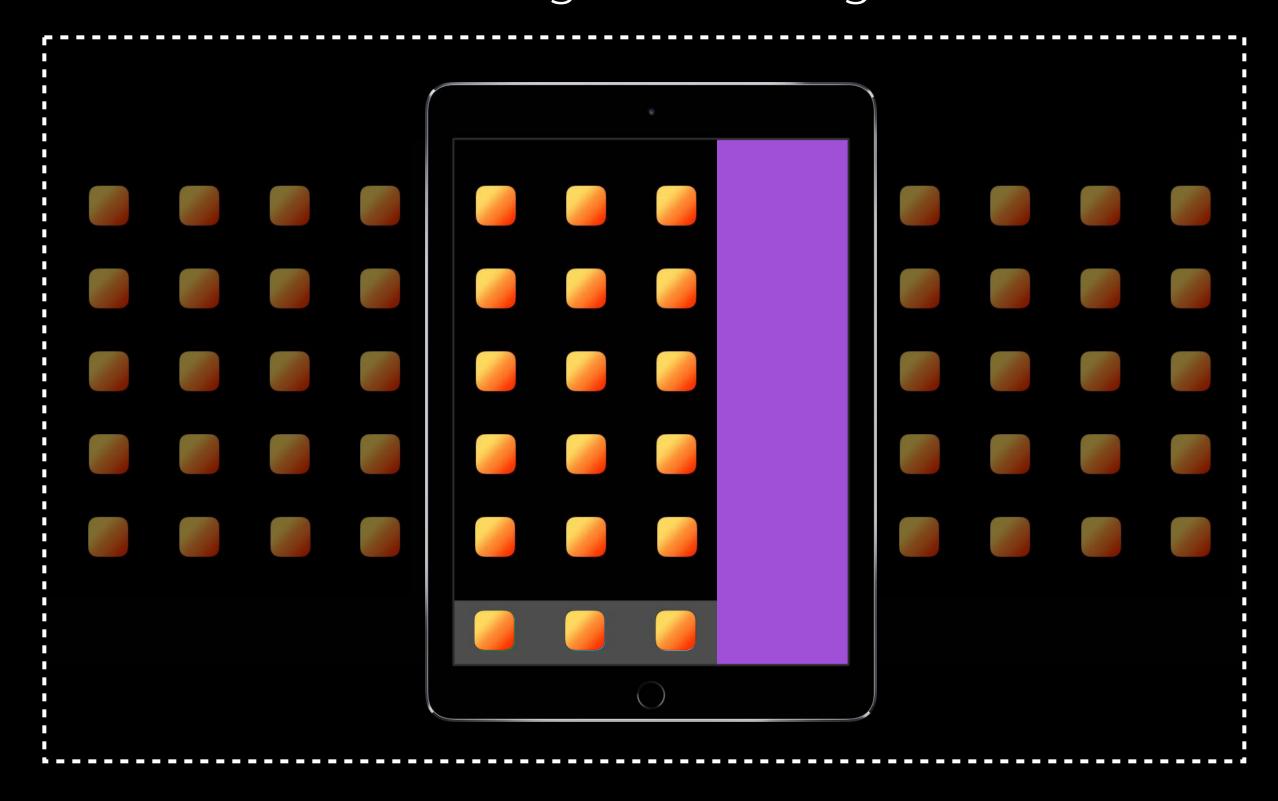
System IconReel Free

Working Set = 3 Pages

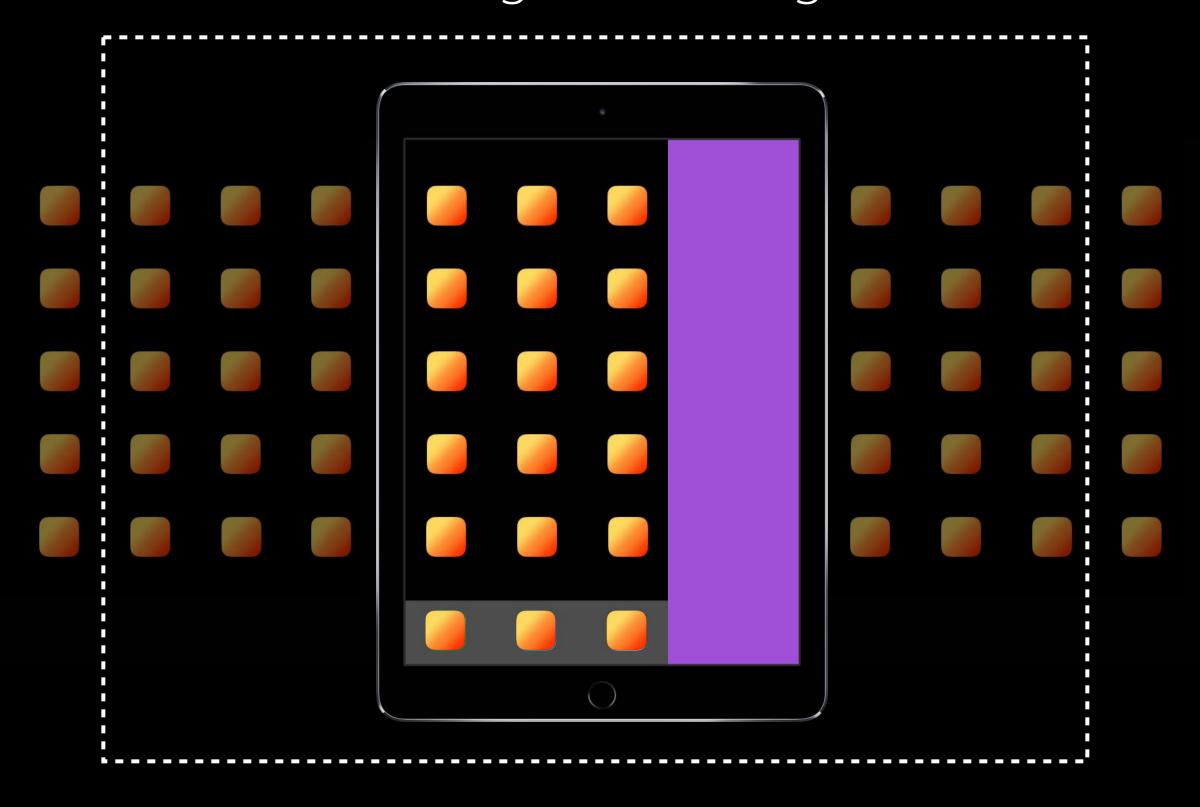


System IconReel Free

Working Set = 3 Pages



Working Set = 3 Pages



Managing Memory Growth

Memory warnings

Managing Memory Growth Memory warnings

Happens when

- The system is under memory pressure
- Your process is approaching its memory limit

Managing Memory Growth

Memory warnings

Happens when

- The system is under memory pressure
- Your process is approaching its memory limit

What to do

- Remove anything not in the working set
 - Clear cached data
 - Release images
 - Release view controllers

Memory Warnings APIs

In your UIApplicationDelegate and UIViewControllers

- -[UIApplicationDelegate applicationDidReceiveMemoryWarning:]
- -[UIViewController didReceiveMemoryWarning]

Anywhere in an application process

UIApplicationDidReceiveMemoryWarningNotification

In libraries and extensions

DISPATCH_SOURCE_TYPE_MEMORYPRESSURE

Managing Memory Growth NSCache

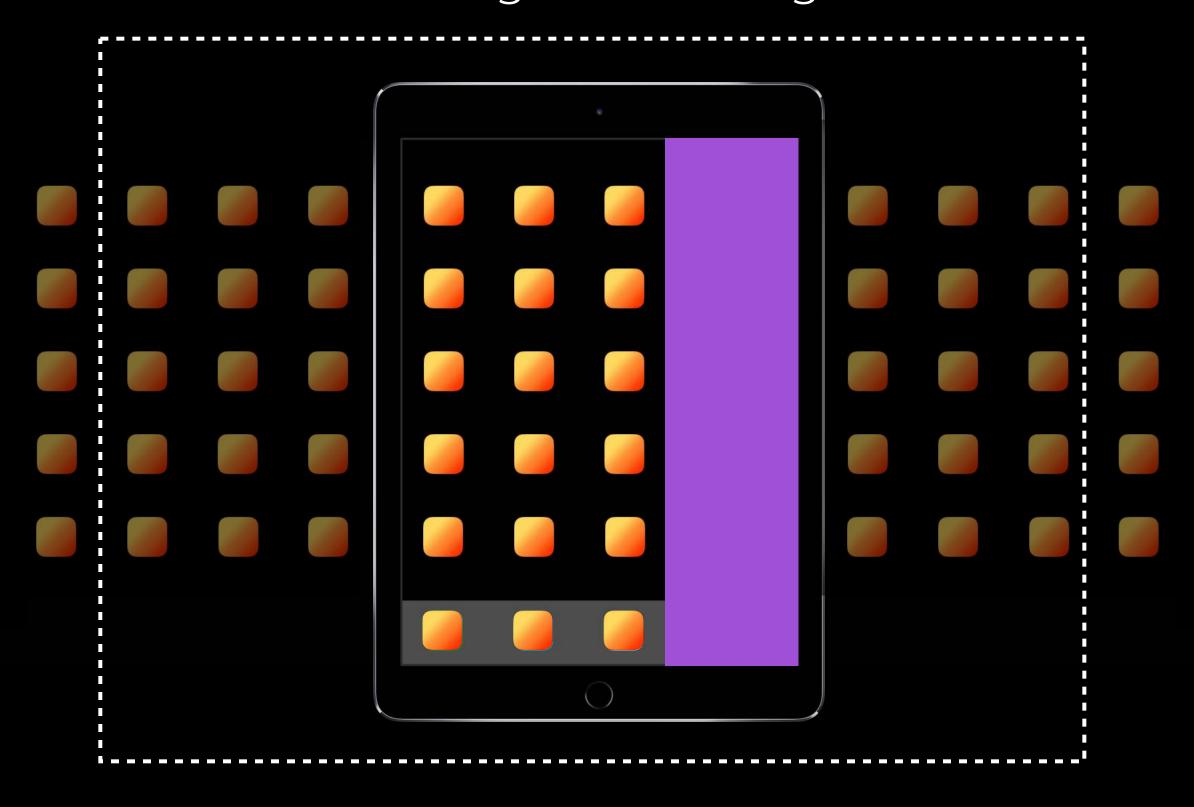
NSDictionary-like

Ideal for objects that can be regenerated on demand

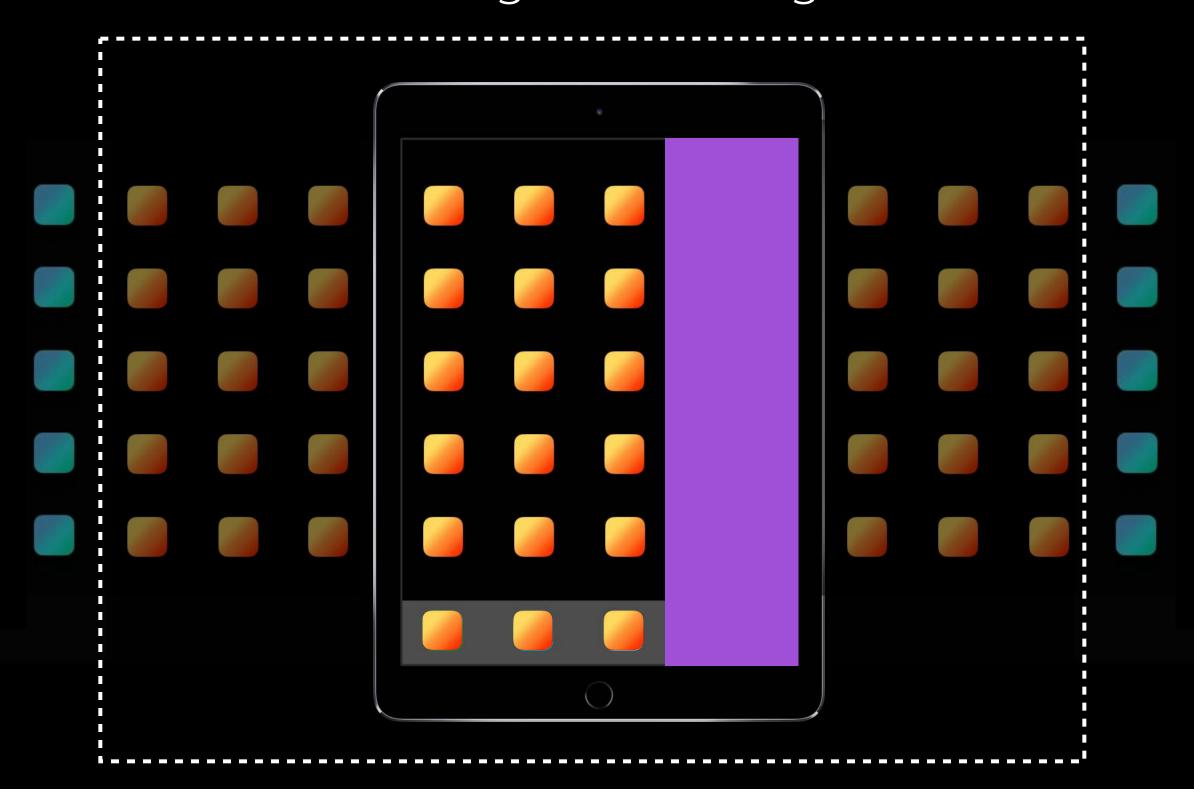
Trims under memory pressure

Trims for application lifecycle changes

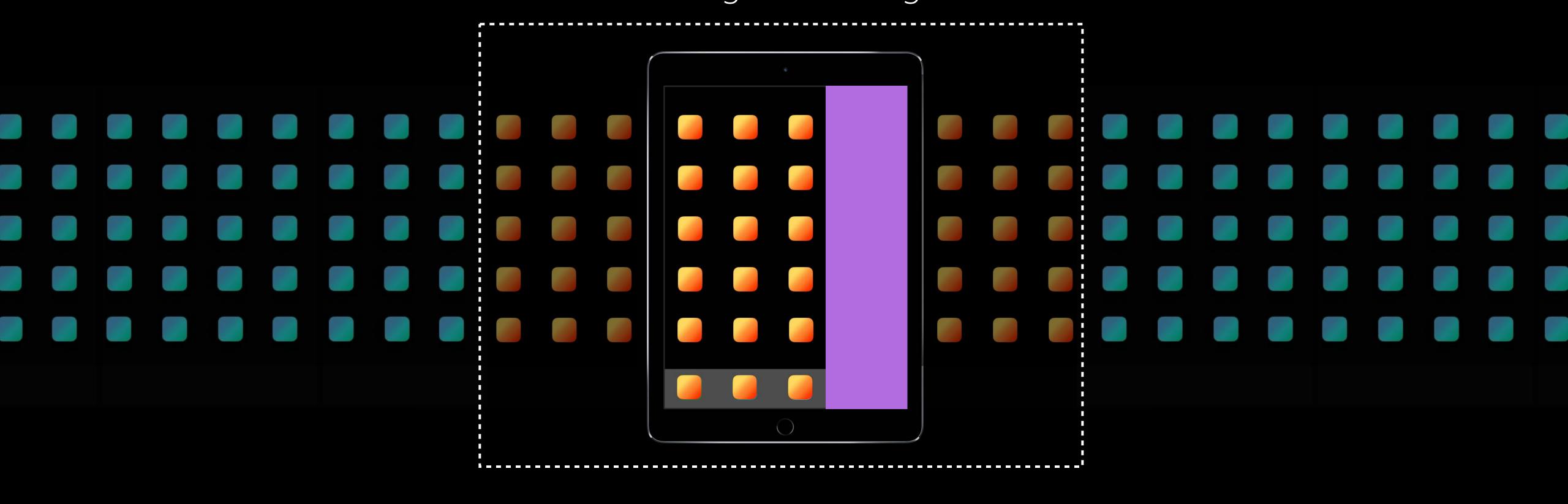
Working Set = 3 Pages



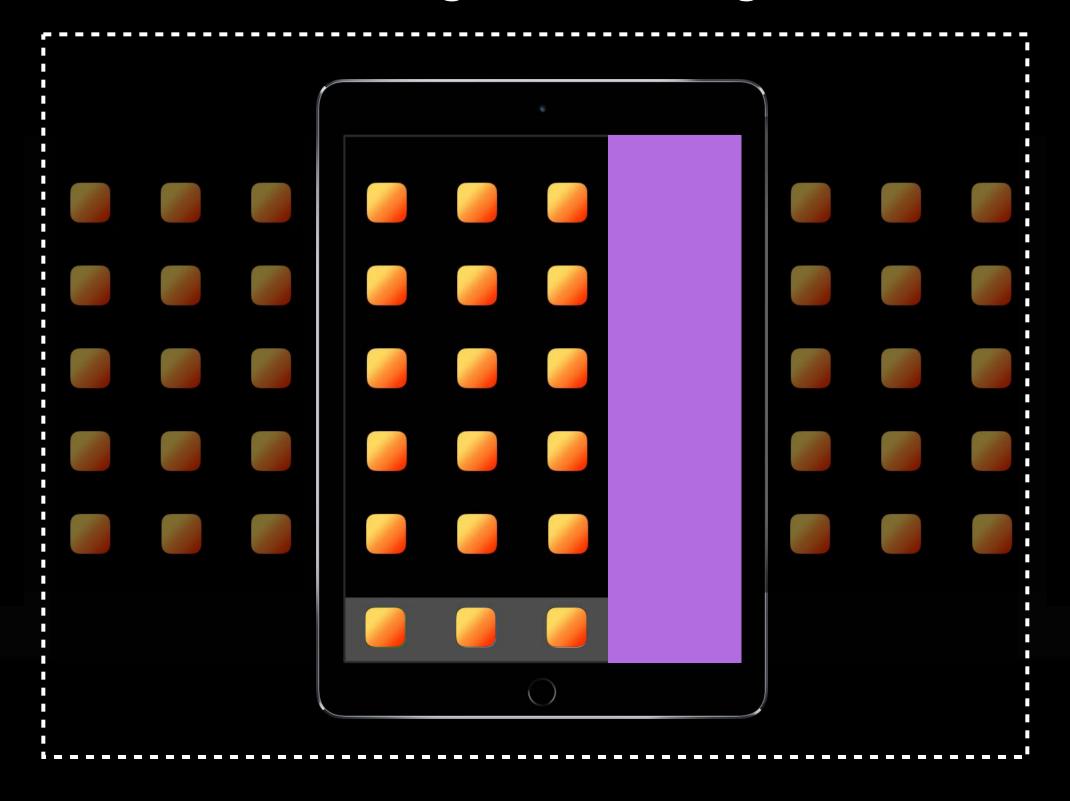
Working Set = 3 Pages



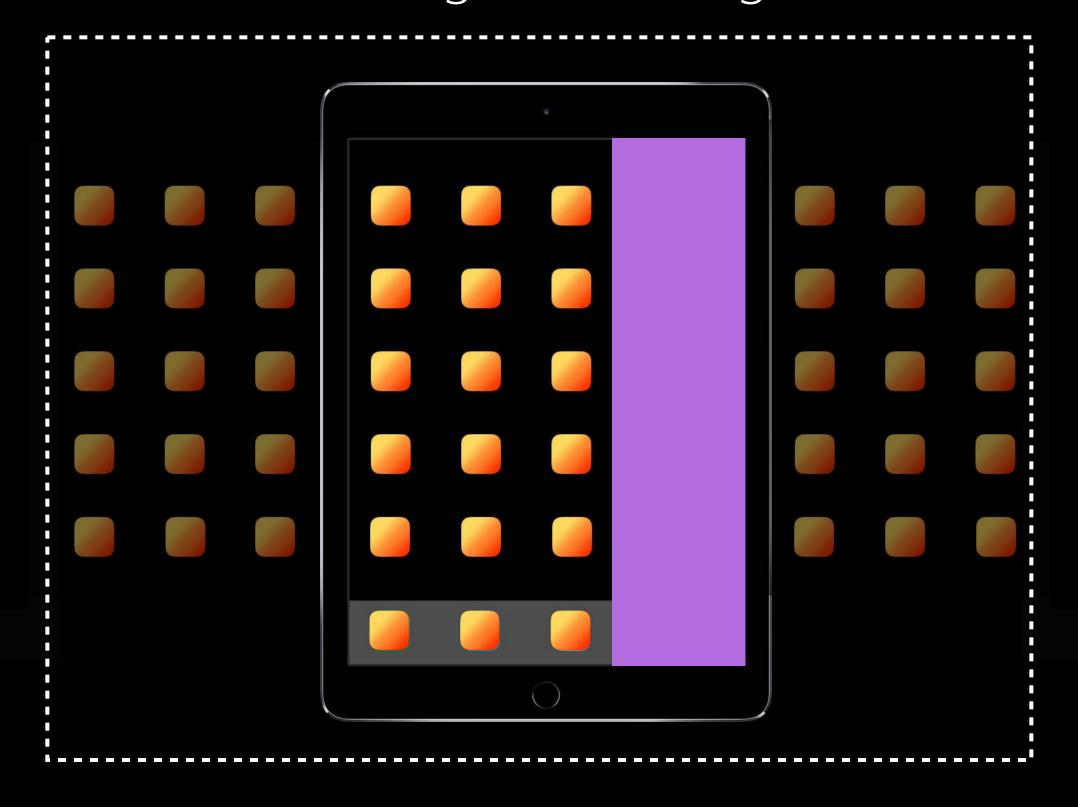
Working Set = 3 Pages



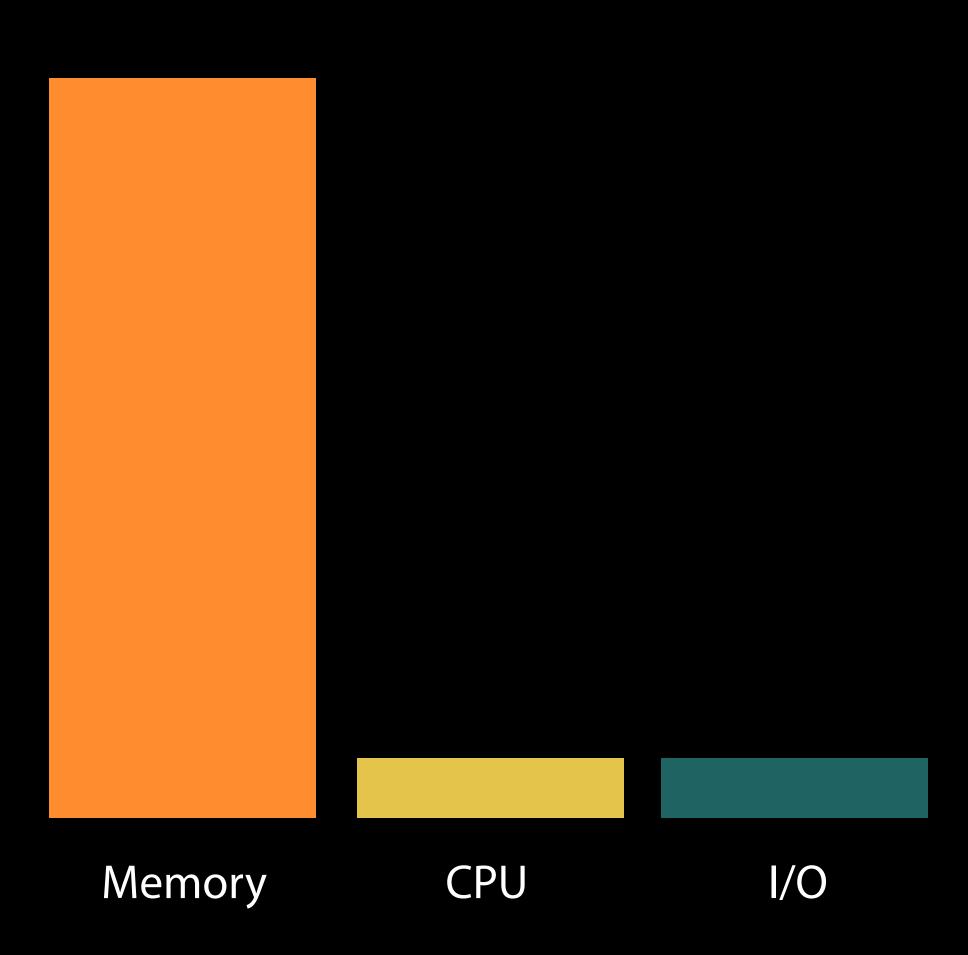
Working Set = 3 Pages



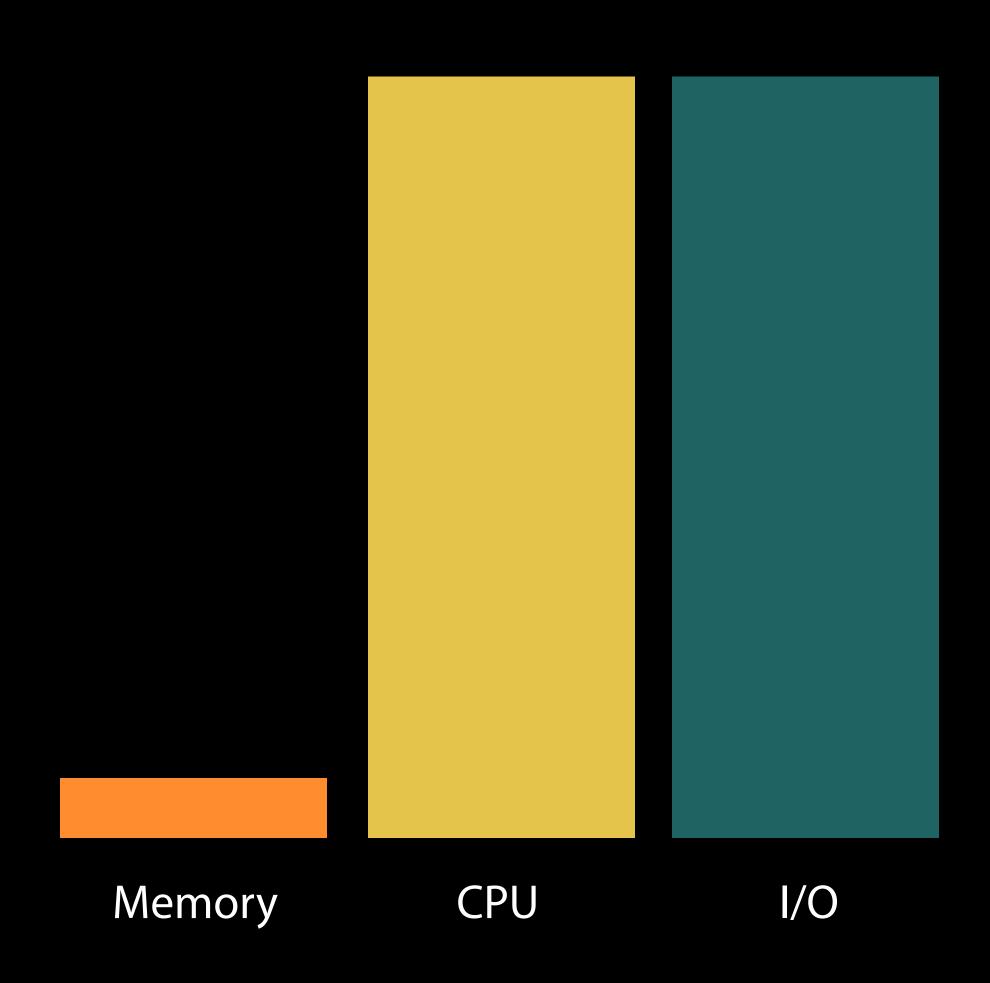
Working Set = 3 Pages



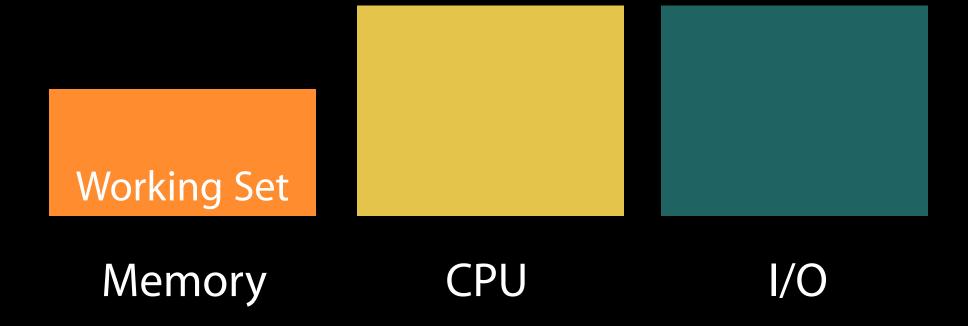
Recap



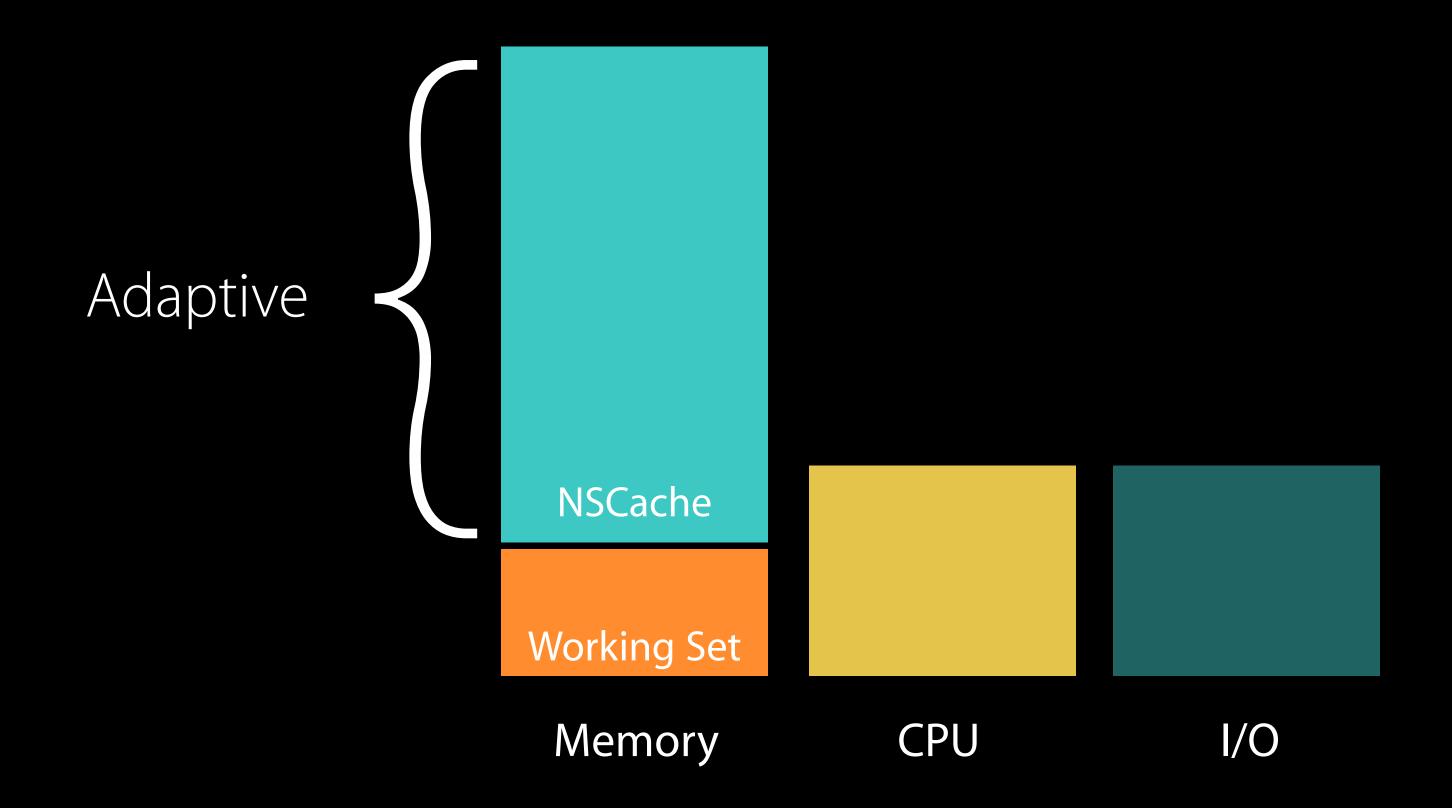
Recap



Recap

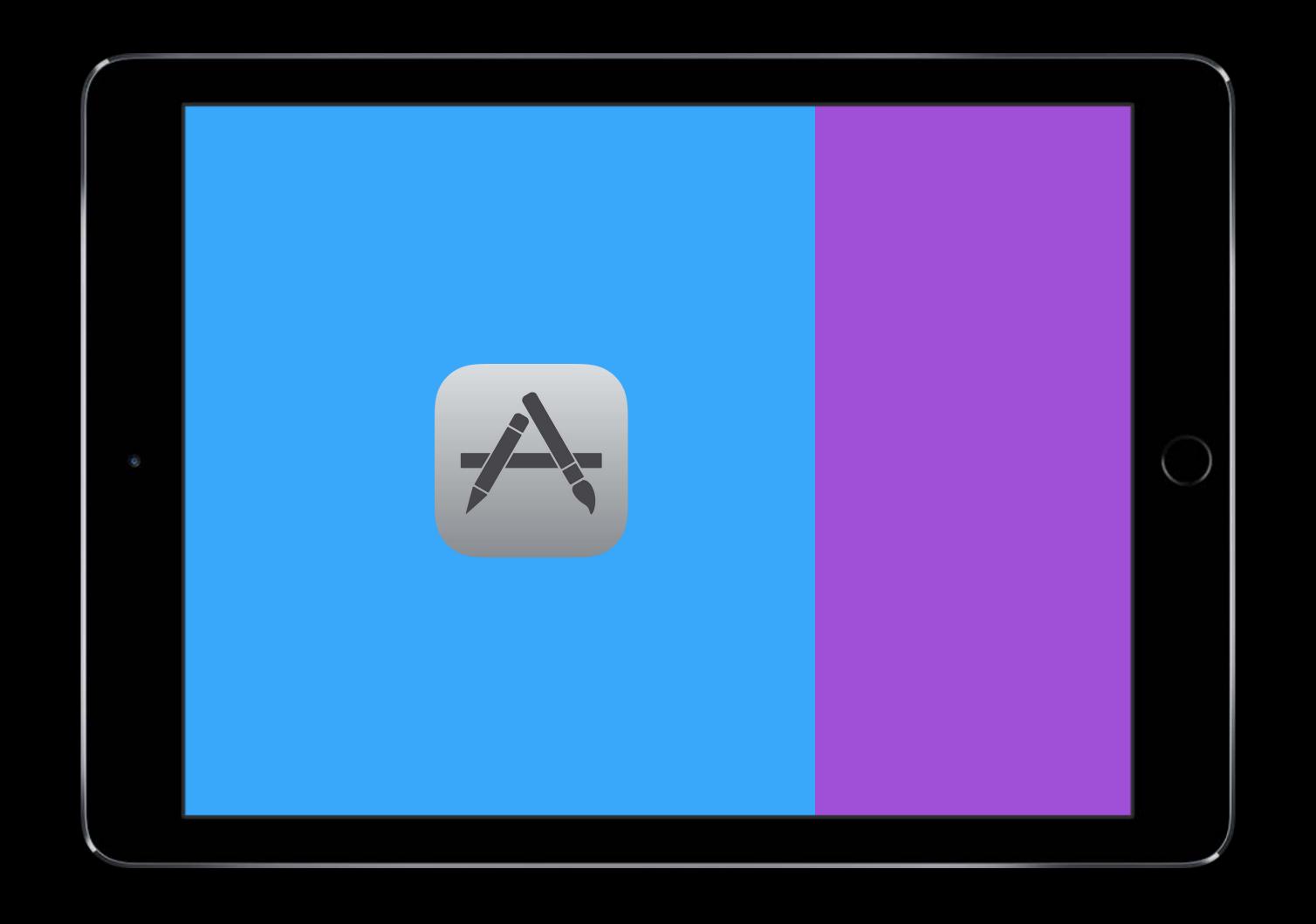


Recap



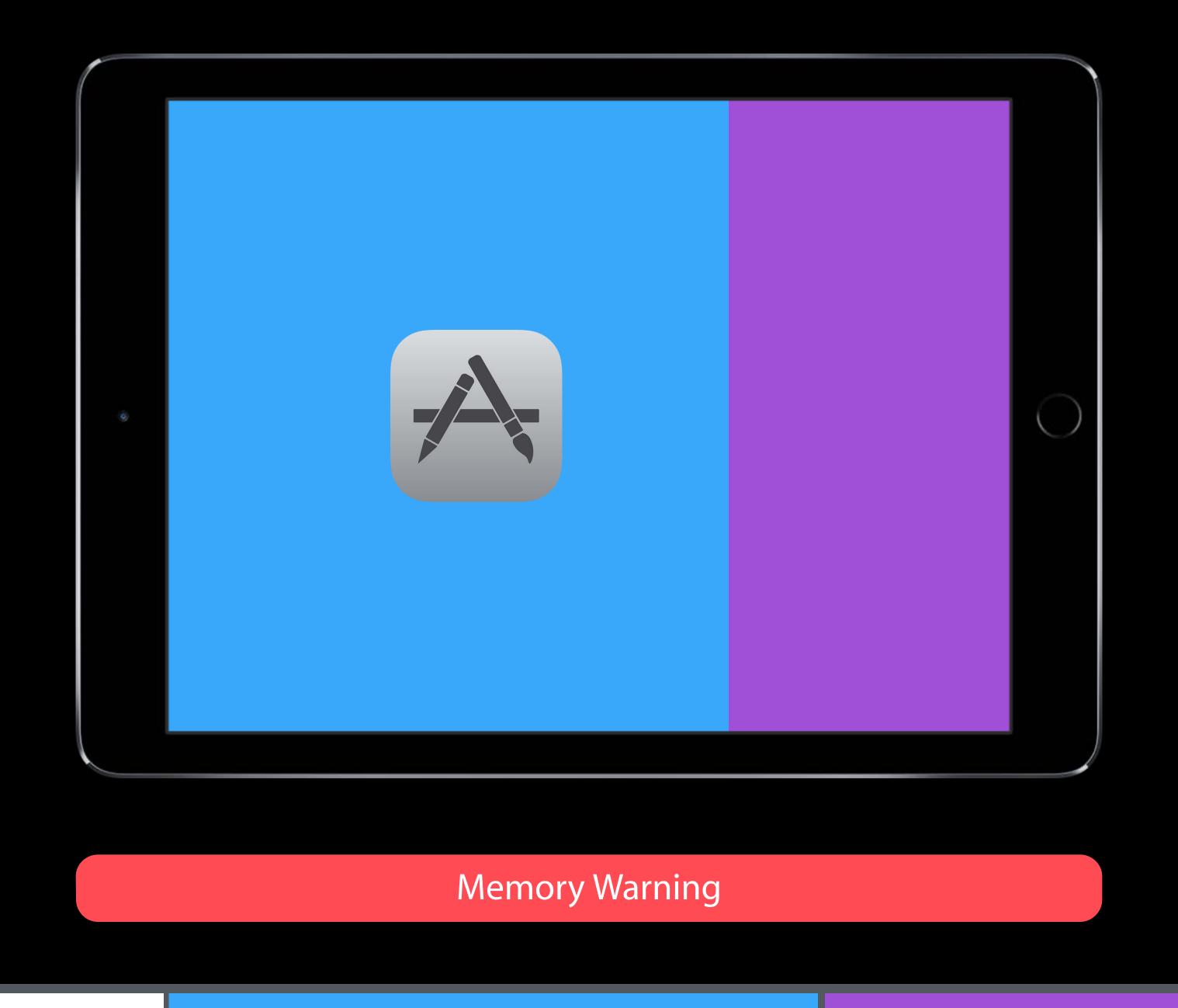
Leveraging the Virtual Memory System

Jon Drummond iOS SpringBoard Engineer

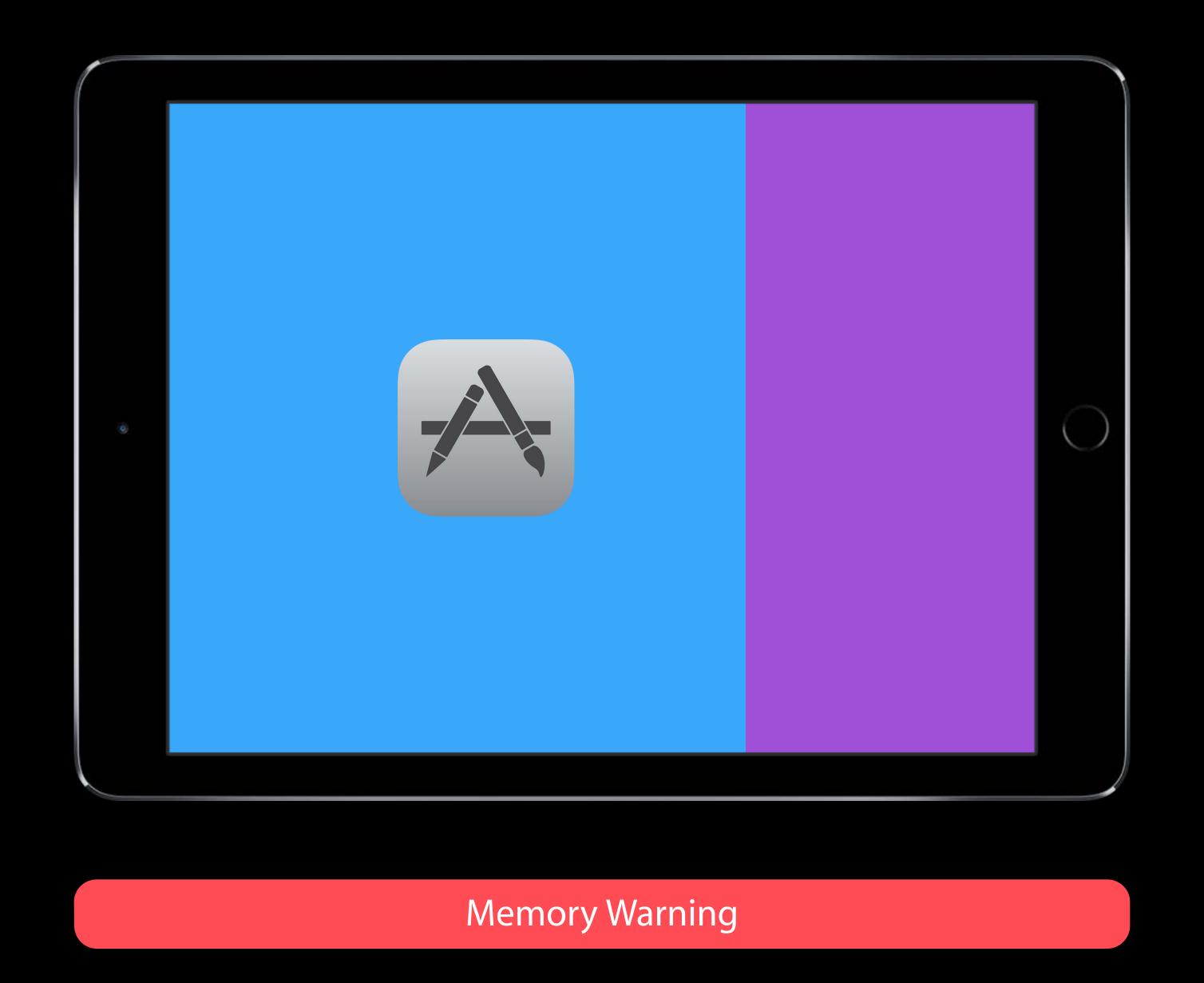


Primary App

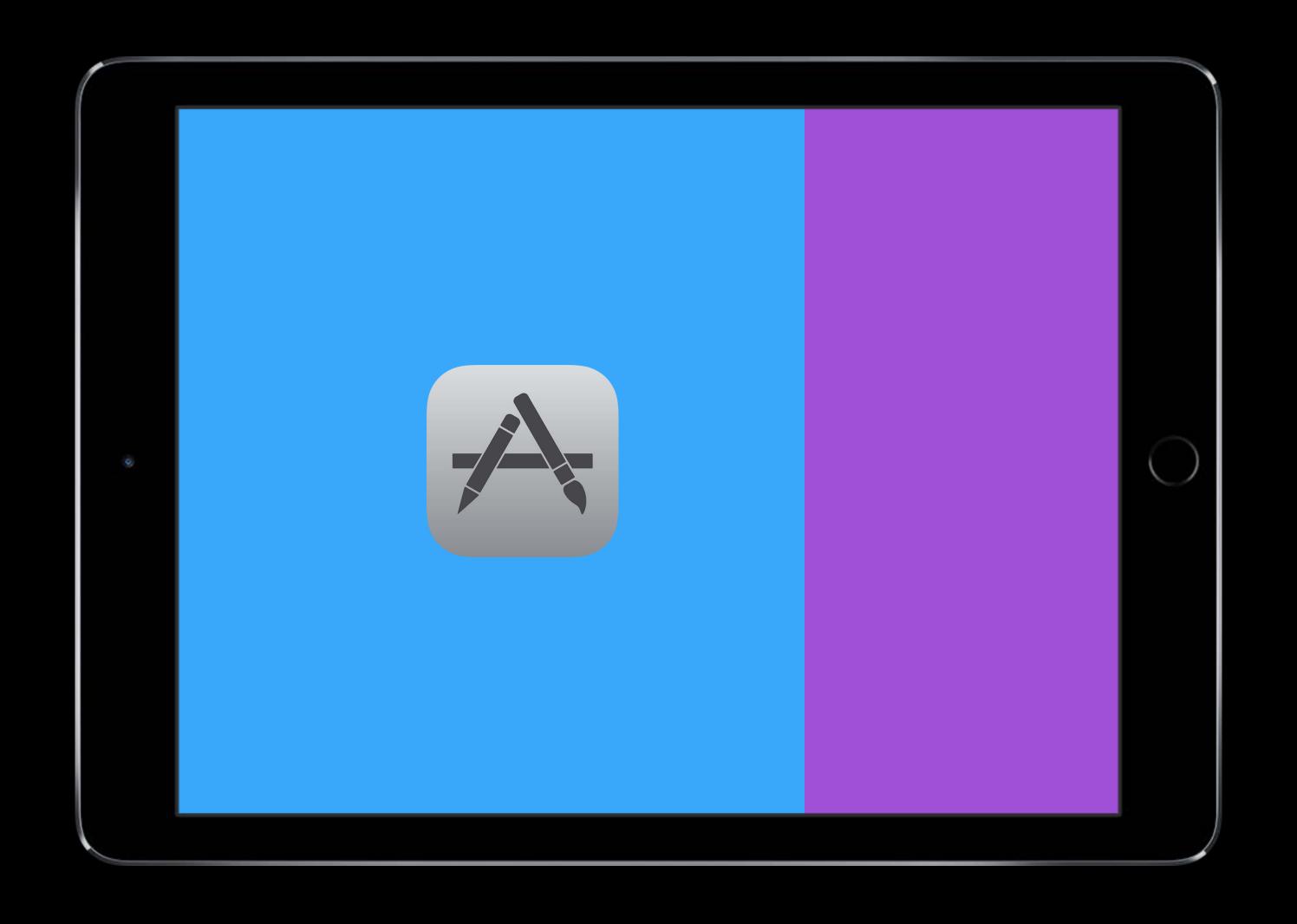
System



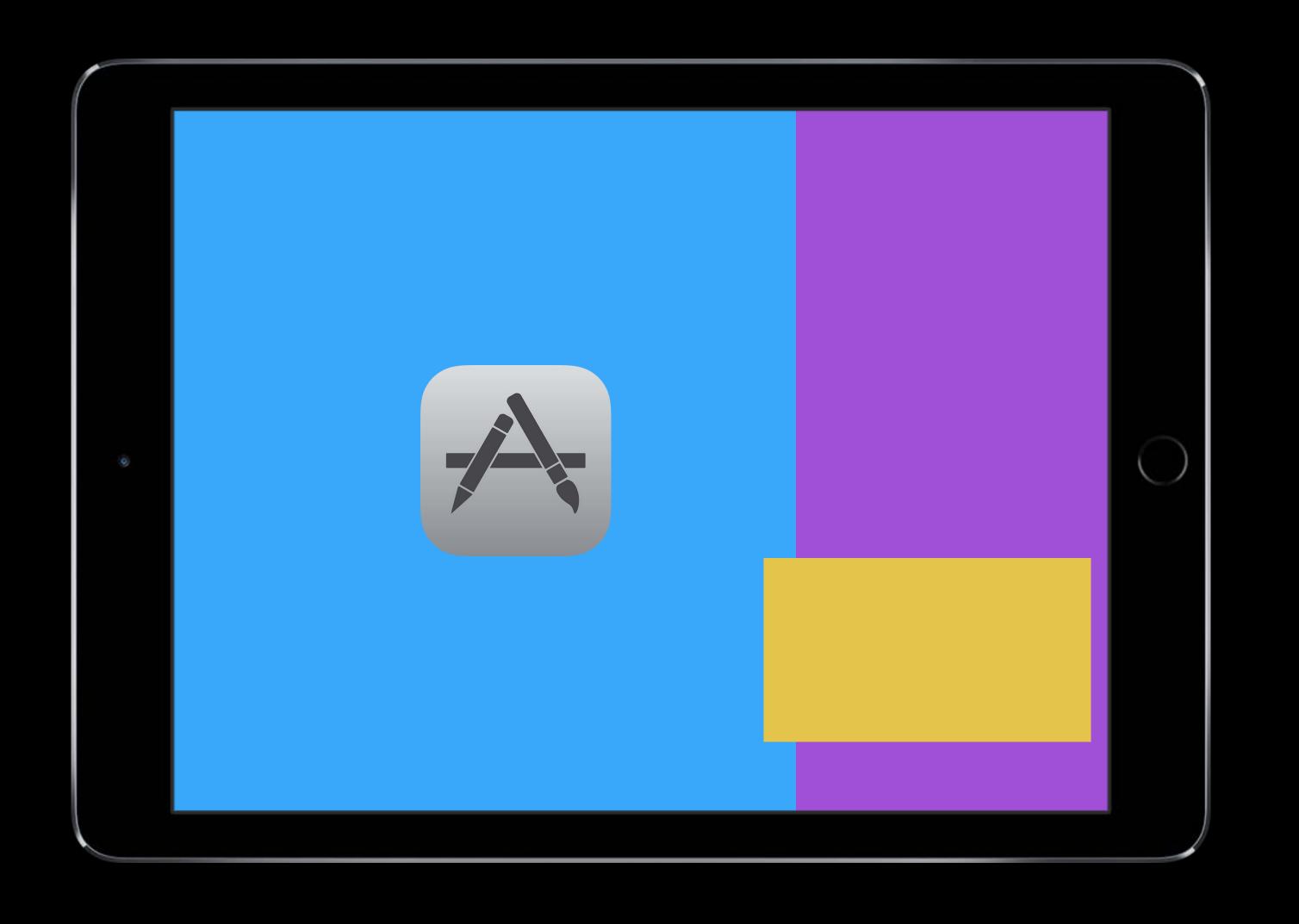
Primary App



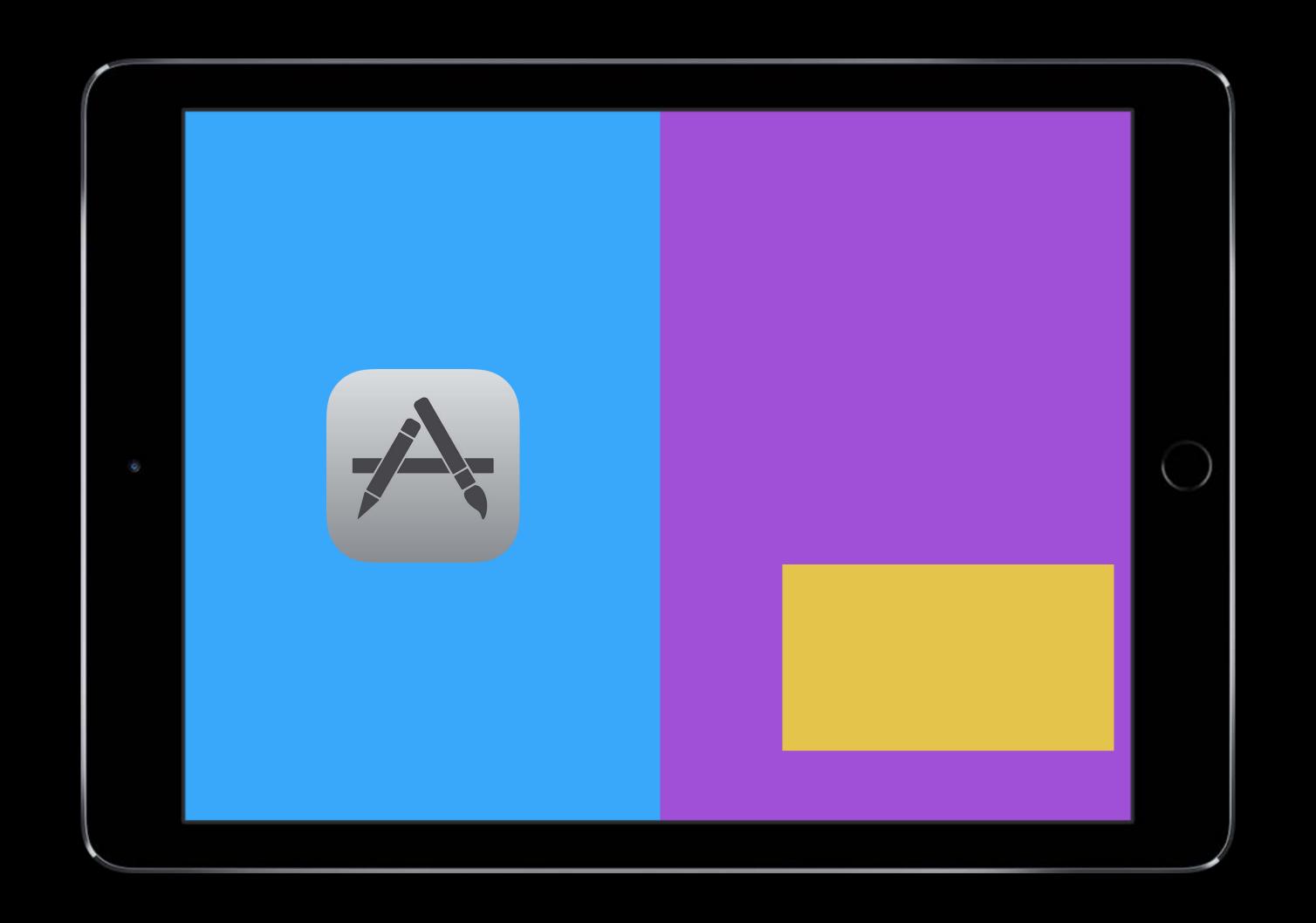
Primary App



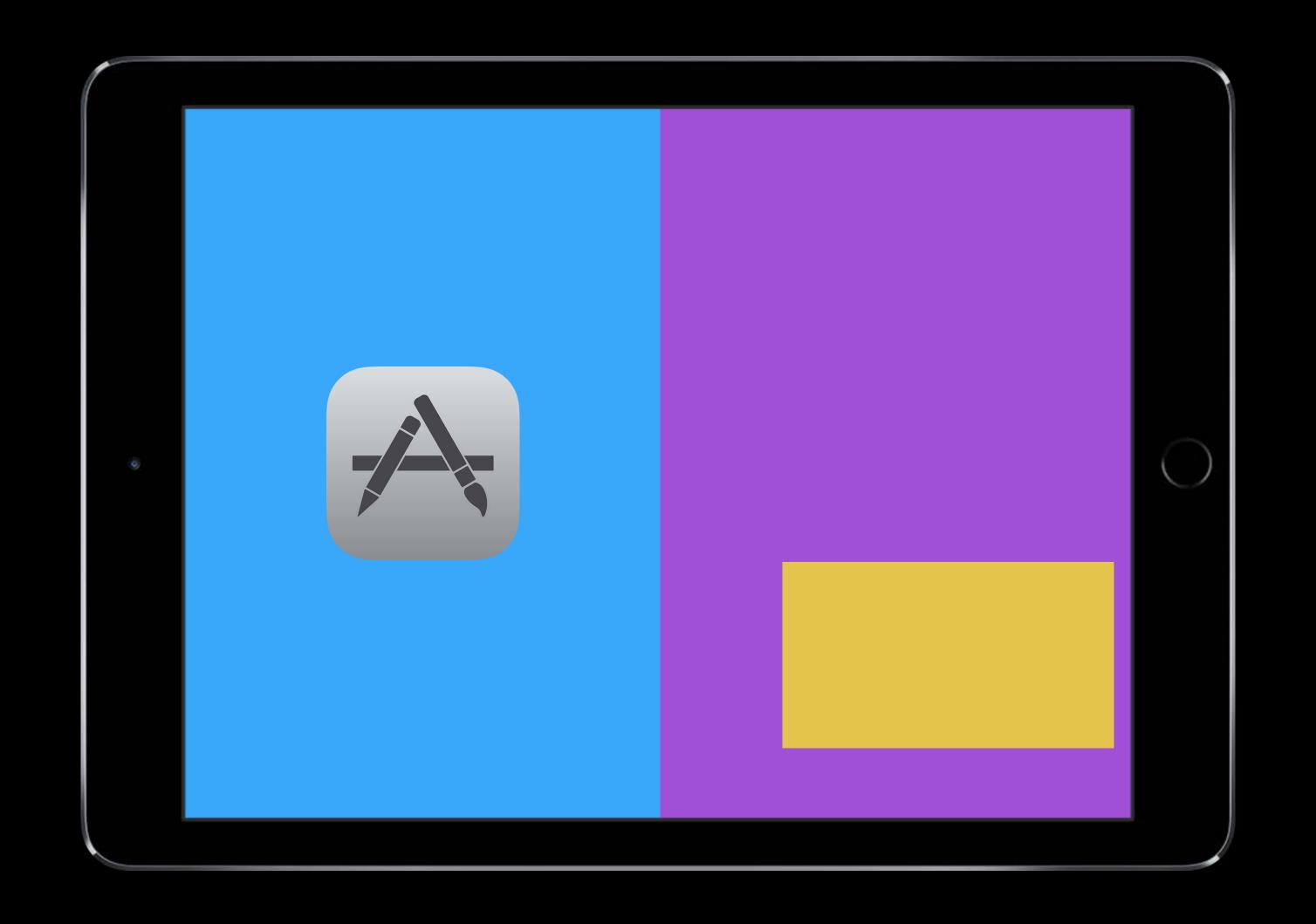
Primary App



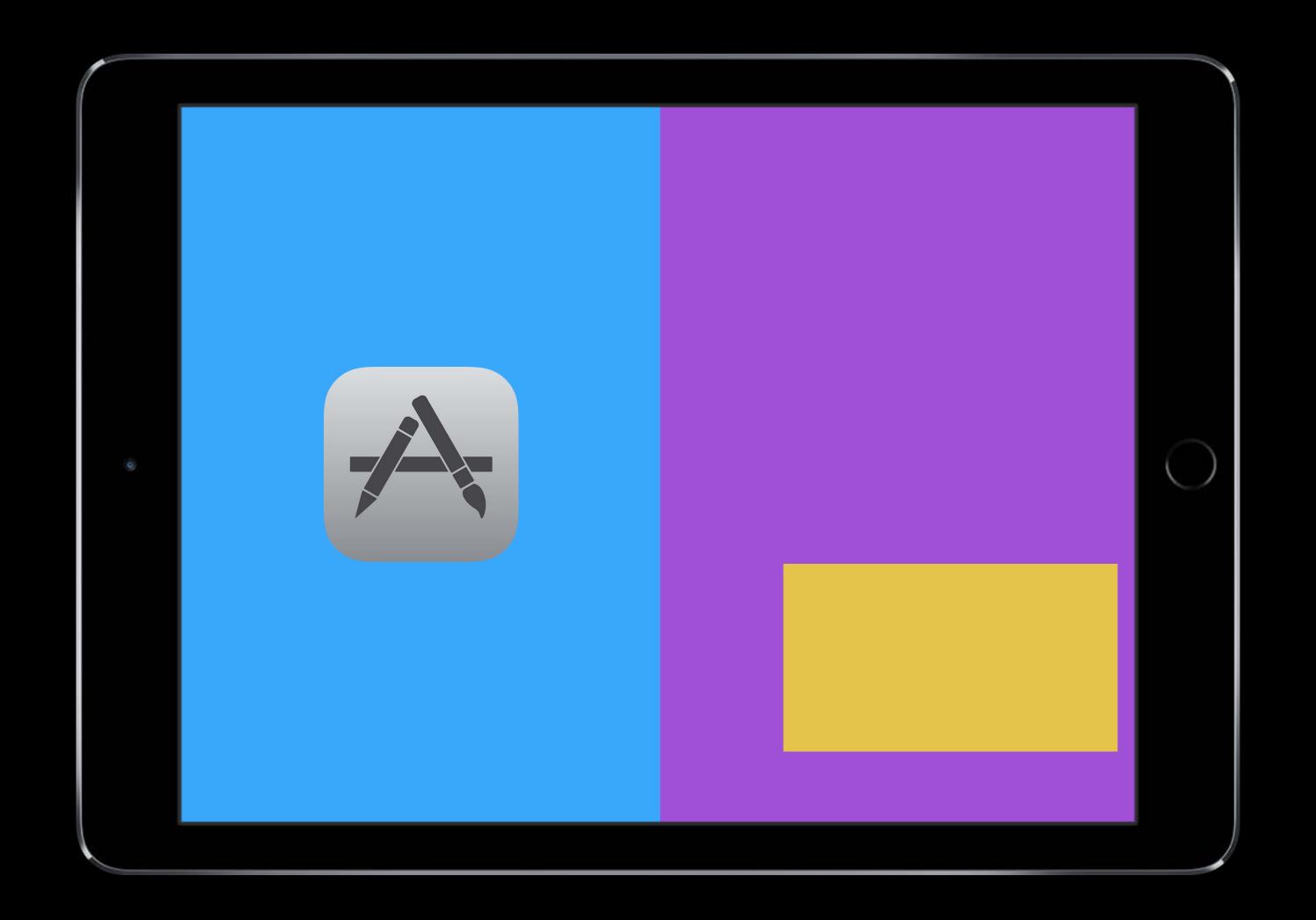
System Primary App Secondary App PiP



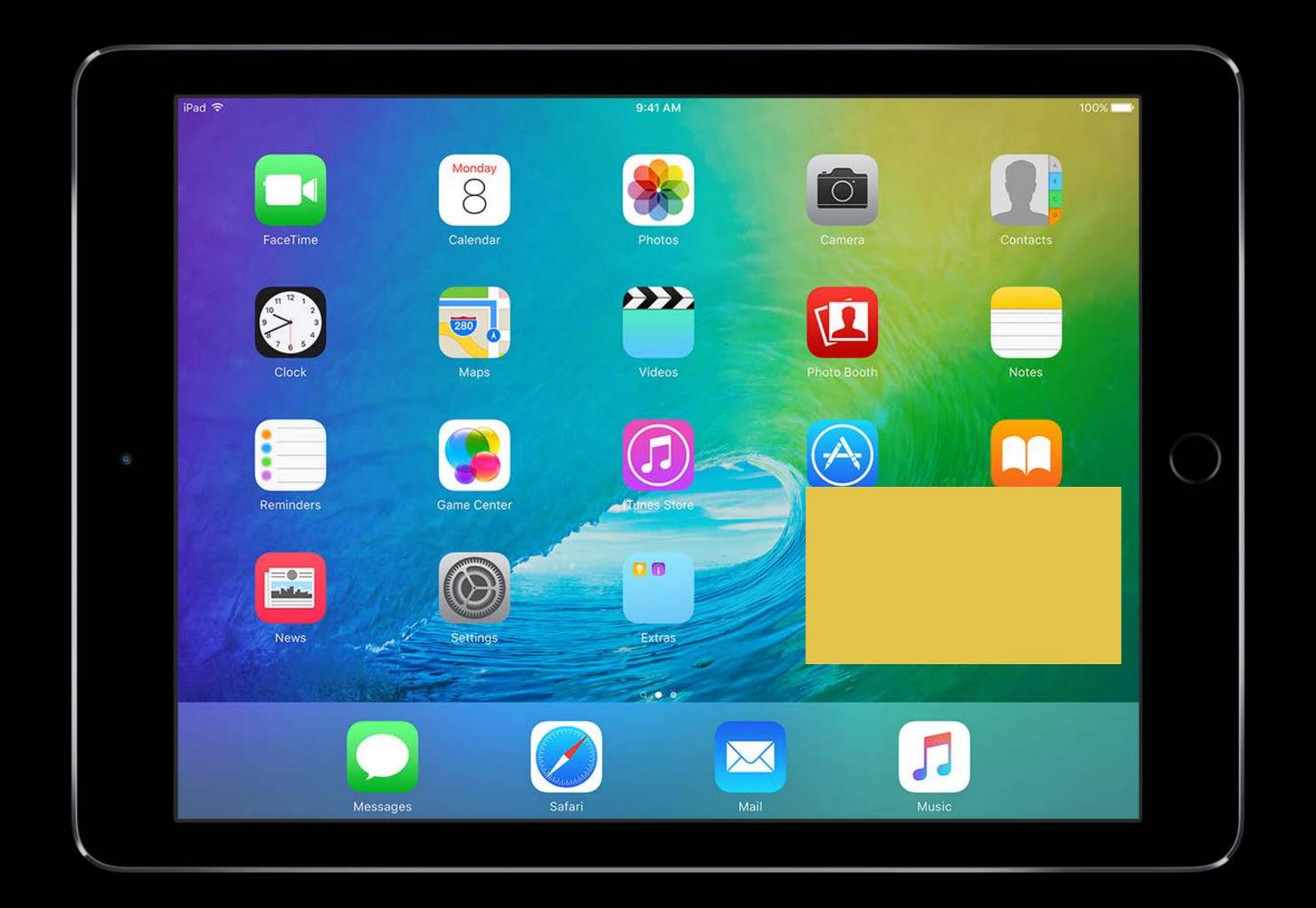
System Primary App Secondary App PiP



System Primary App Secondary App PiP





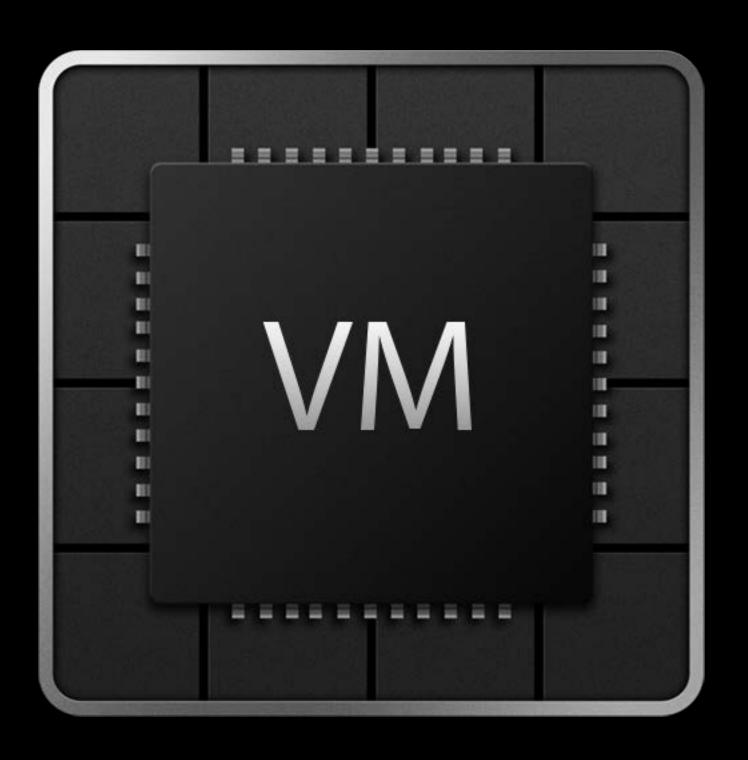


Secondary App

PiP

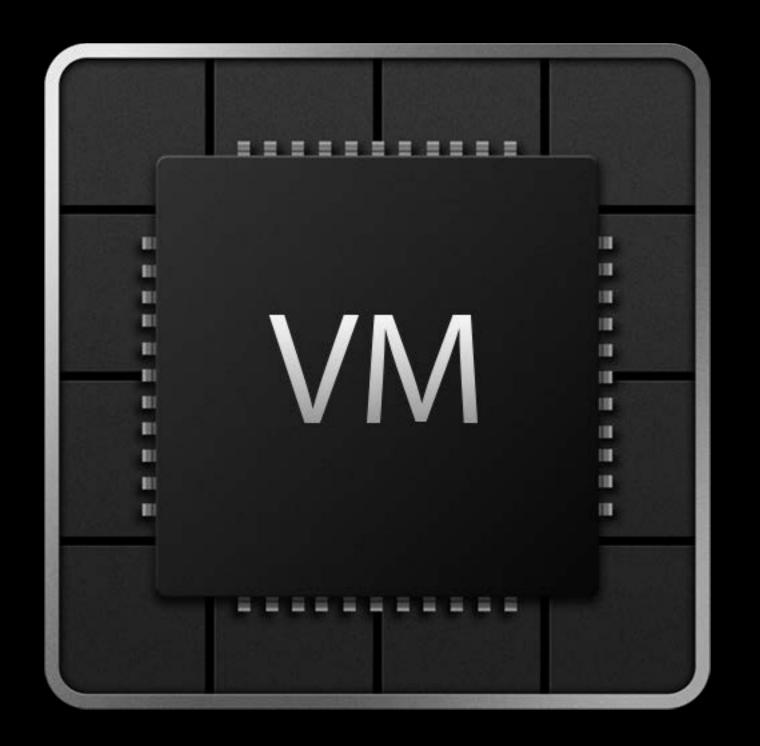
"The world outside your process should be regarded as hostile and bent upon your destruction."

Me, almost everyday



Memory is the most constrained resource on iOS

No swap

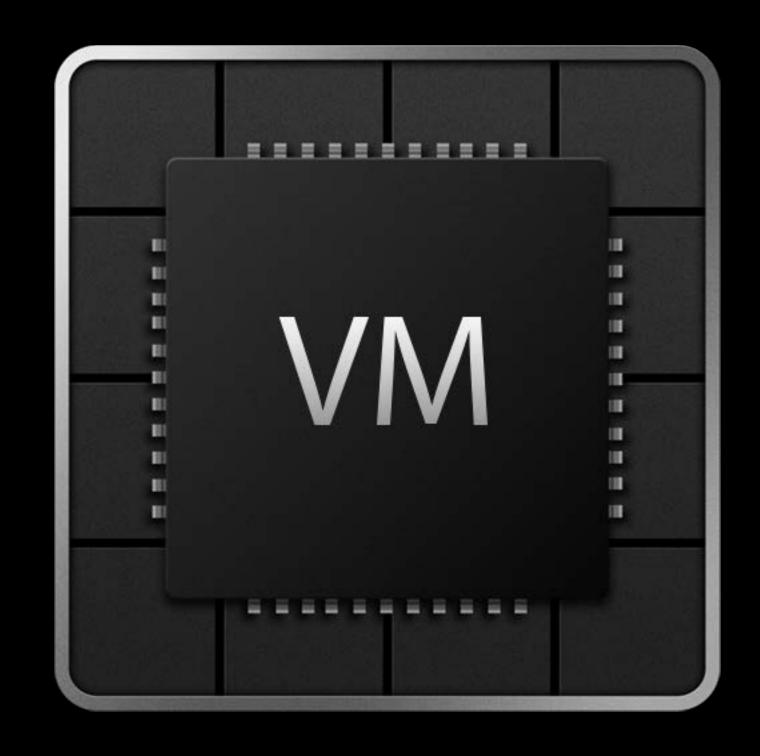


Memory is the most constrained resource on iOS

No swap

System can require memory faster than it can be released

 CPU contention during memory warnings can slow or prevent apps from releasing it



Classification

Description

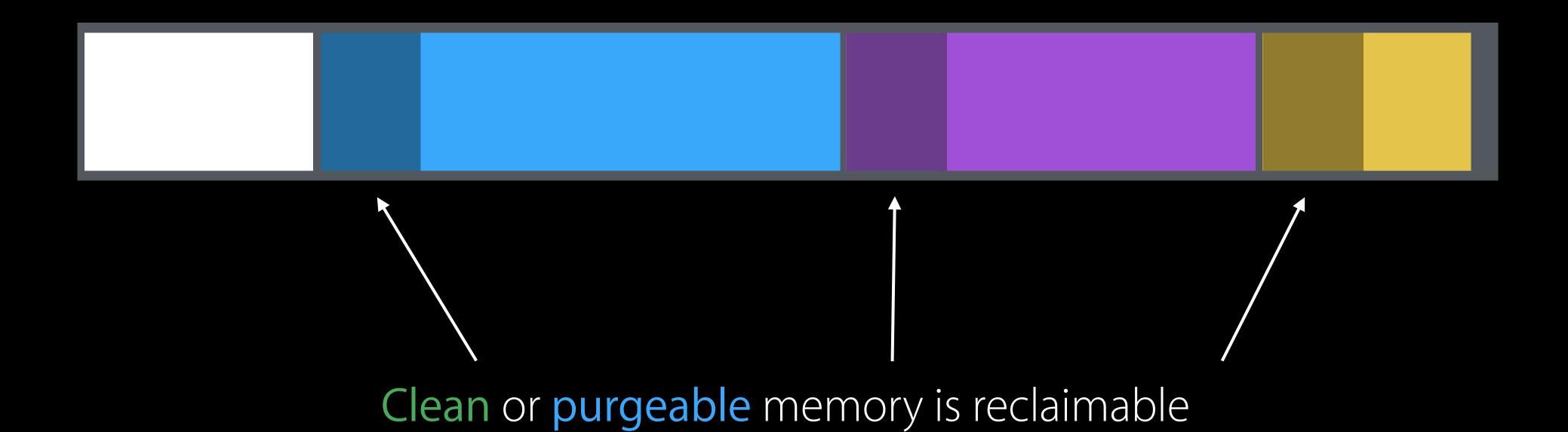
Reclaimable?

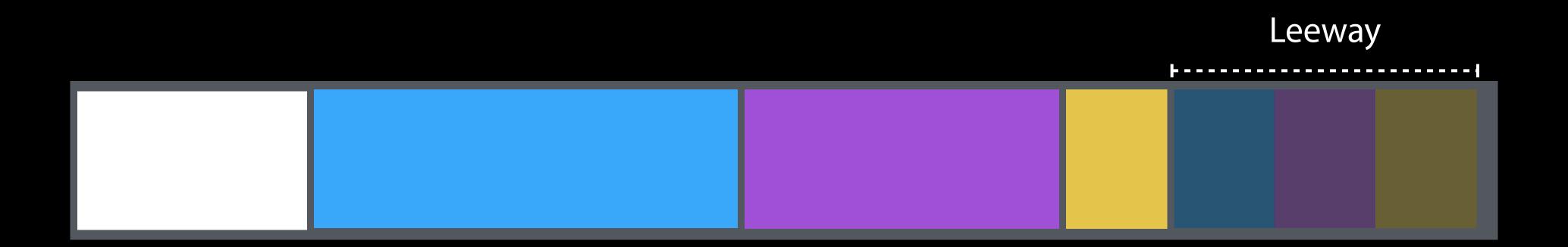
Classification Description Reclaimable?

Dirty Memory in active use

Classification	Description	Reclaimable?
Dirty	Memory in active use	
Purgeable	Otherwise-dirty memory designated as not in-use	

Classification	Description	Reclaimable?
Dirty	Memory in active use	
Purgeable	Otherwise-dirty memory designated as not in-use	
Clean	Read-only memory backed by files on disk	





Leveraging the Virtual Memory System Goals

Minimize dirty memory usage

Maximize purgeable and clean memory usage

iOS App Performance: Memory

WWDC12

Minimizing Dirty Memory

Minimizing Dirty Memory

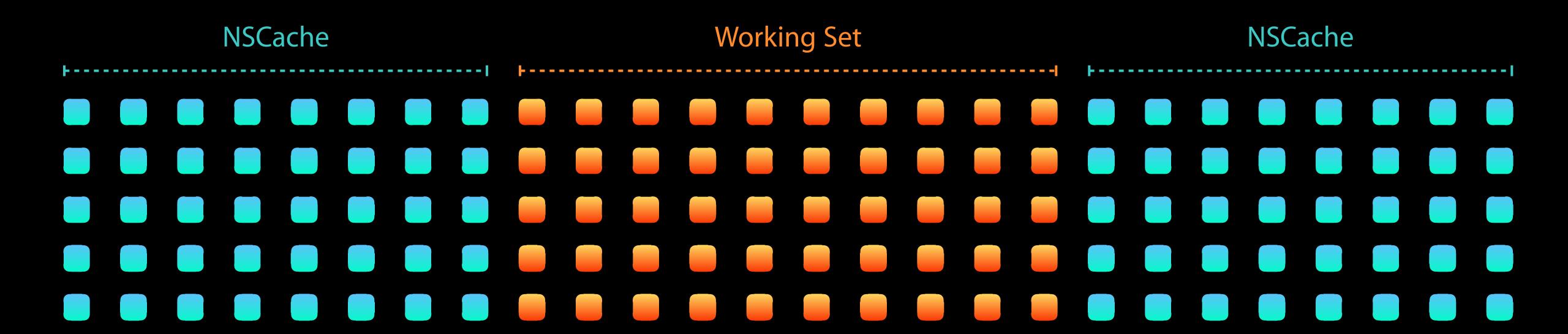
Use less of it!

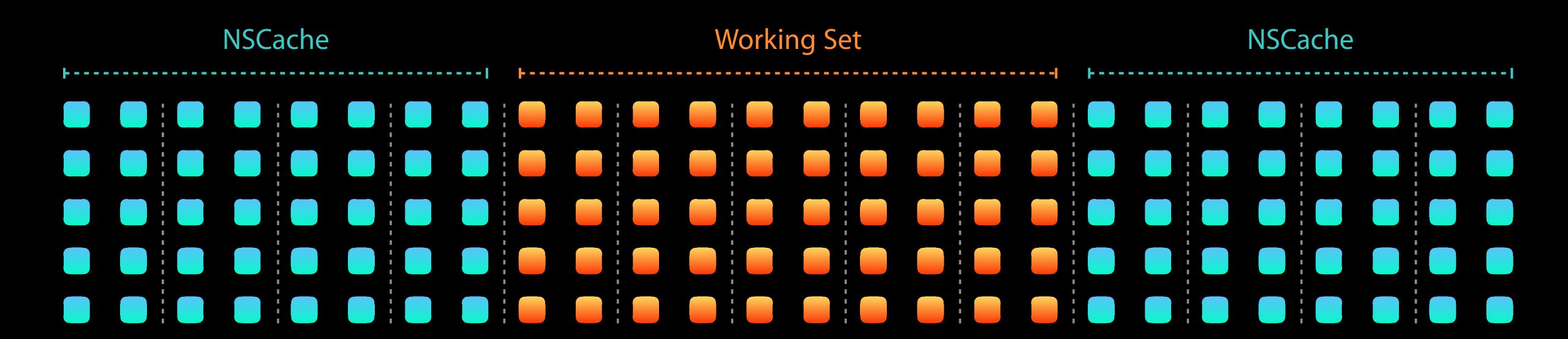
Minimizing Dirty Memory

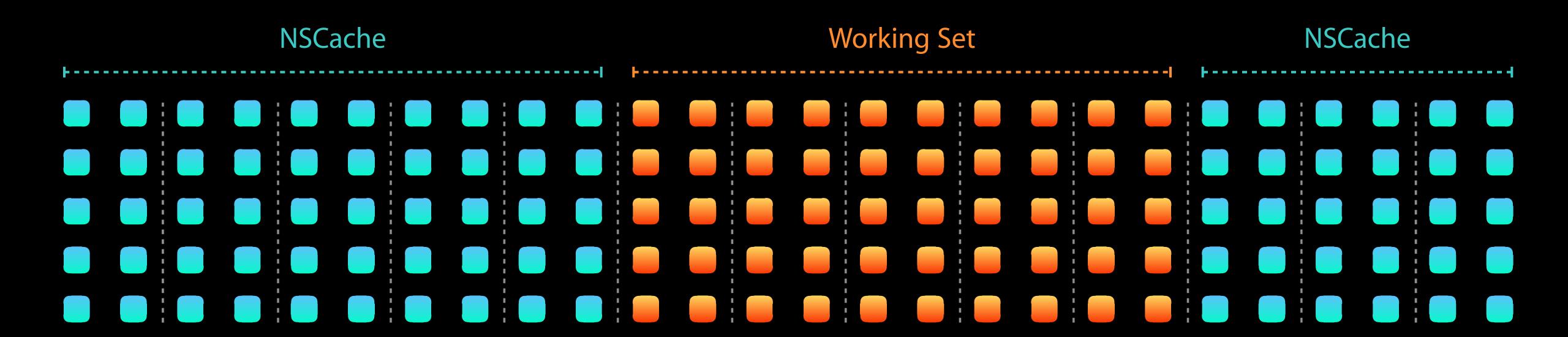
Use less of it!

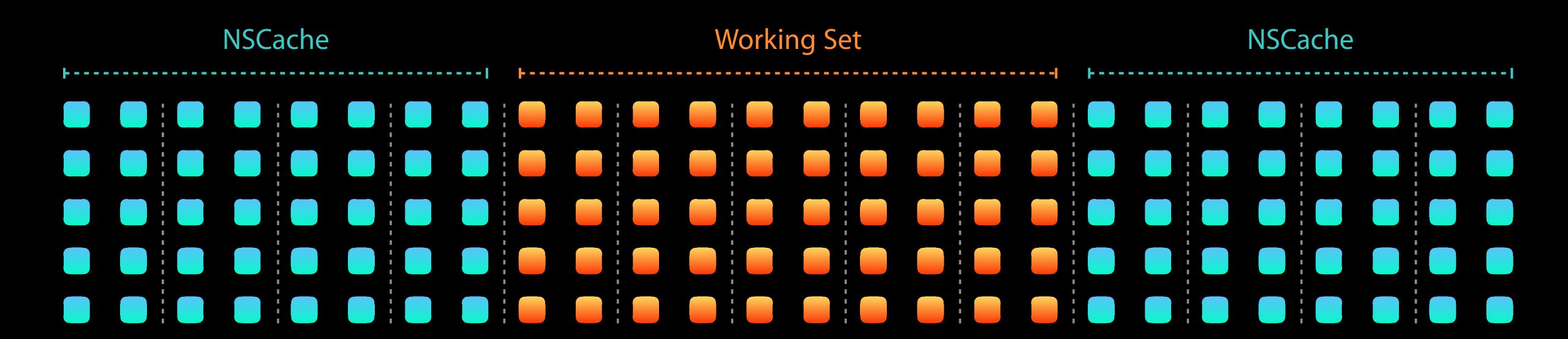
Classify otherwise dirty memory as purgeable

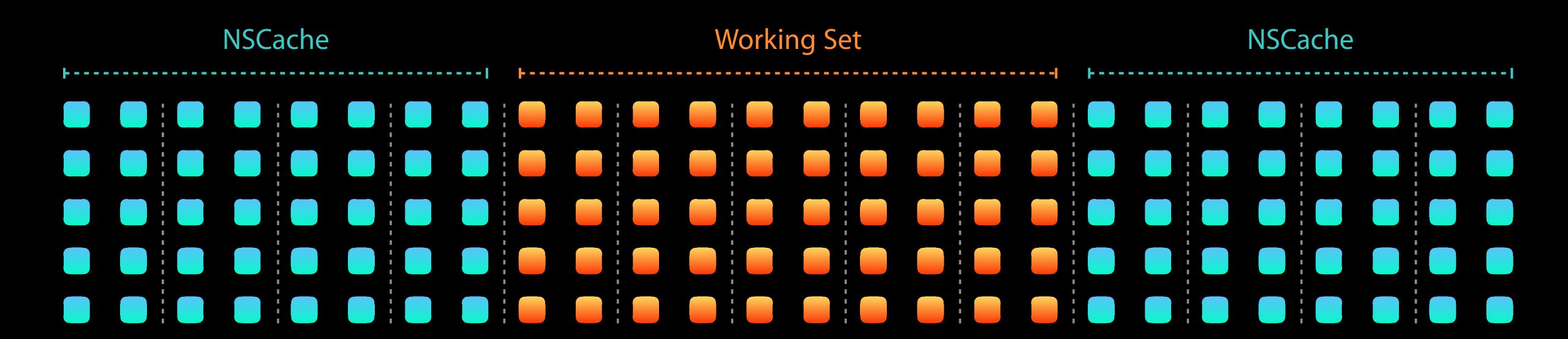
- Can be automatically reclaimed when not in use
- Best for "nice to have" data that can be recomputed

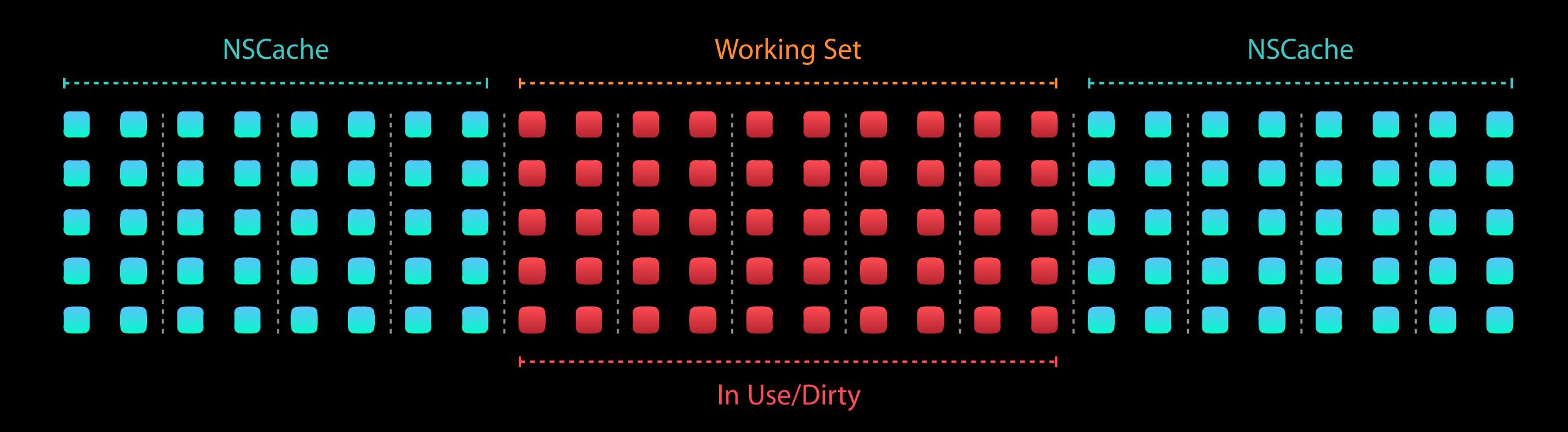


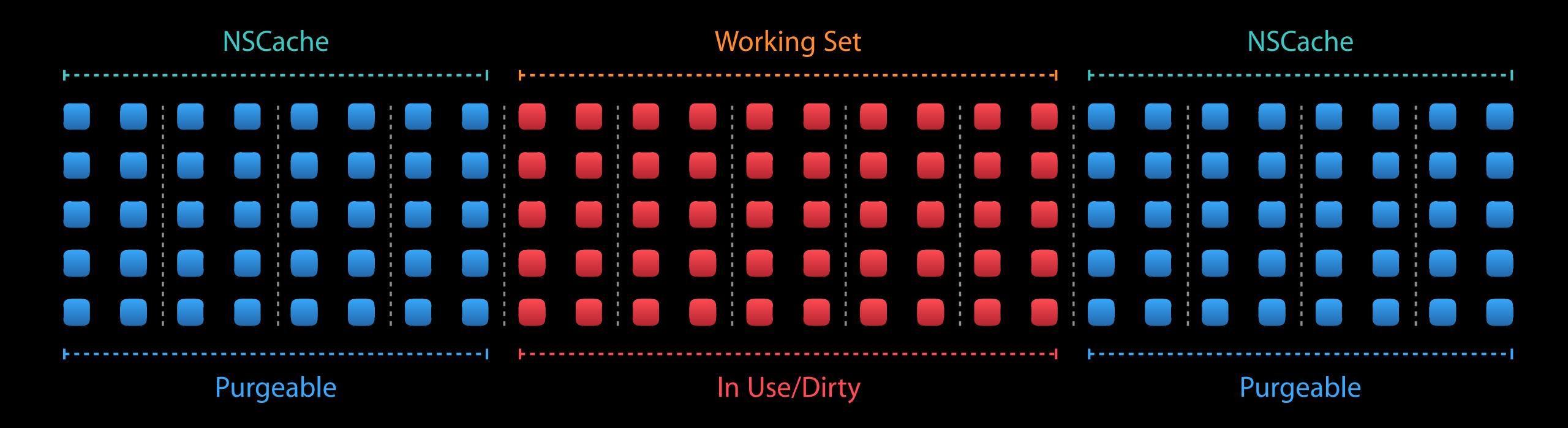


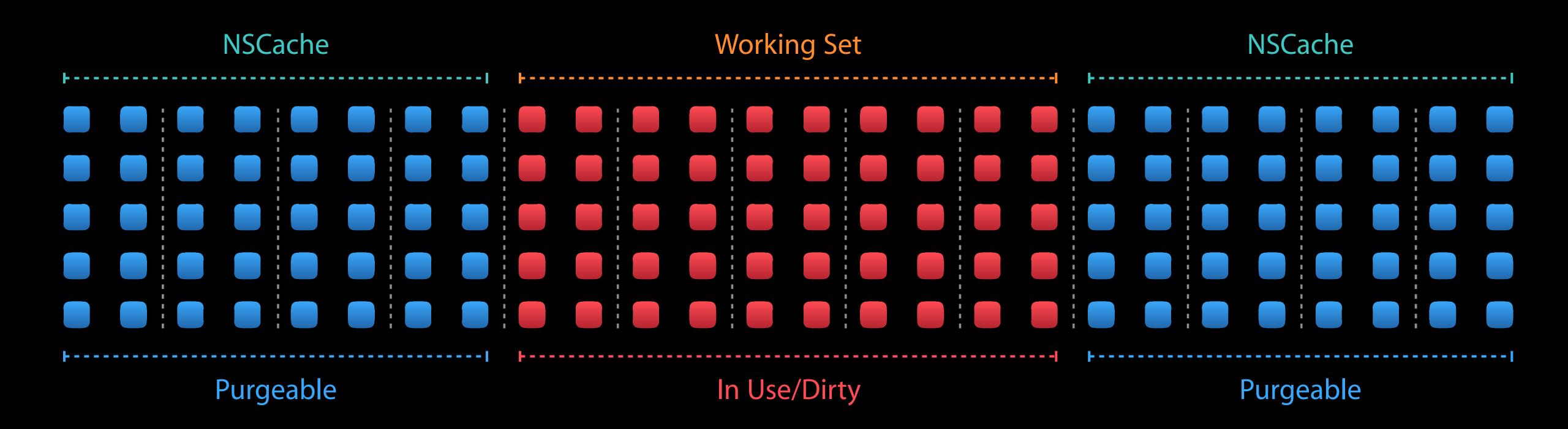


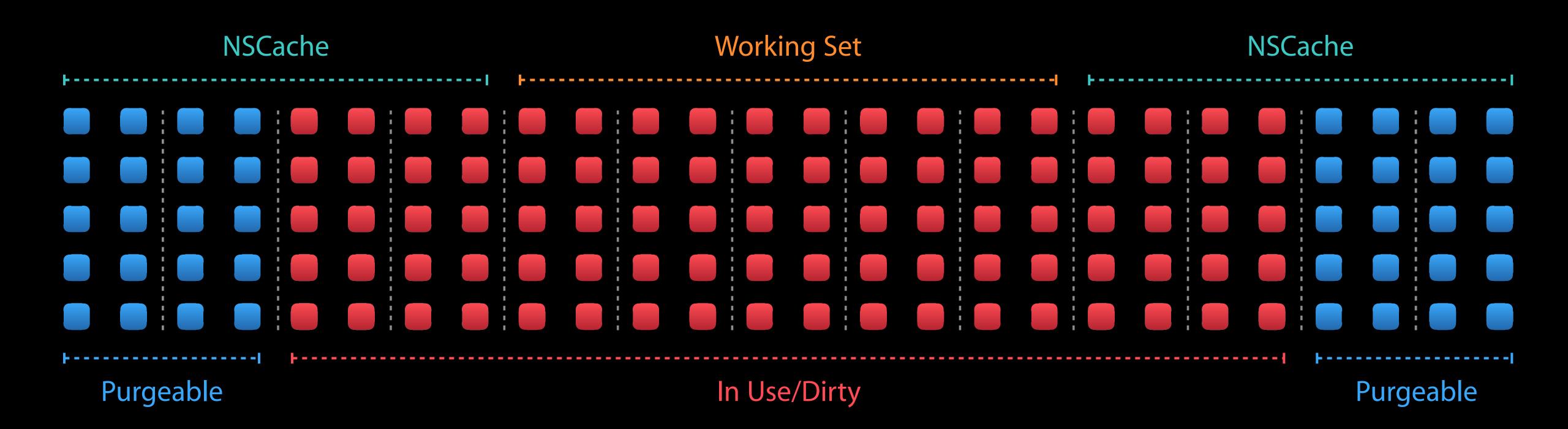


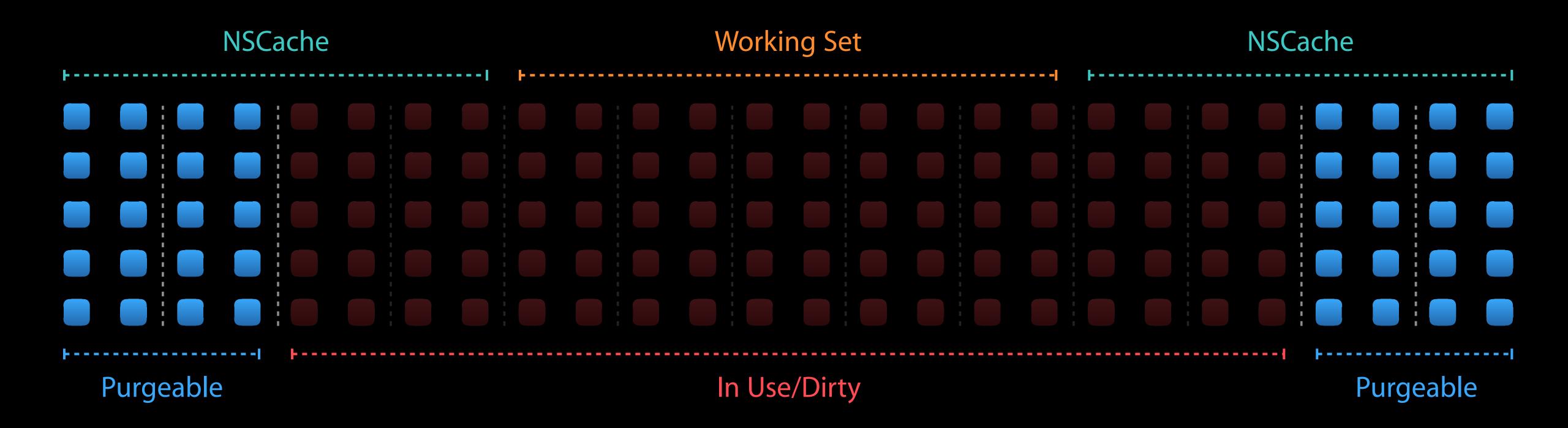








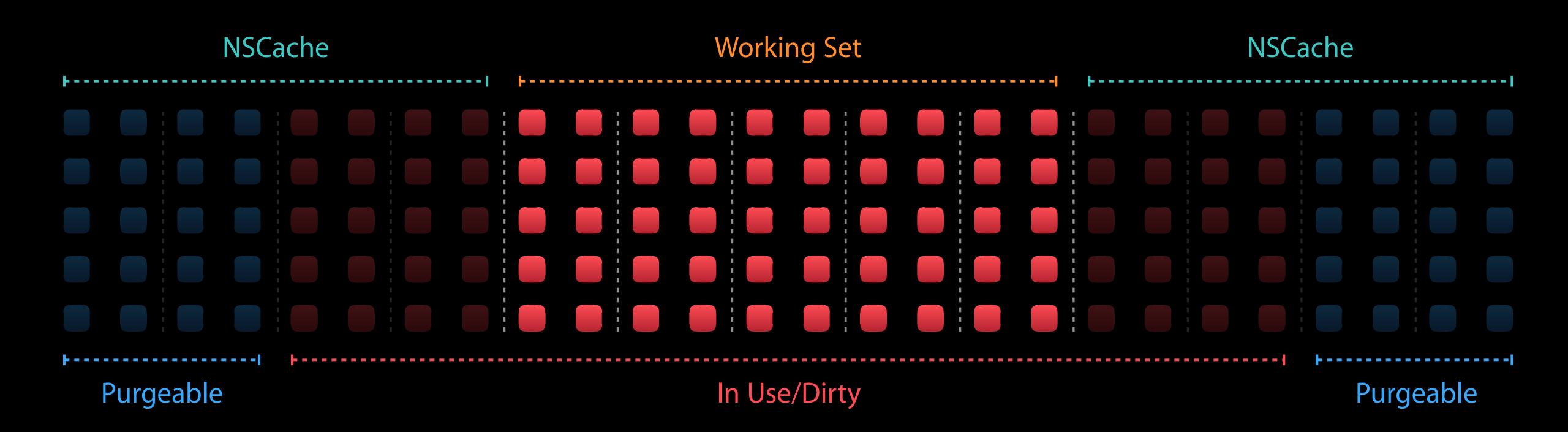


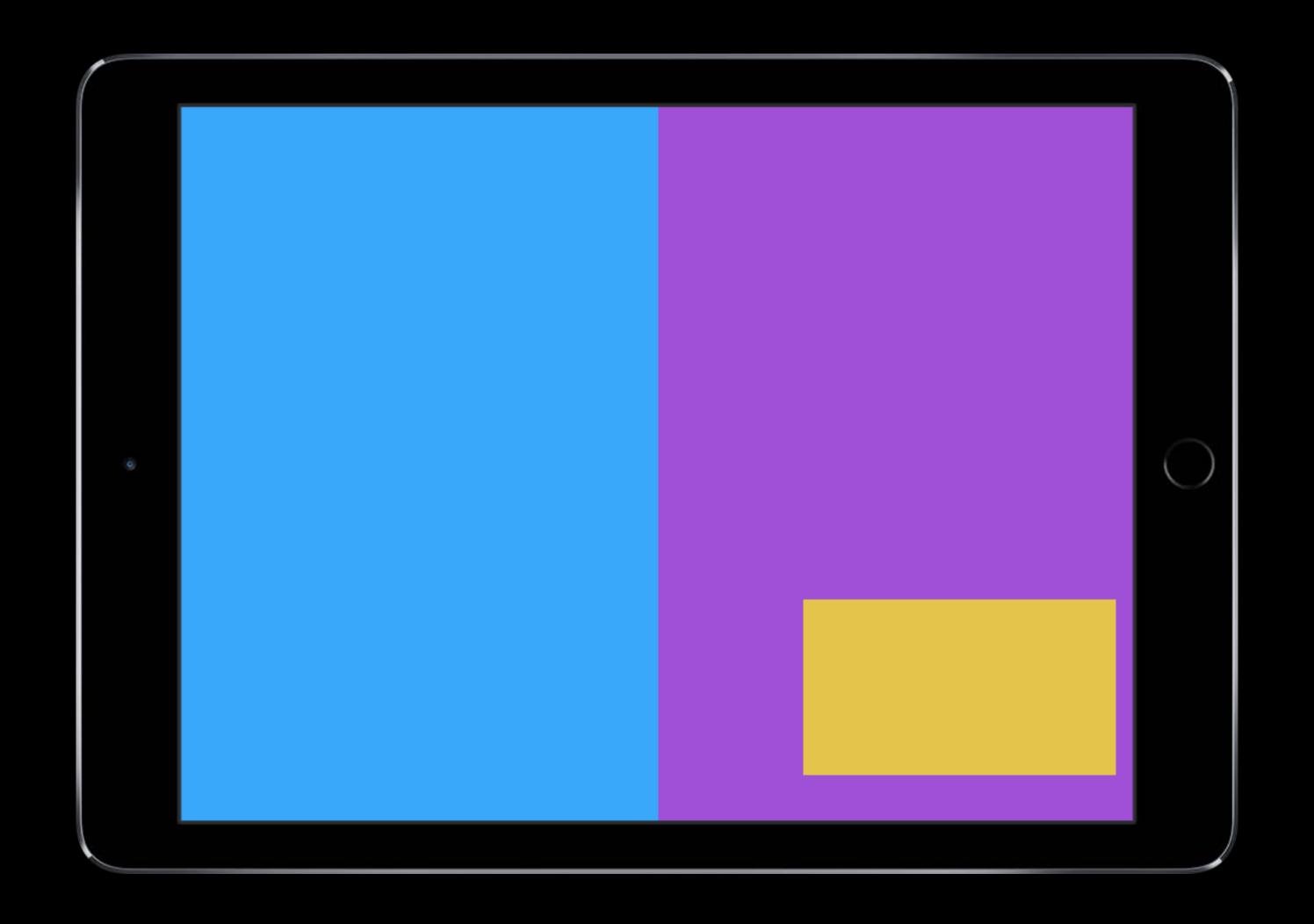


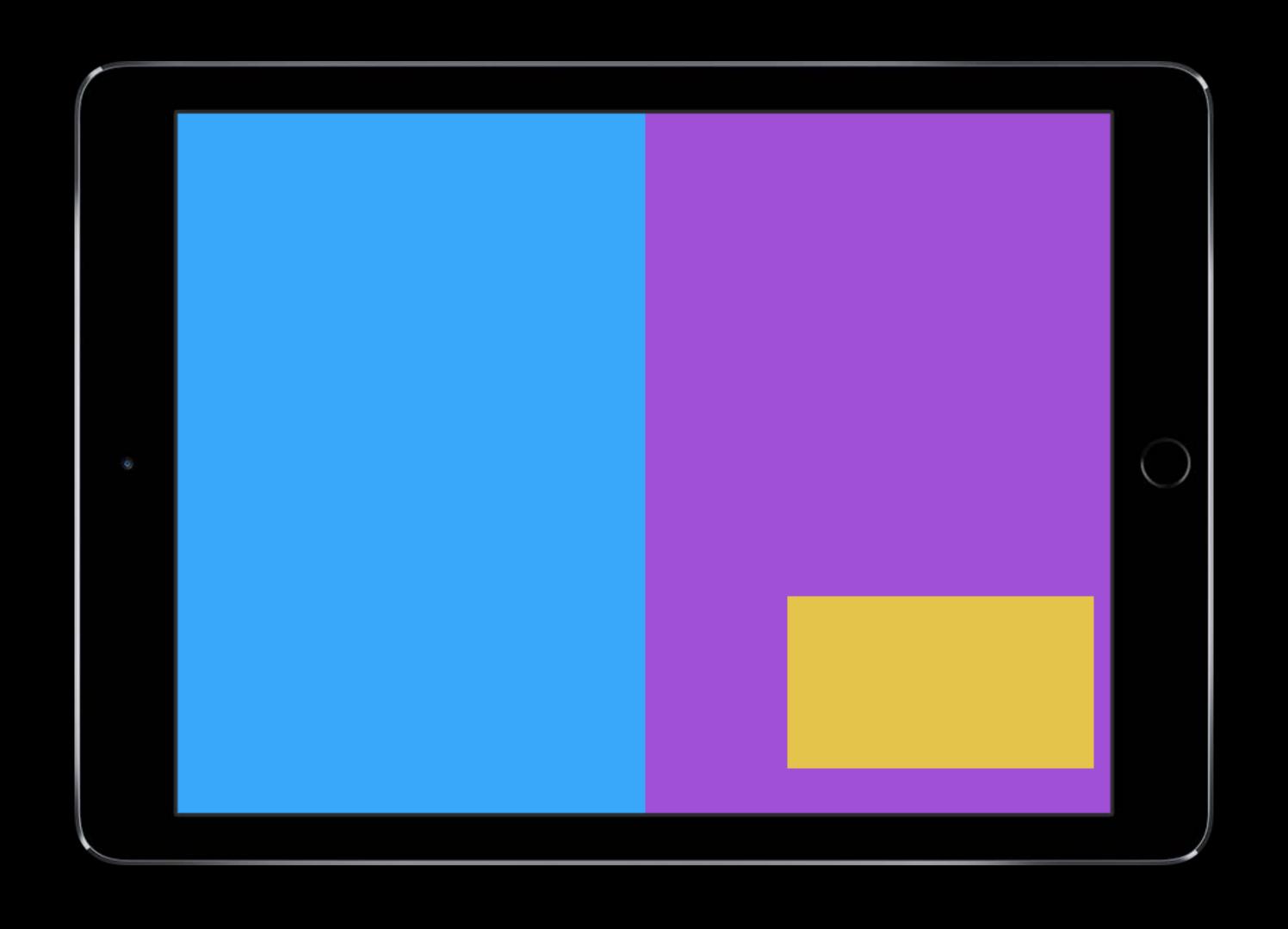


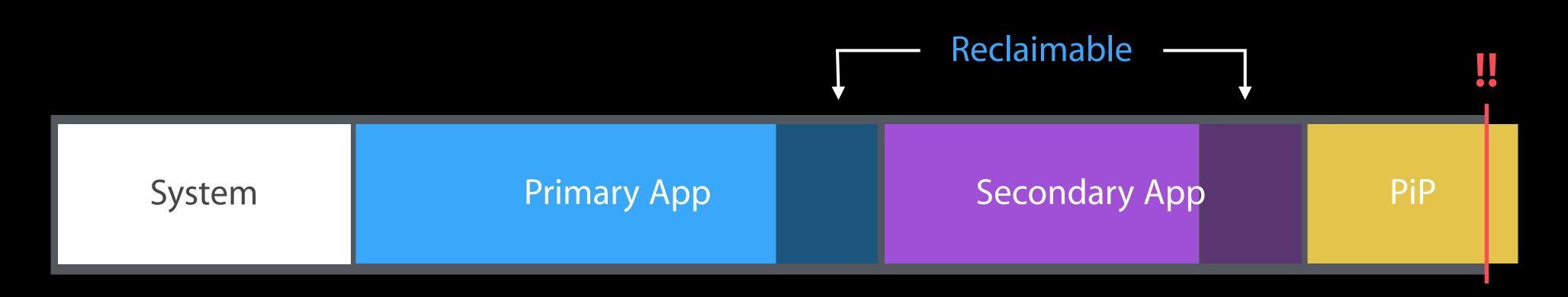
IconReel

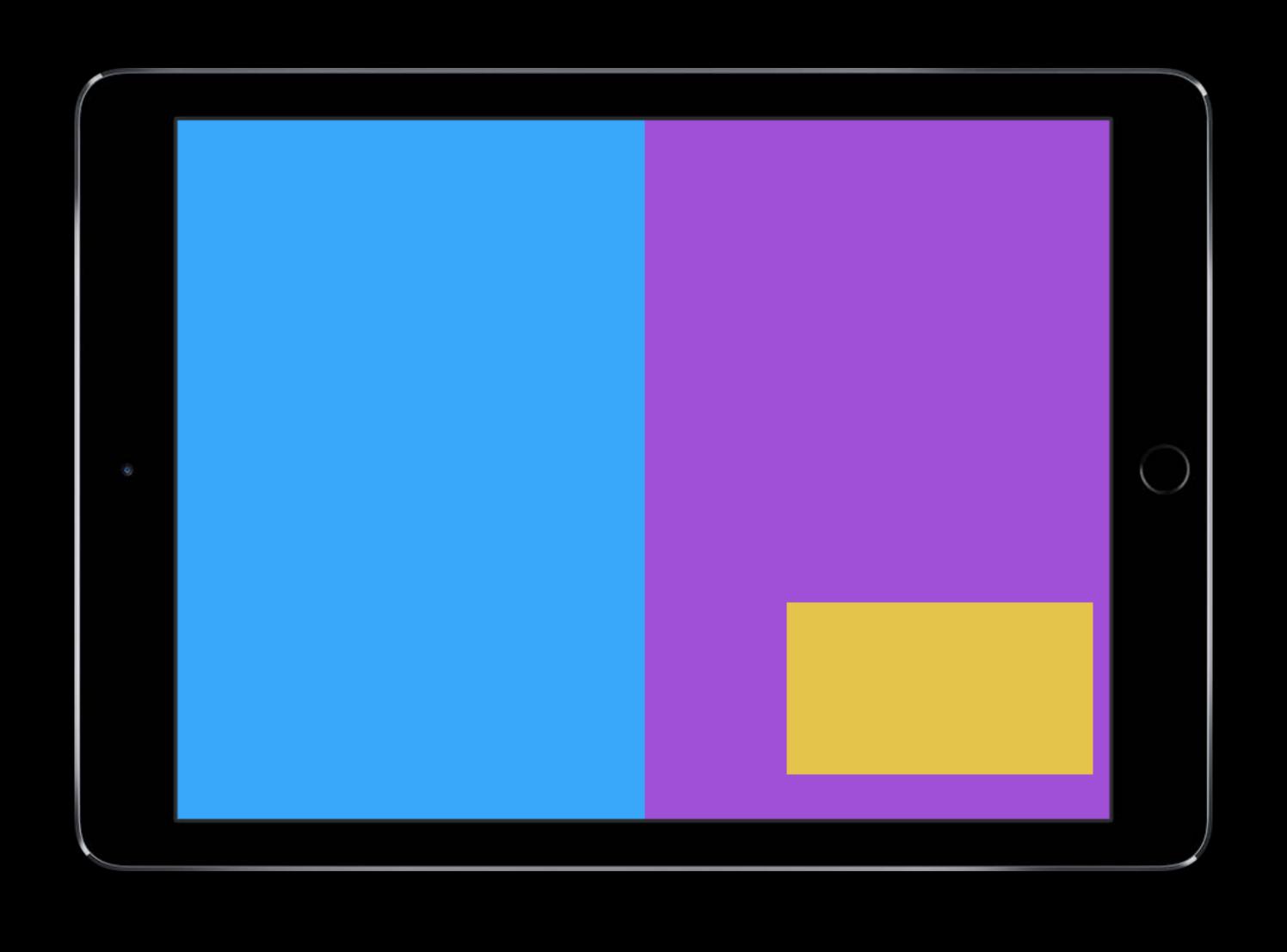
Adopting purgeable data









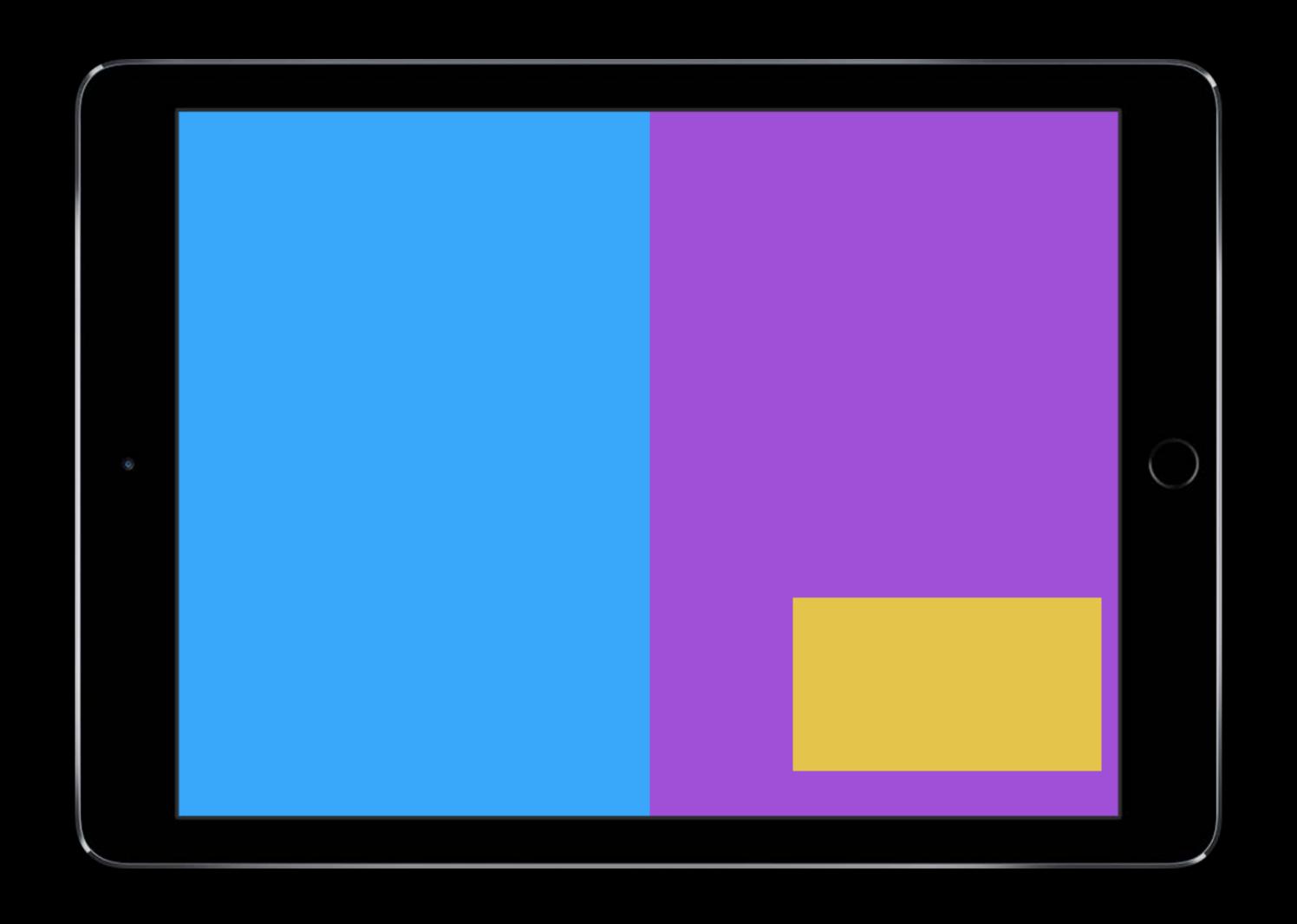


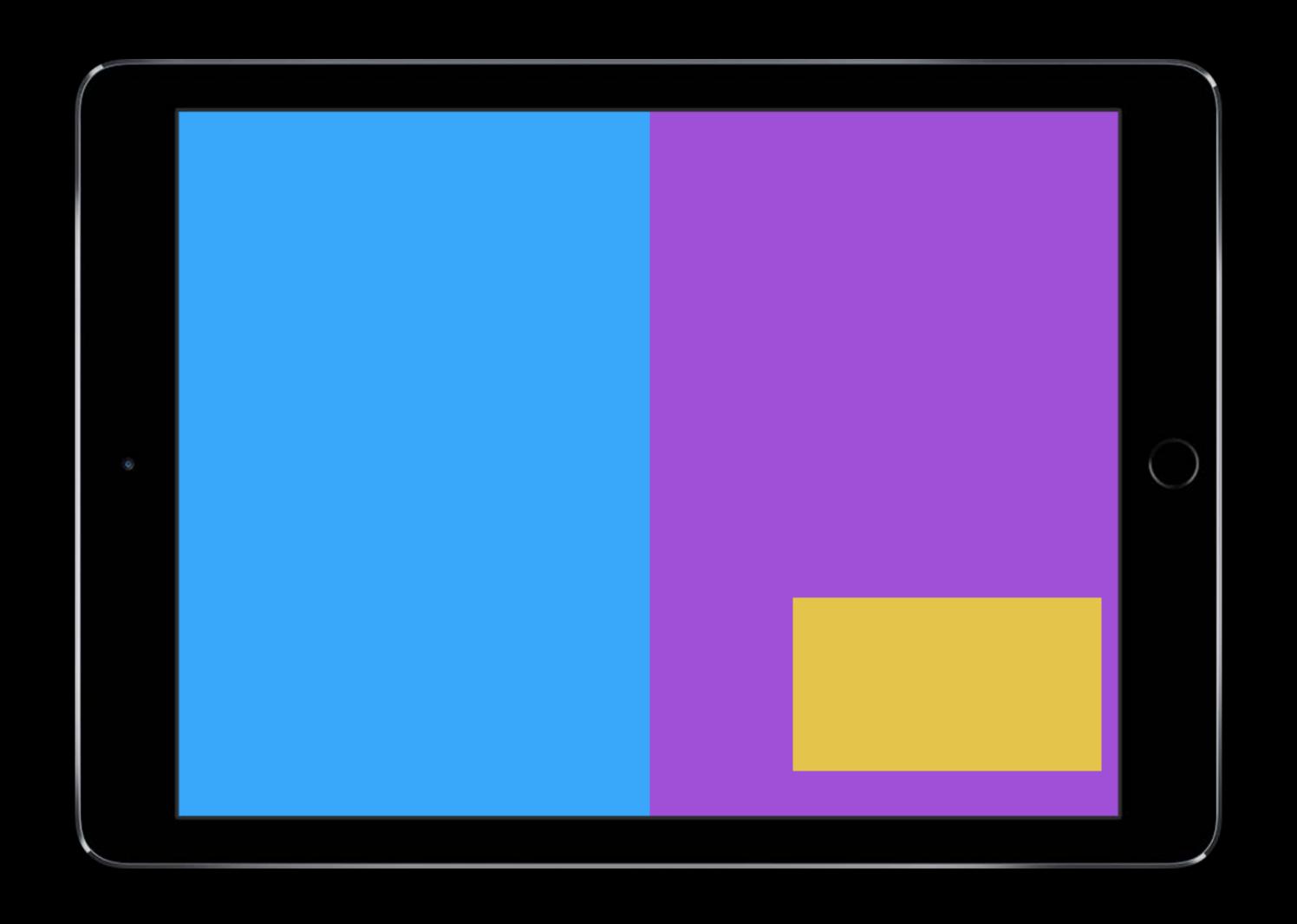


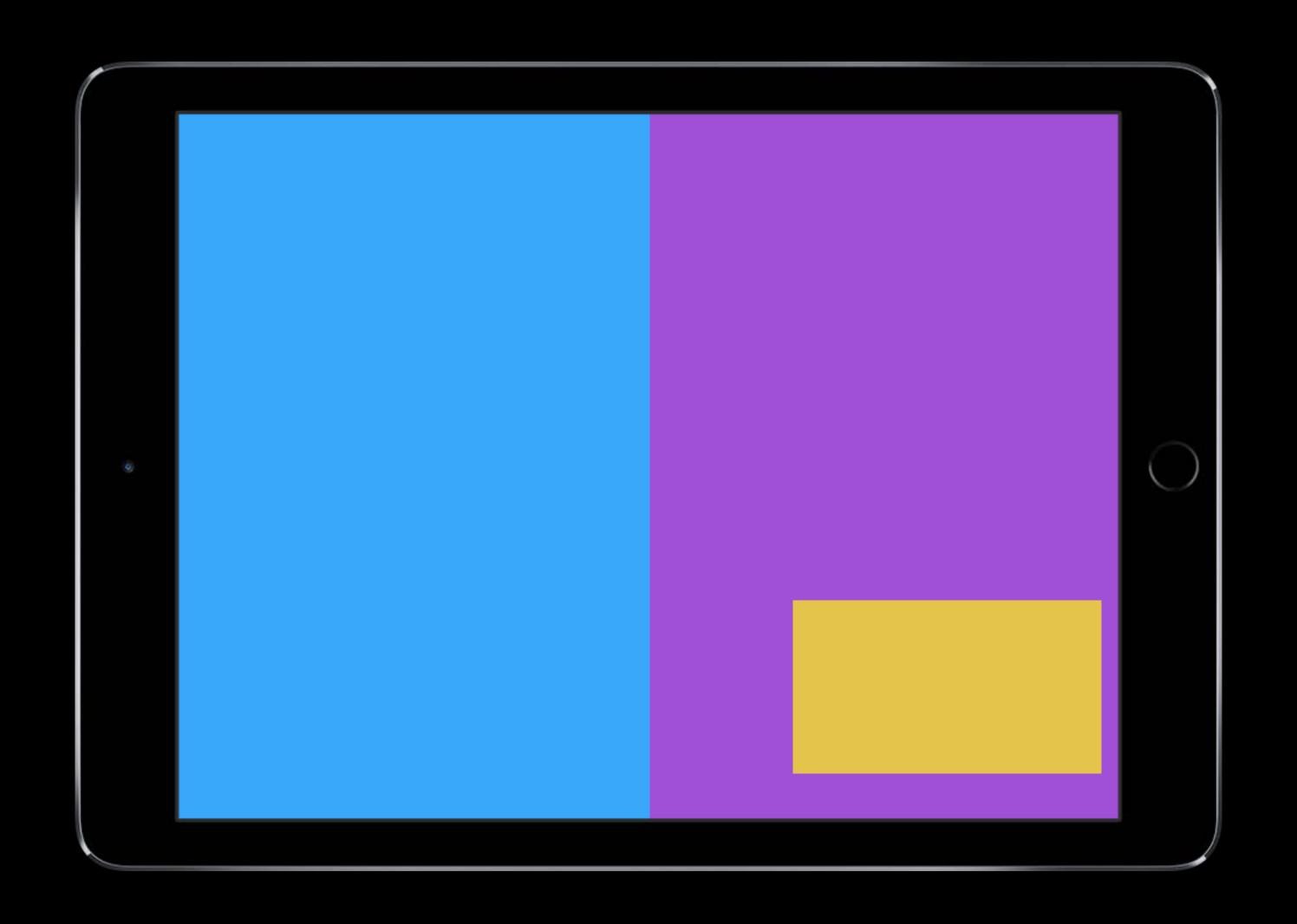
Secondary App PiP

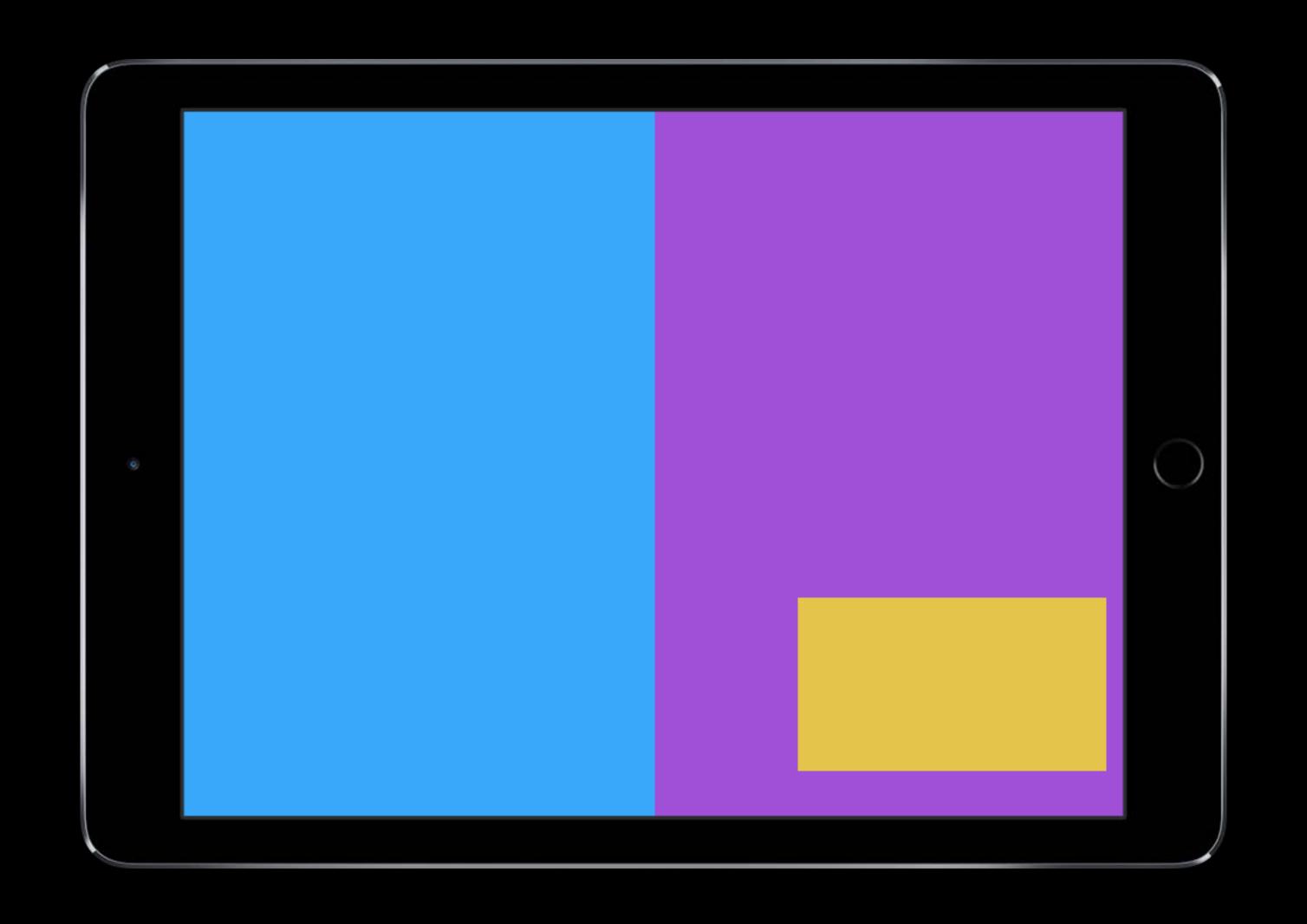
System

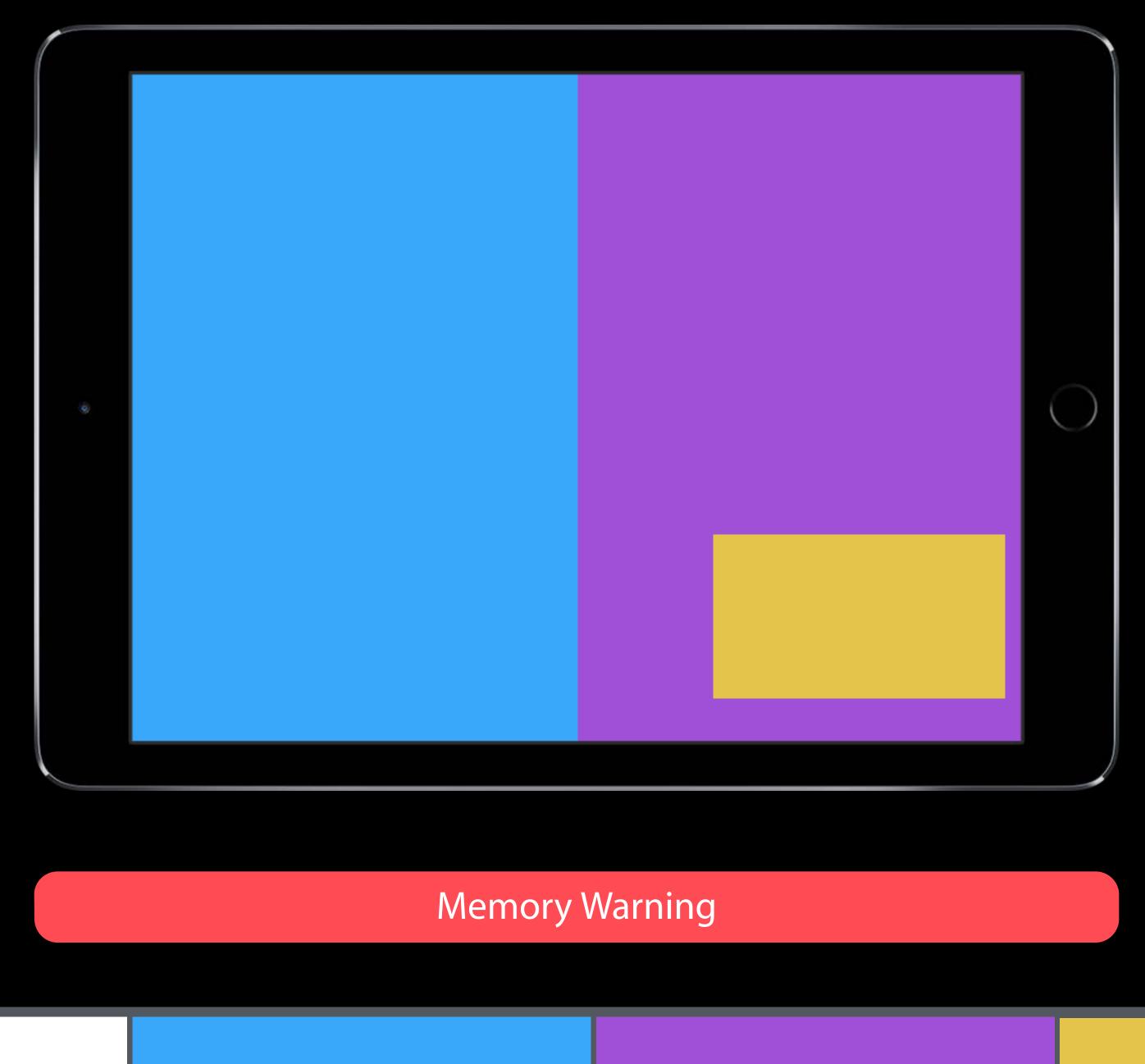
Primary App

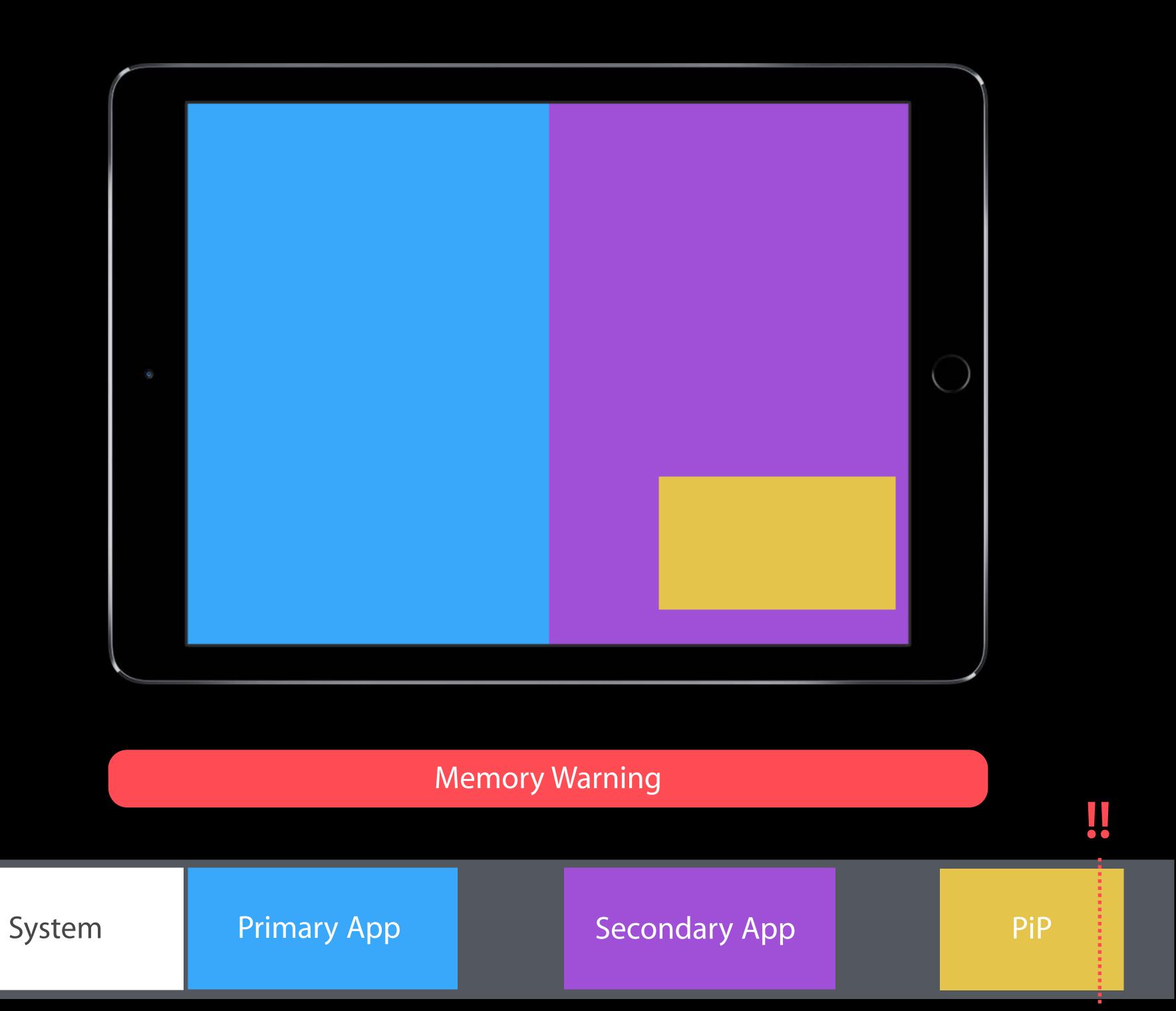


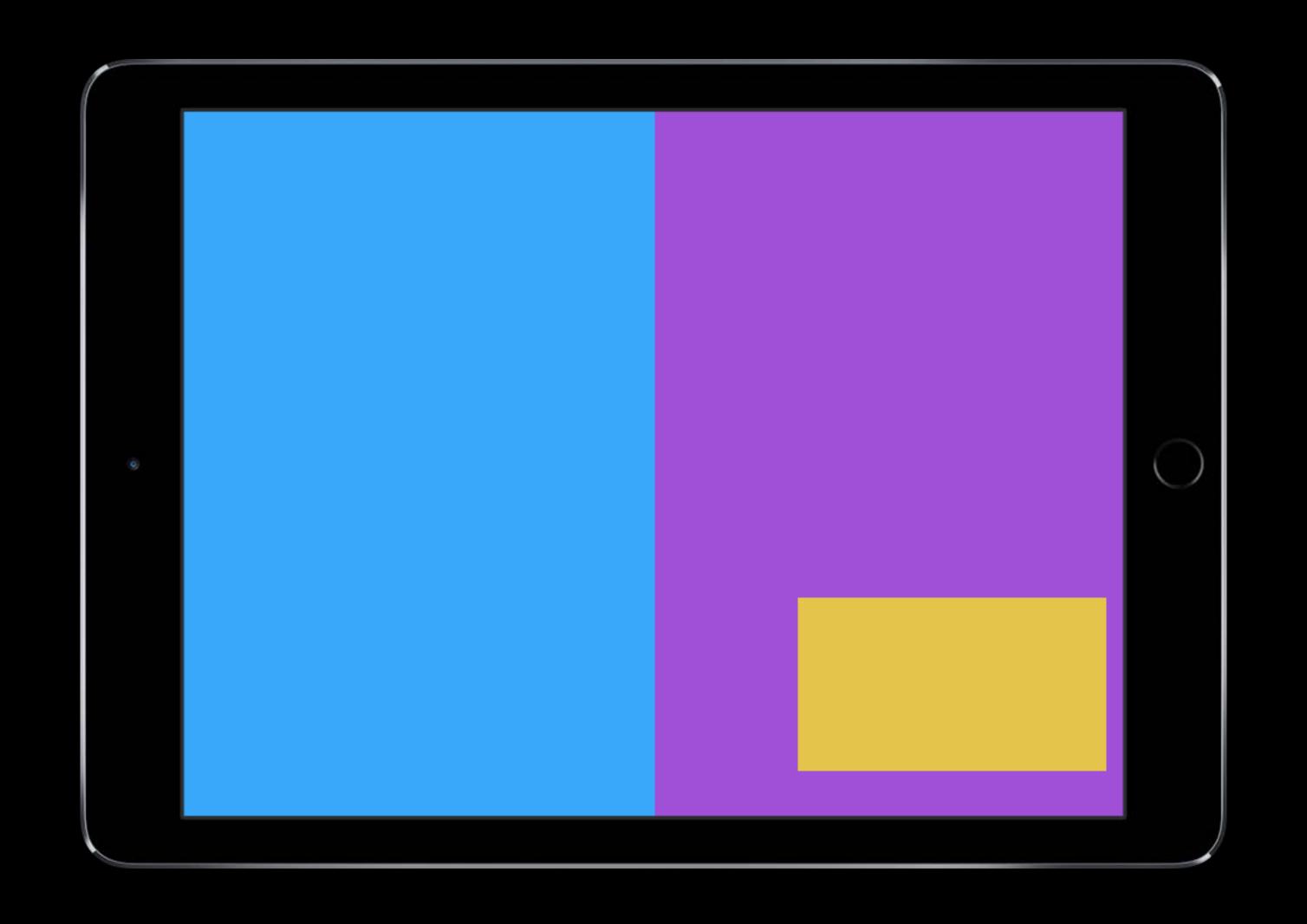


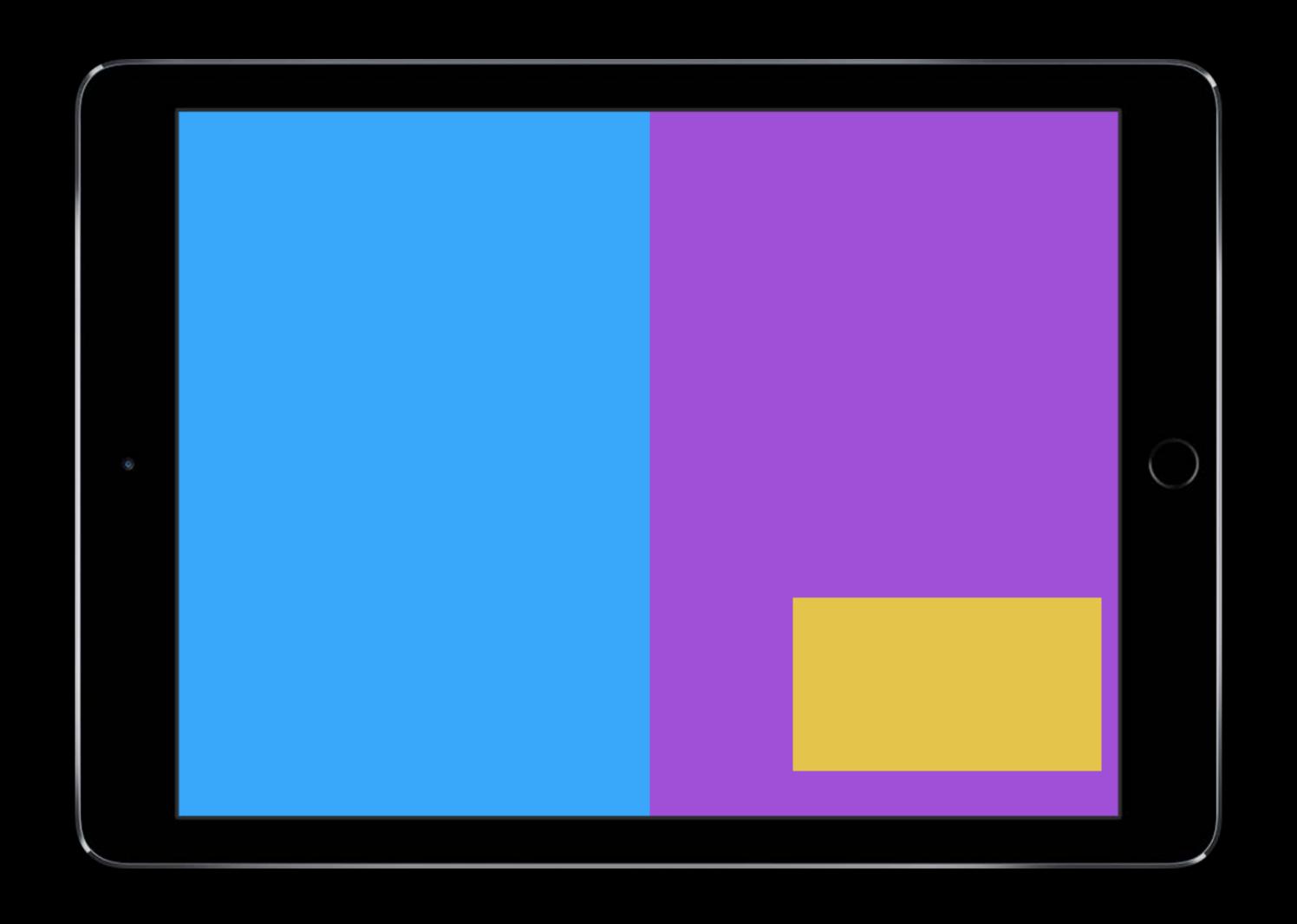








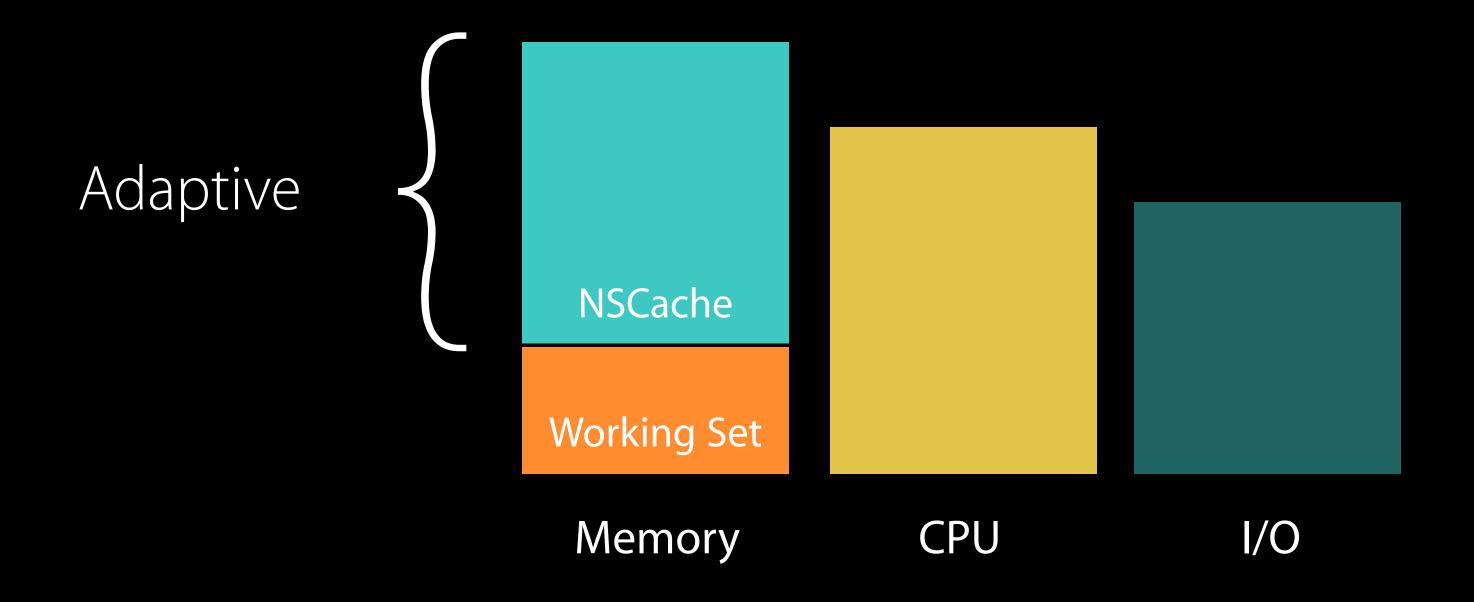


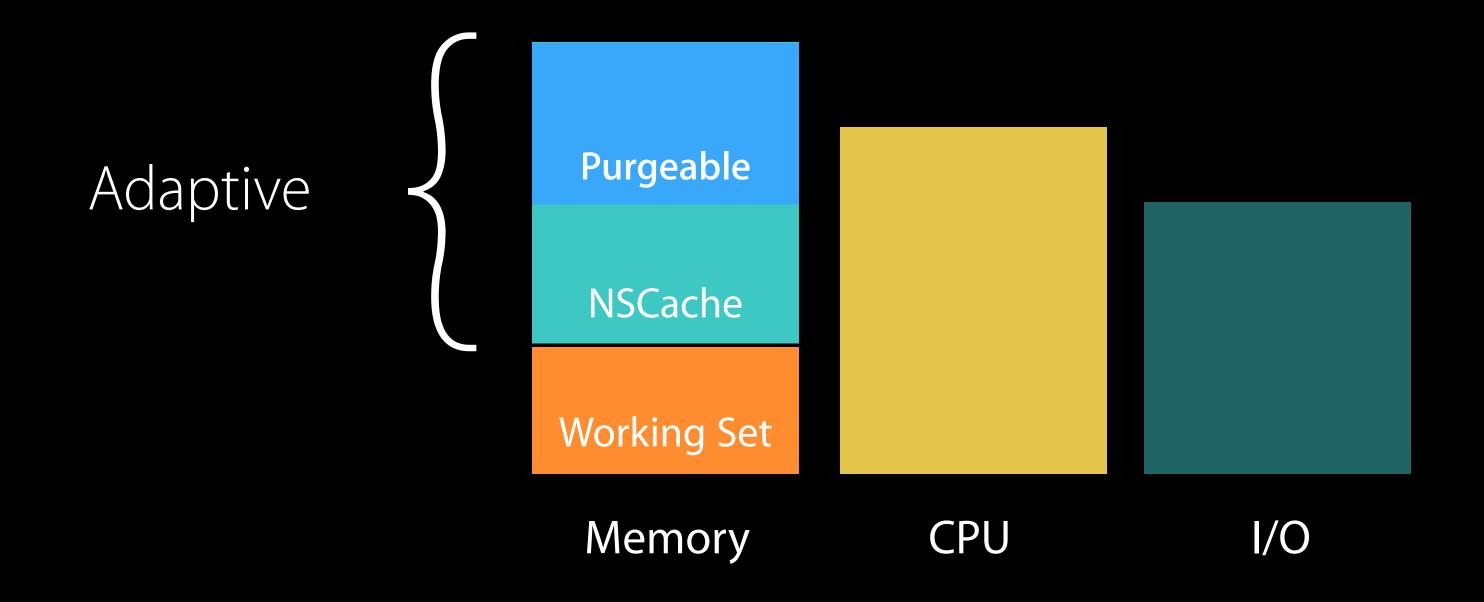


Purgeable Data API

NSPurgeableData

- beginContentAccess
- endContentAccess
- isContentDiscarded
- memory now considered dirty
- memory now considered purgeable
- has the memory been reclaimed?





IconReel

Data characteristics

Absolutely essential

- Absolutely essential
- Expensive to generate

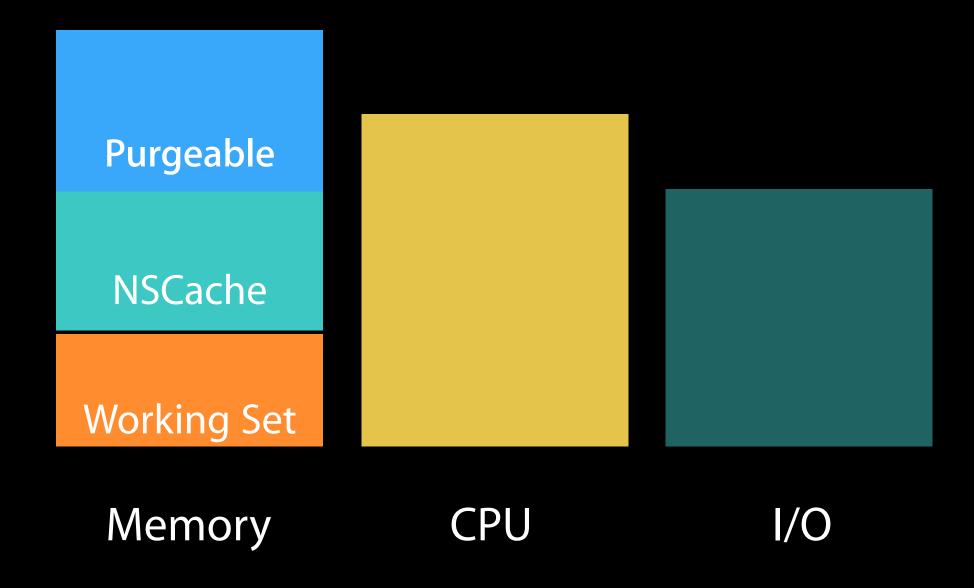
- Absolutely essential
- Expensive to generate
- Can be pre-computed

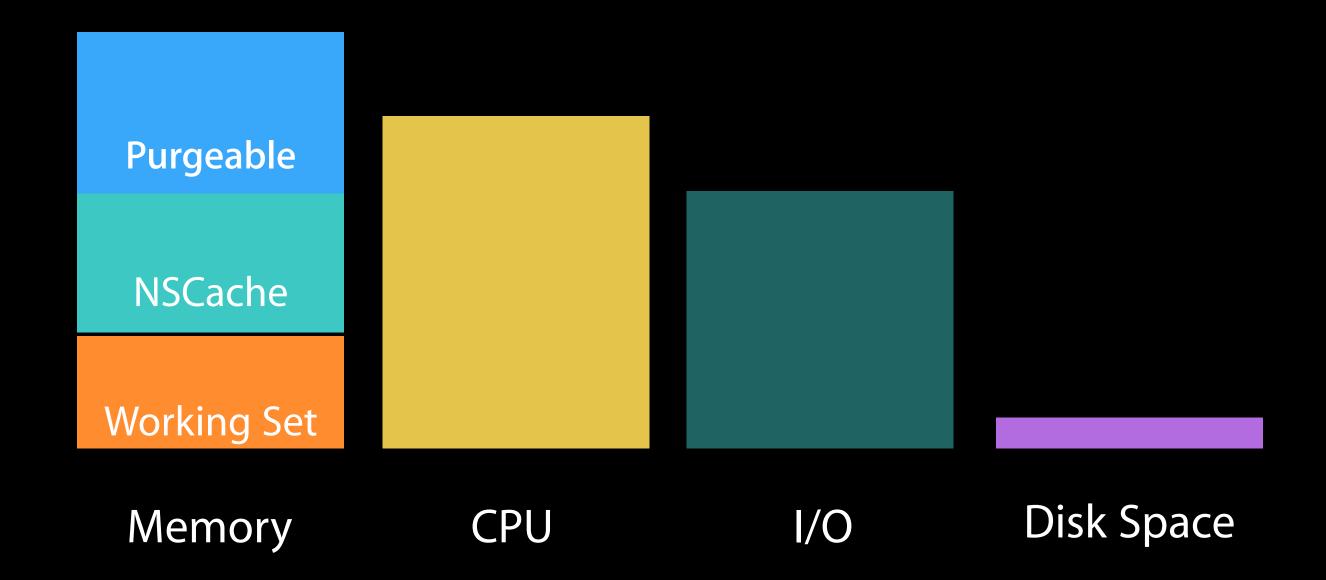
- Absolutely essential
- Expensive to generate
- Can be pre-computed
- Static once generated

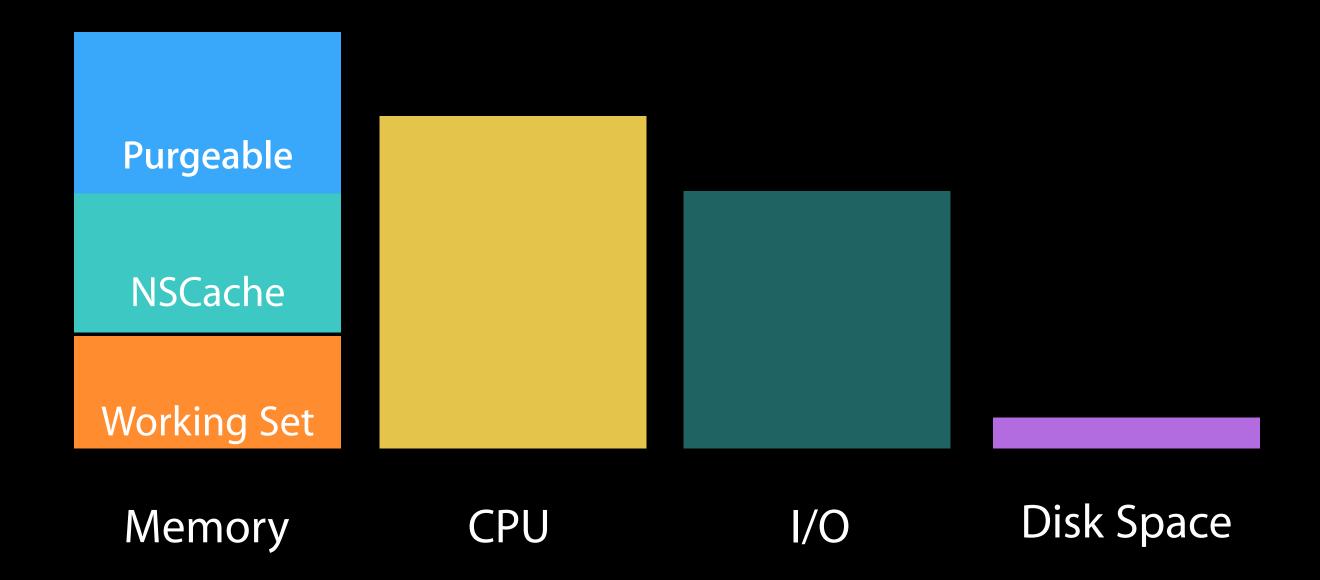
IconReel

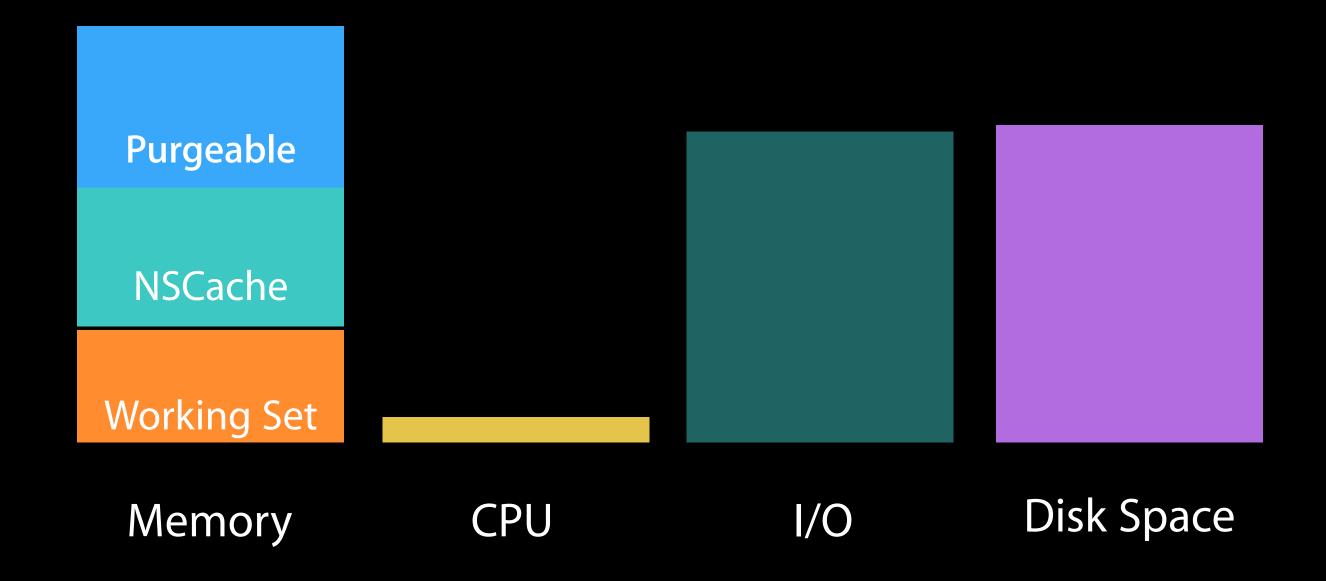
Data characteristics

- Absolutely essential
- Expensive to generate
- Can be pre-computed
- Static once generated
- Perfect candidate for caching to a file!









Memory backed by a file on disk is considered clean

• Data in such a file can be "memory mapped"

Memory backed by a file on disk is considered clean

- Data in such a file can be "memory mapped"
- Memory and file contents must match exactly
- Ideal for read-only data

Memory backed by a file on disk is considered clean

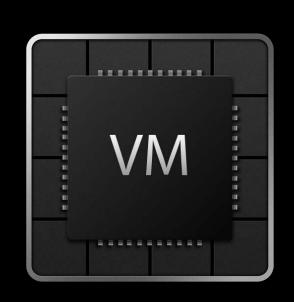
• Data in such a file can be "memory mapped"

Memory and file contents must match exactly

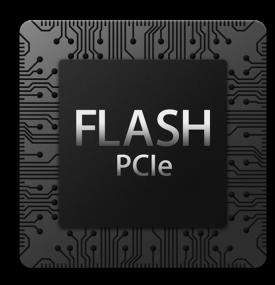
Ideal for read-only data

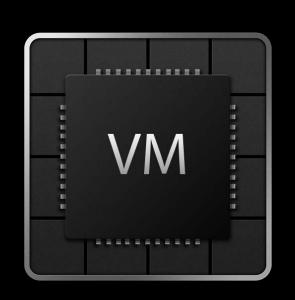
Data in memory is evicted and reloaded on your behalf

Allows for random access

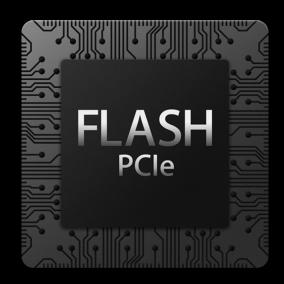


Data in Memory

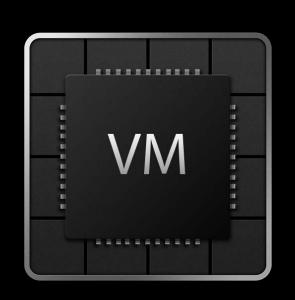




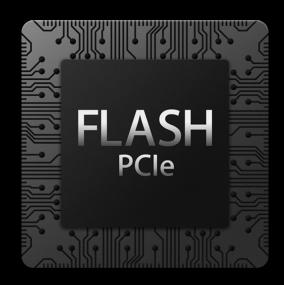
Data in Memory



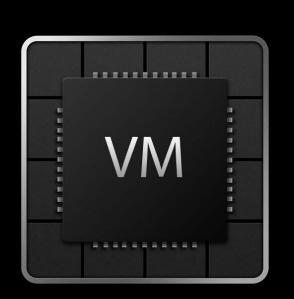
File on Disk

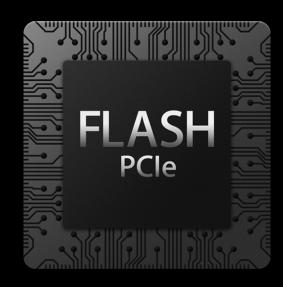


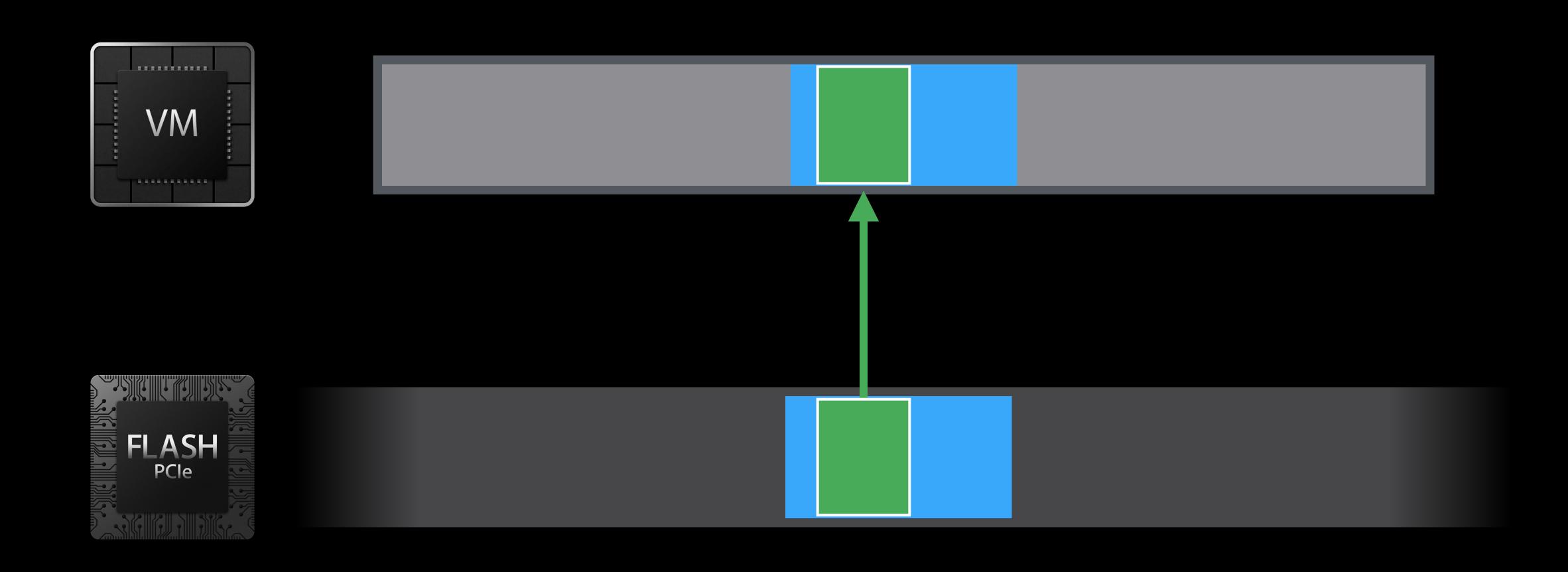
Memory Mapped Data

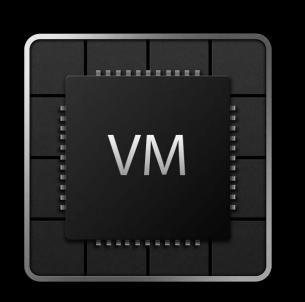


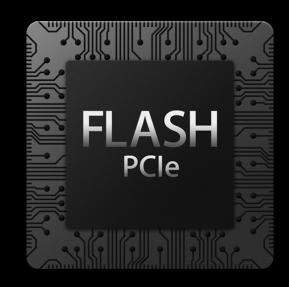
File on Disk



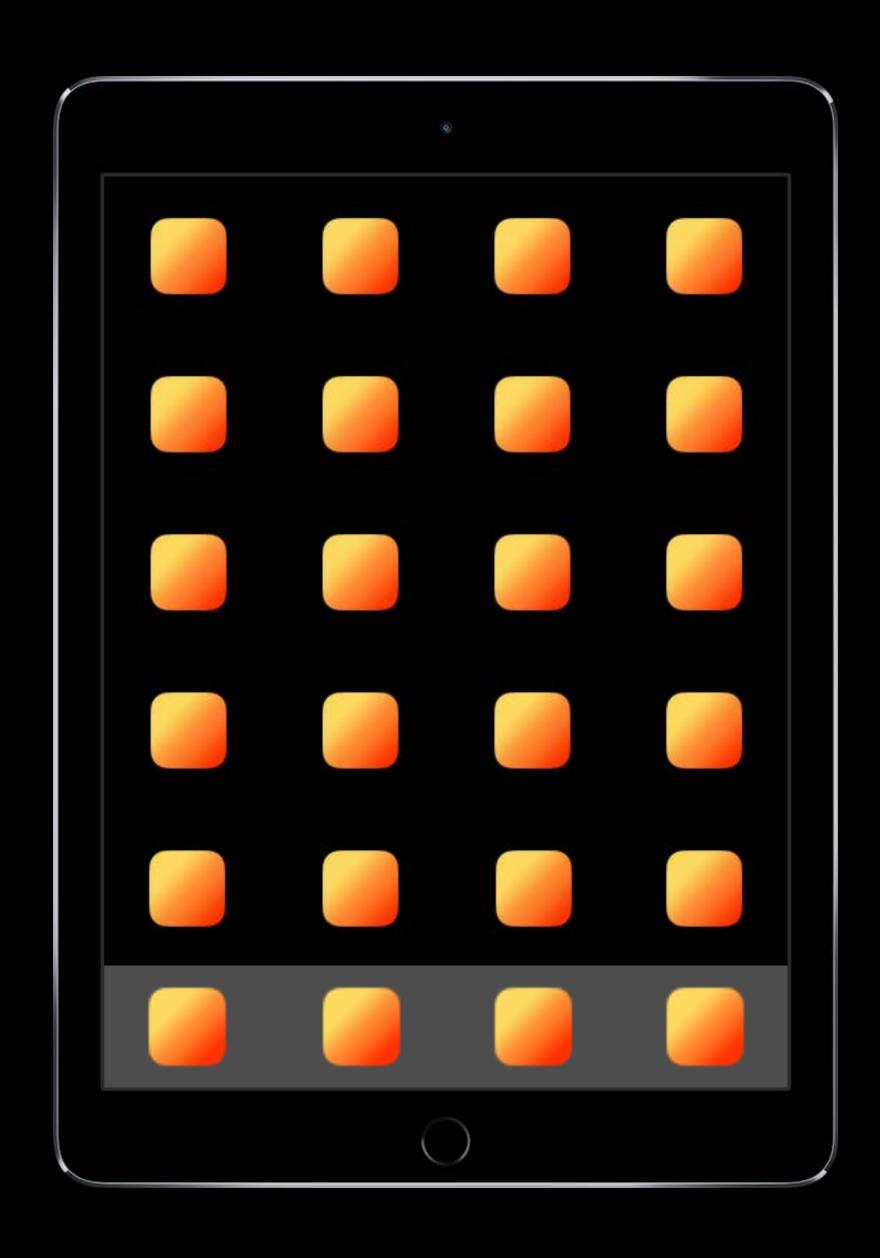


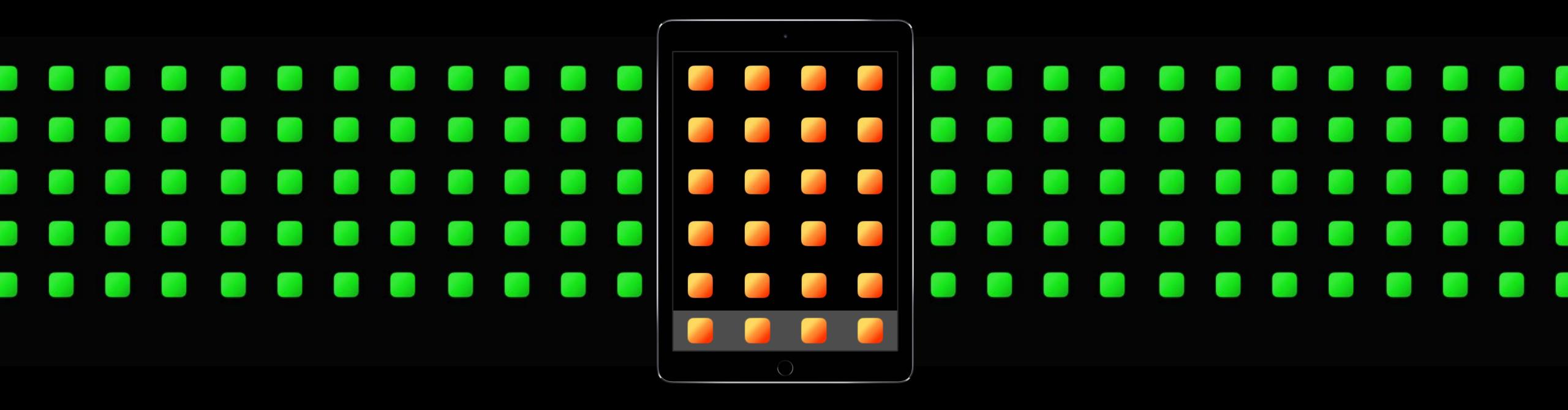


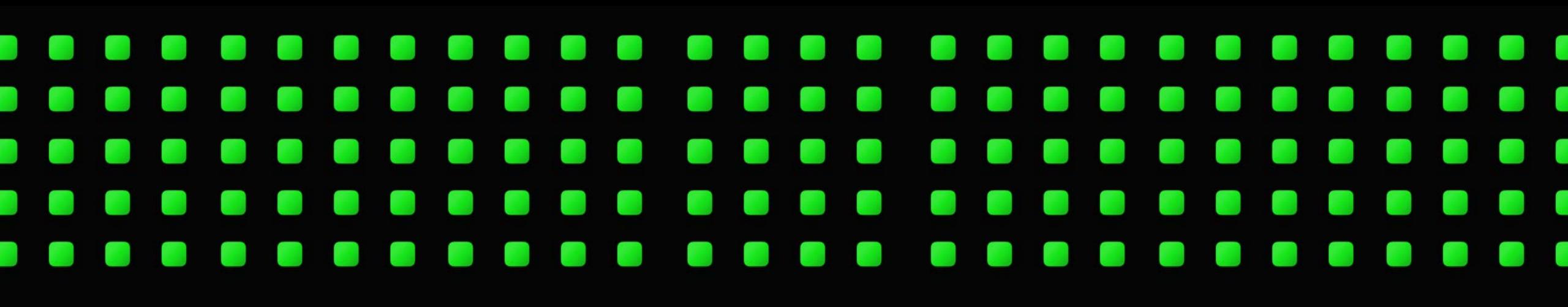


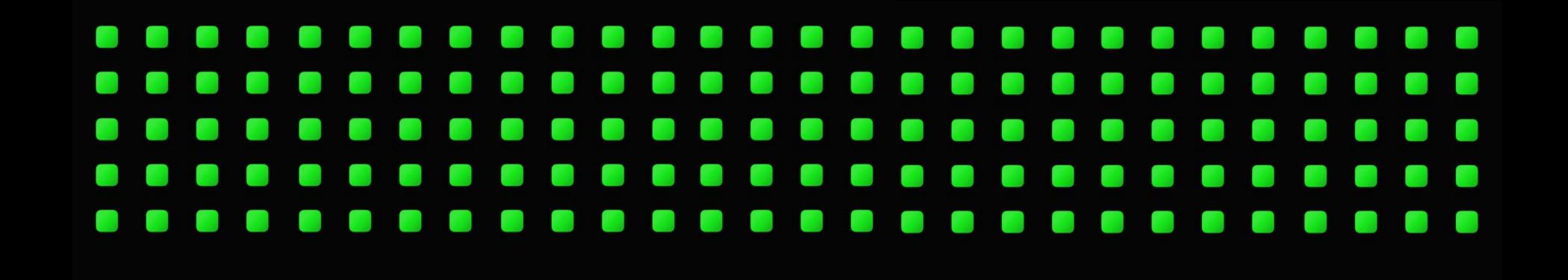


IconReel

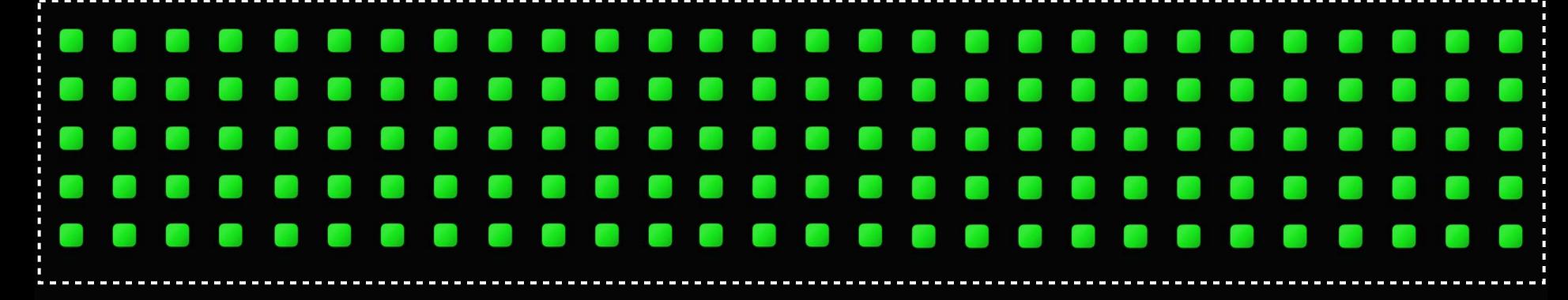


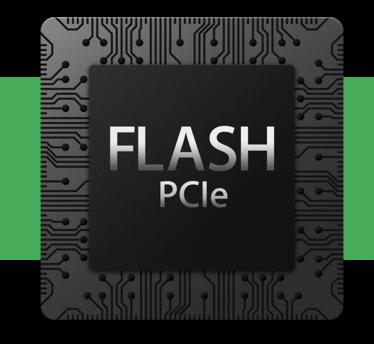


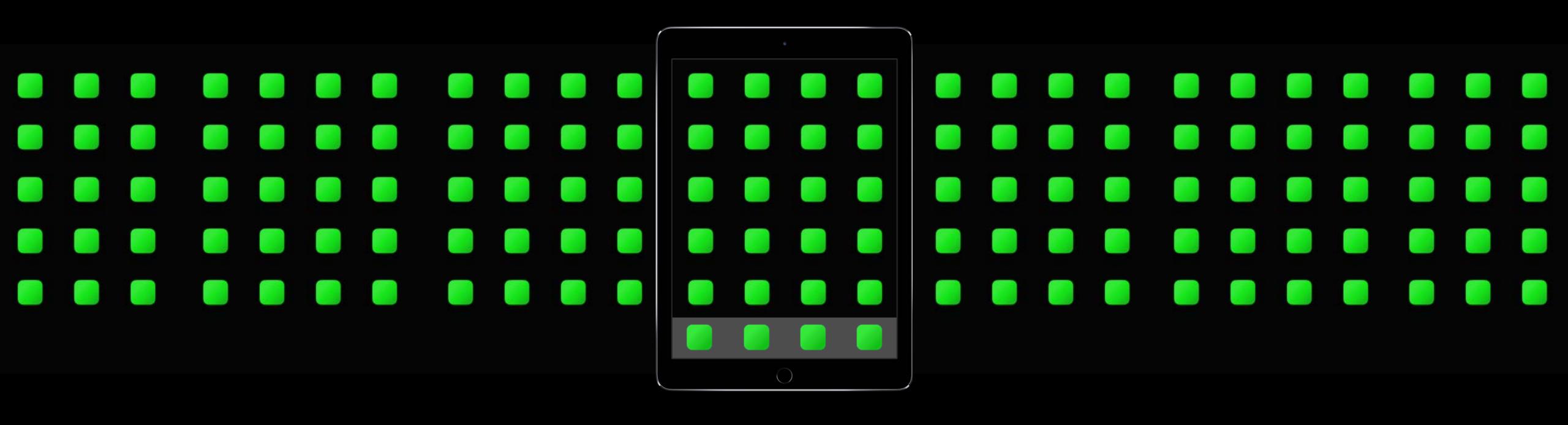


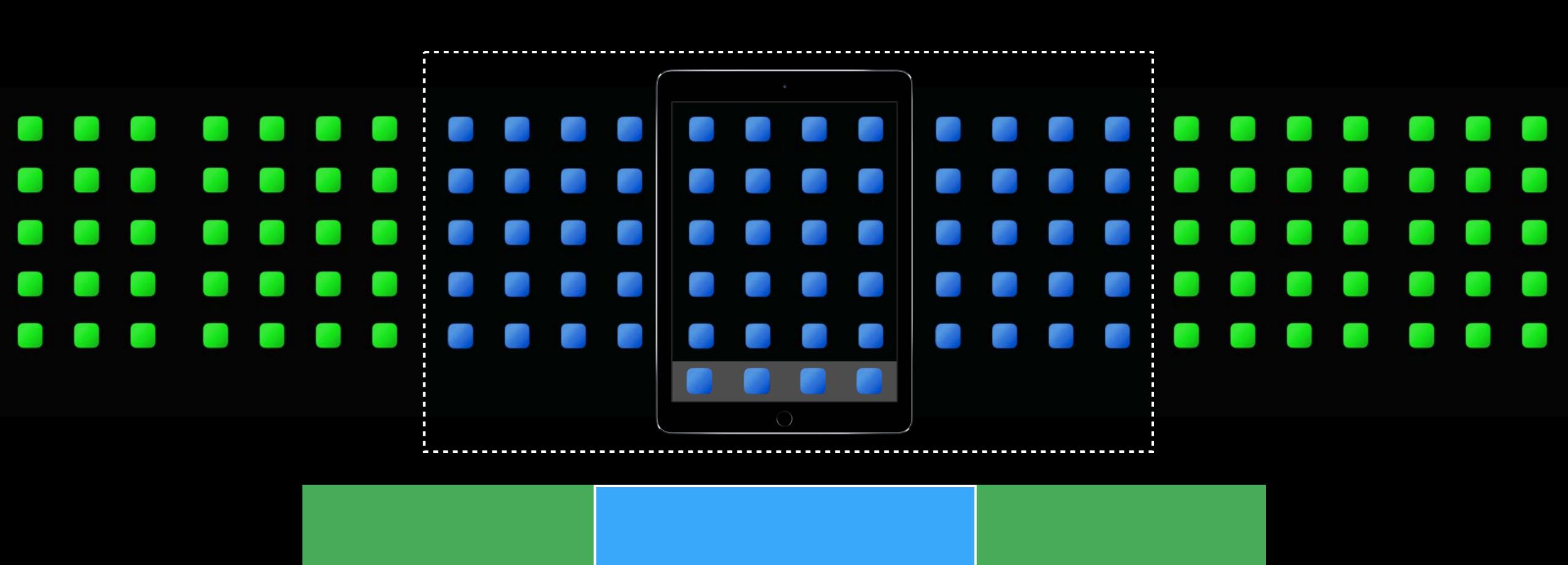


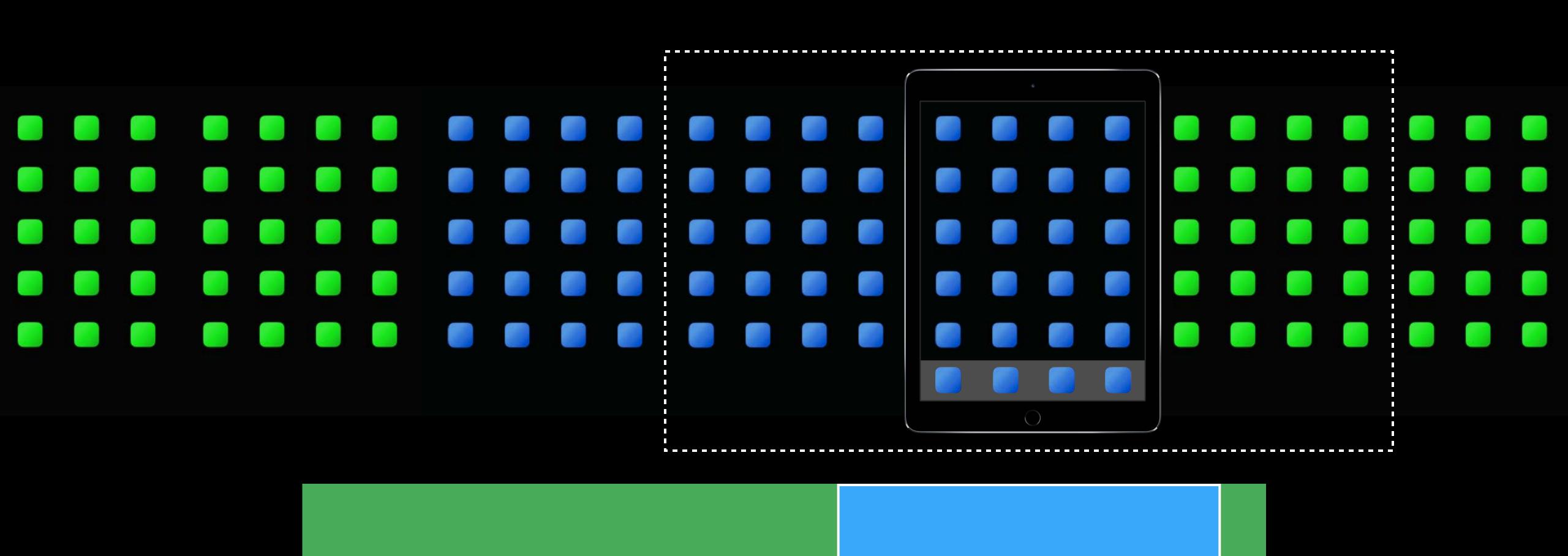
Memory Mapped Image Data

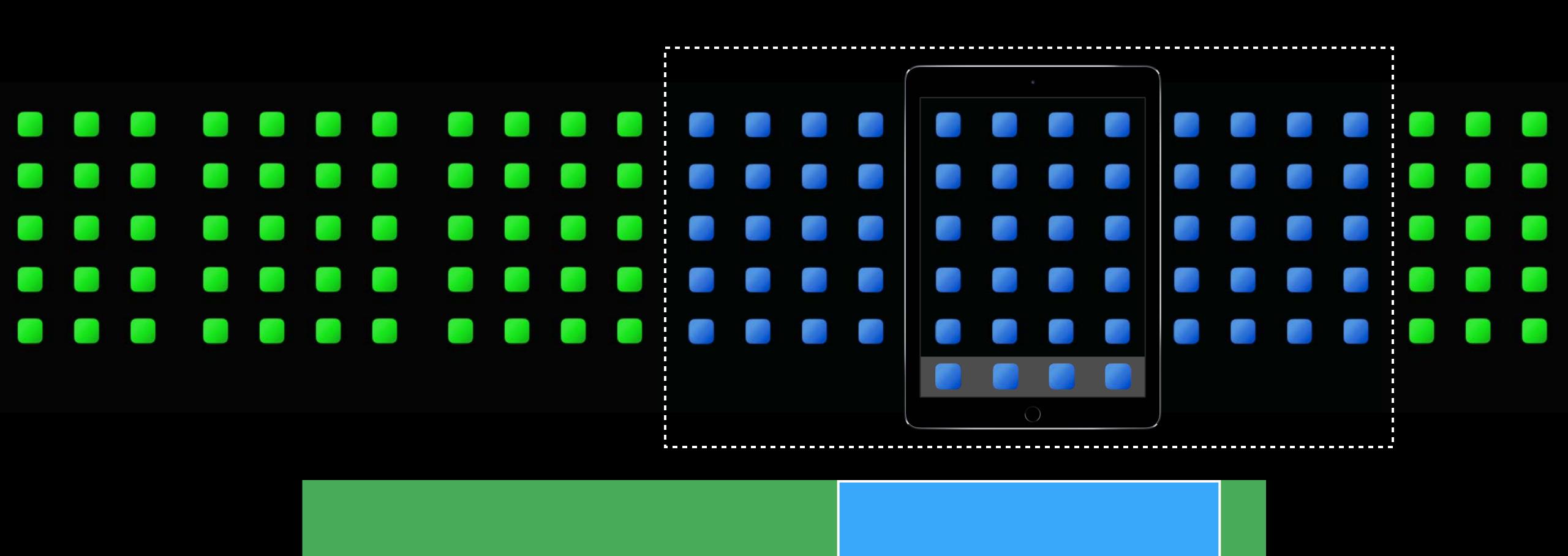


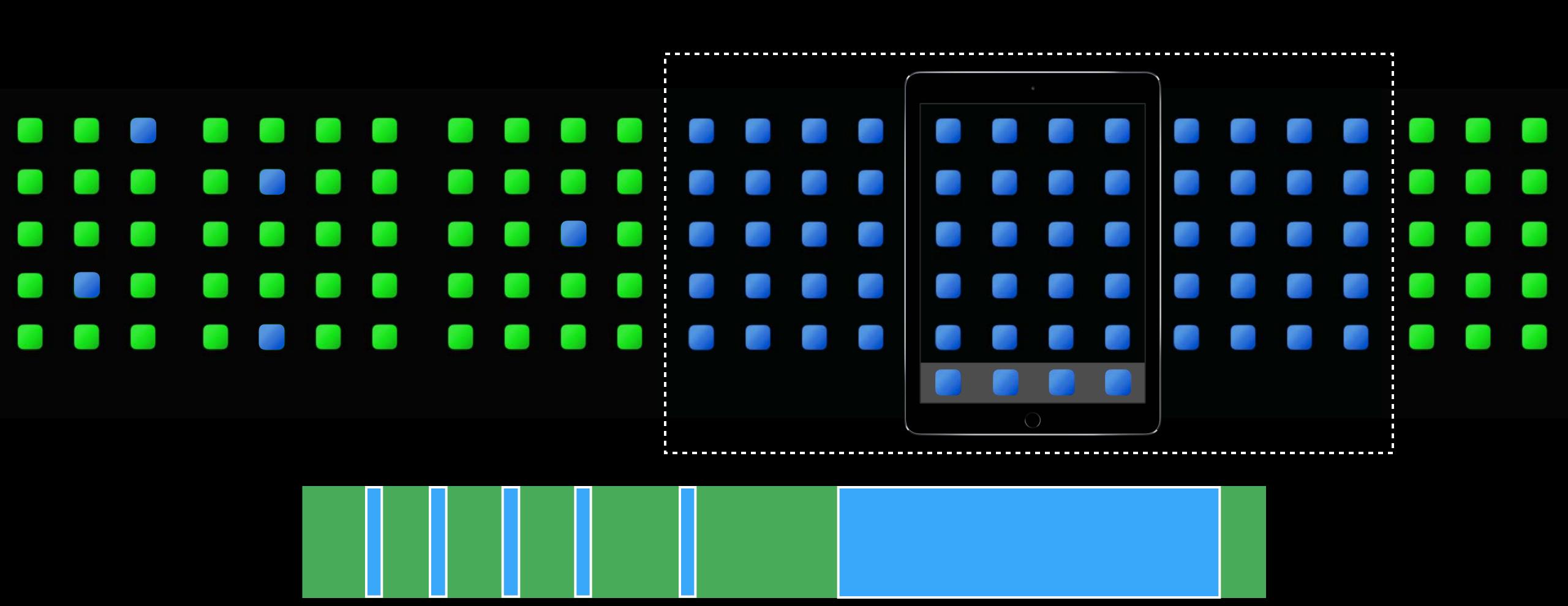




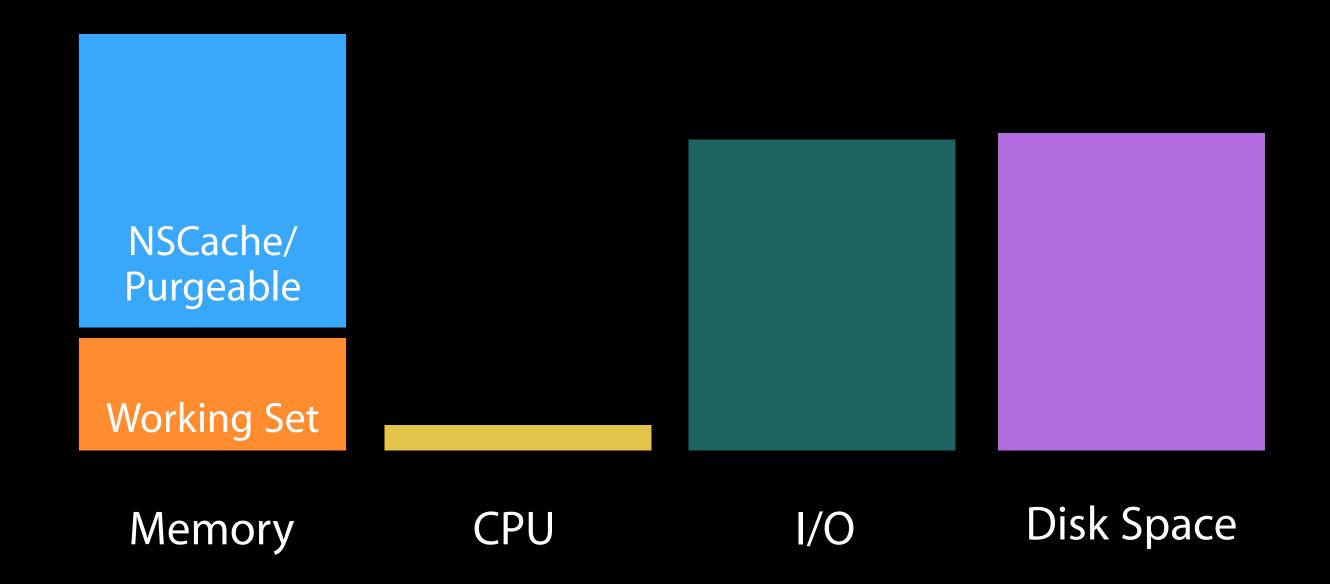




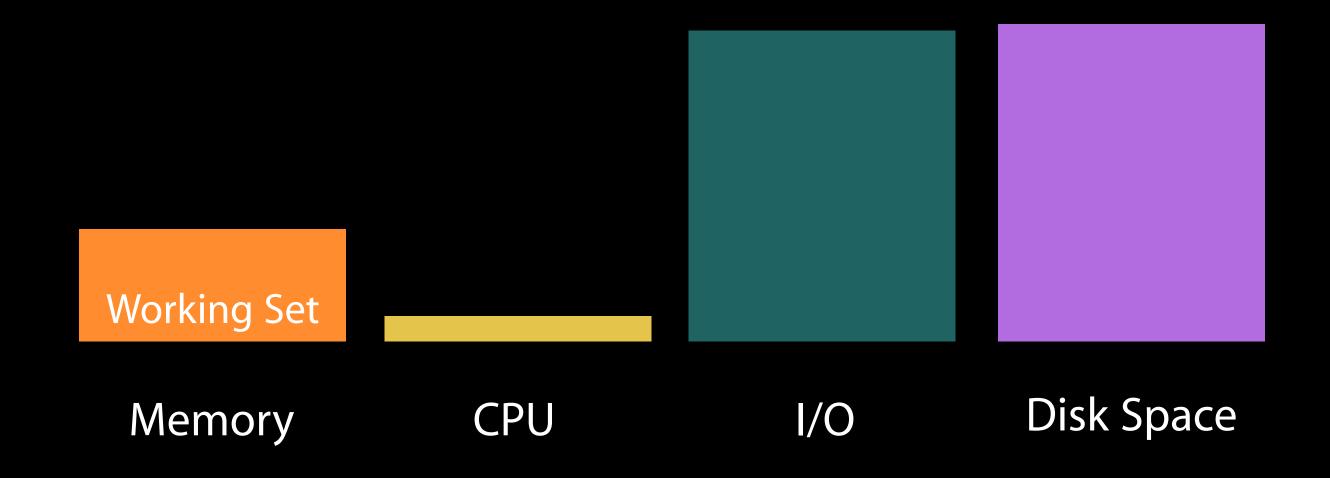




IconReel System resources



IconReel System resources



Memory Mapped Data API

@end

Memory Mapped Data API

Memory Mapped Data

API

Caveats



Caveats

Not appropriate for small chunks of data

VM Addressable Space

0xFFE00000

Caveats

Not appropriate for small chunks of data

Virtual Memory misuse

- Fragmentation
- Exhaustion



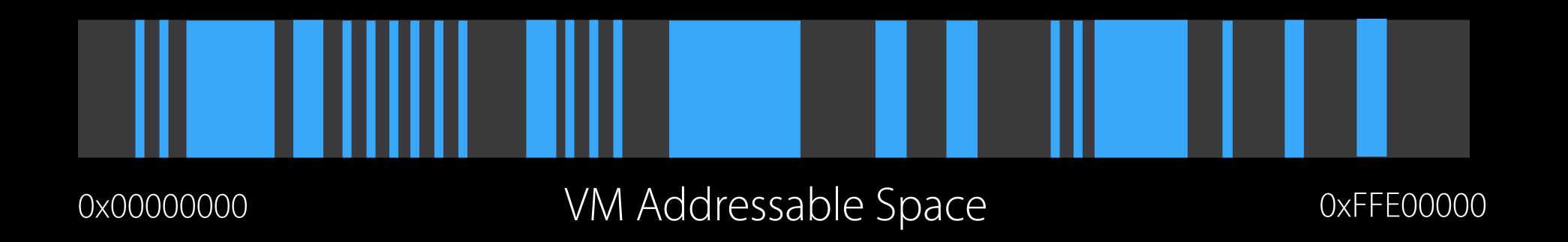
Caveats

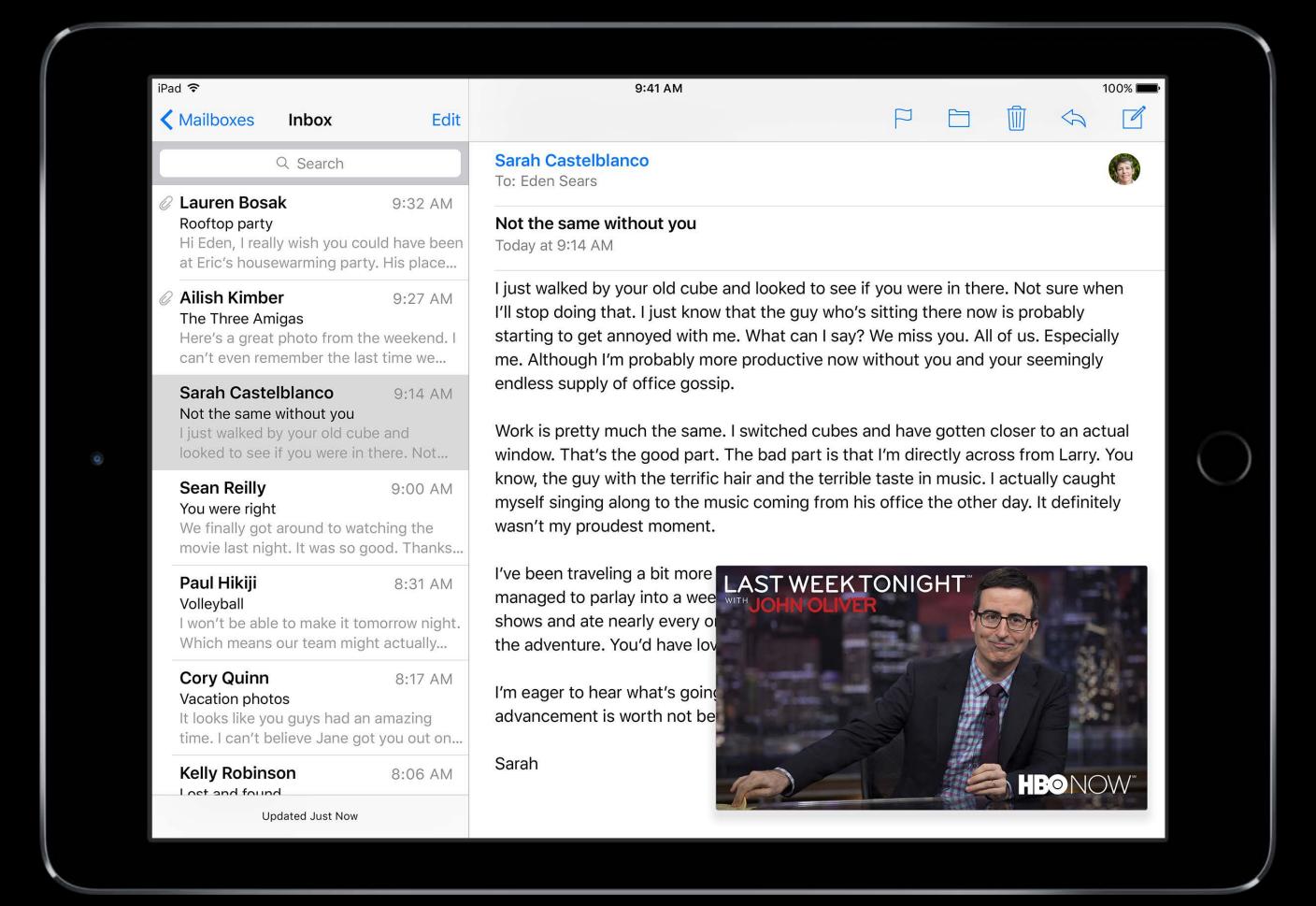
Not appropriate for small chunks of data

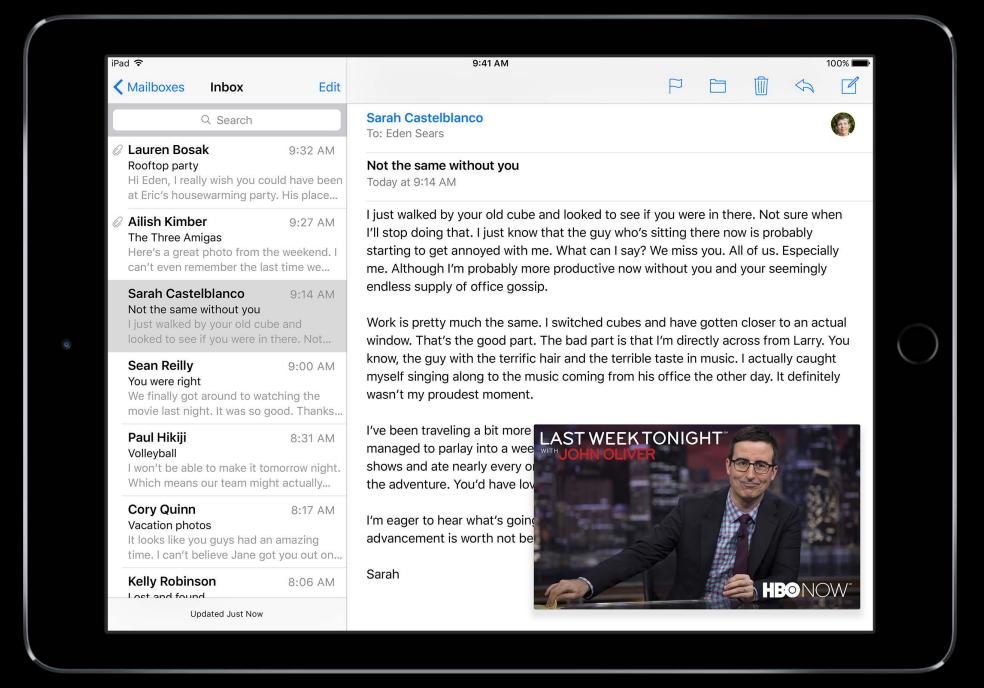
Virtual Memory misuse

- Fragmentation
- Exhaustion

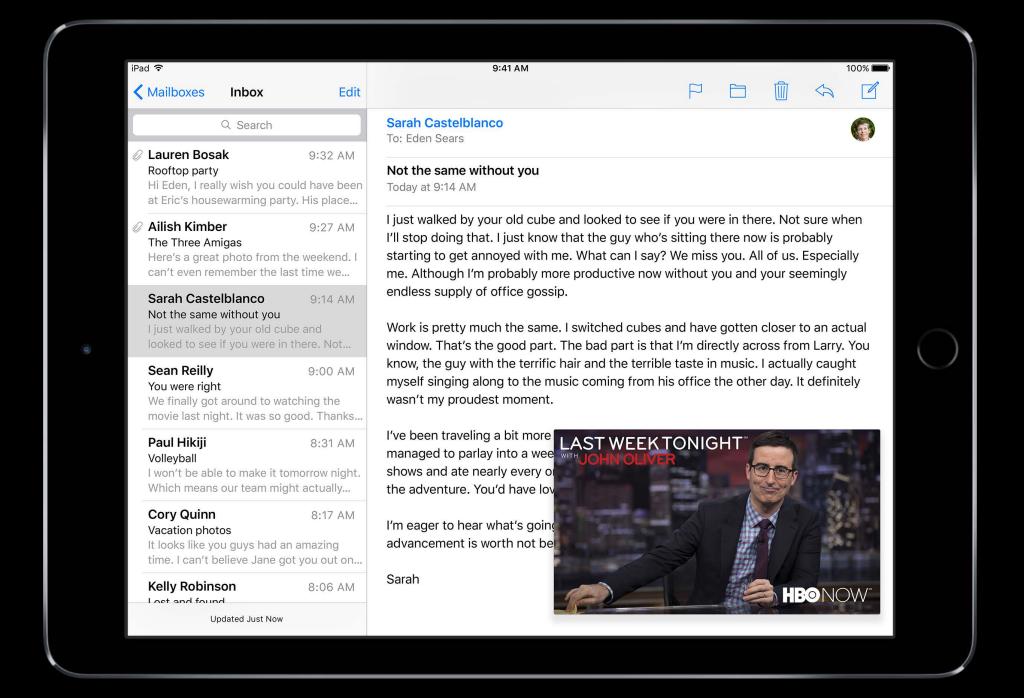
Abusing the VM system can result in process termination





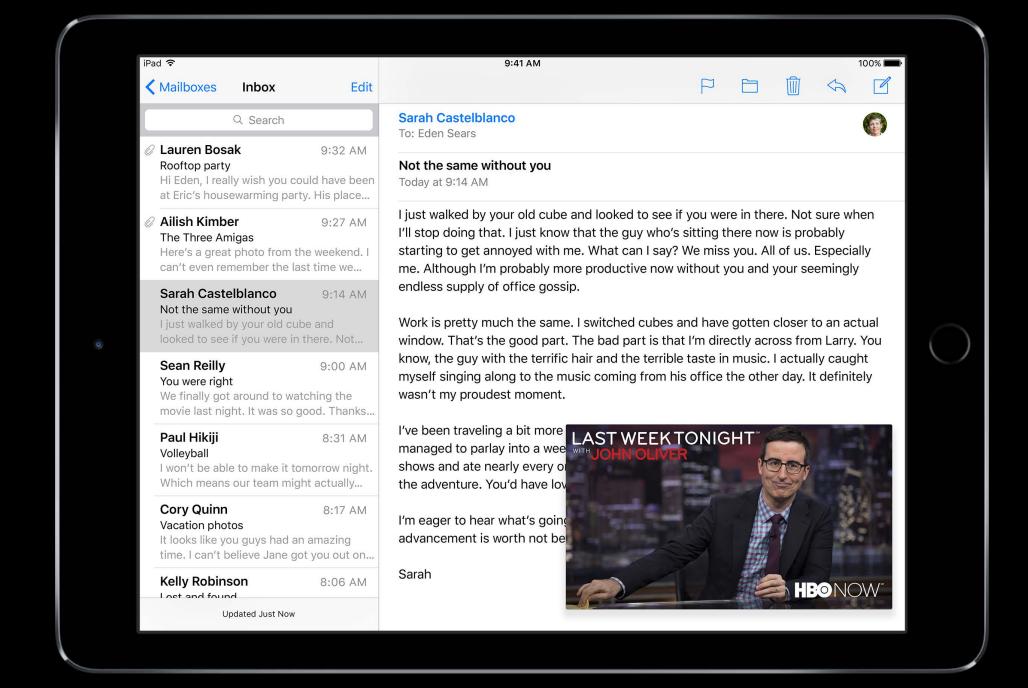


Use Instruments to identify and fix bugs



Use Instruments to identify and fix bugs

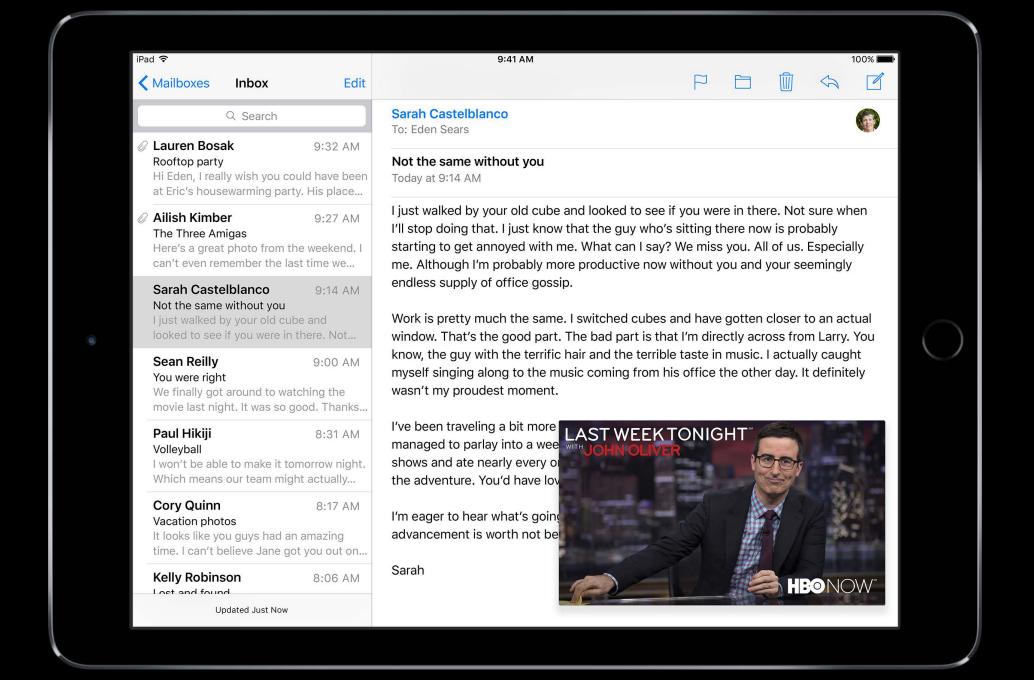
Prioritize your work appropriately and don't block the main thread



Use Instruments to identify and fix bugs

Prioritize your work appropriately and don't block the main thread

Identify and manage your working set

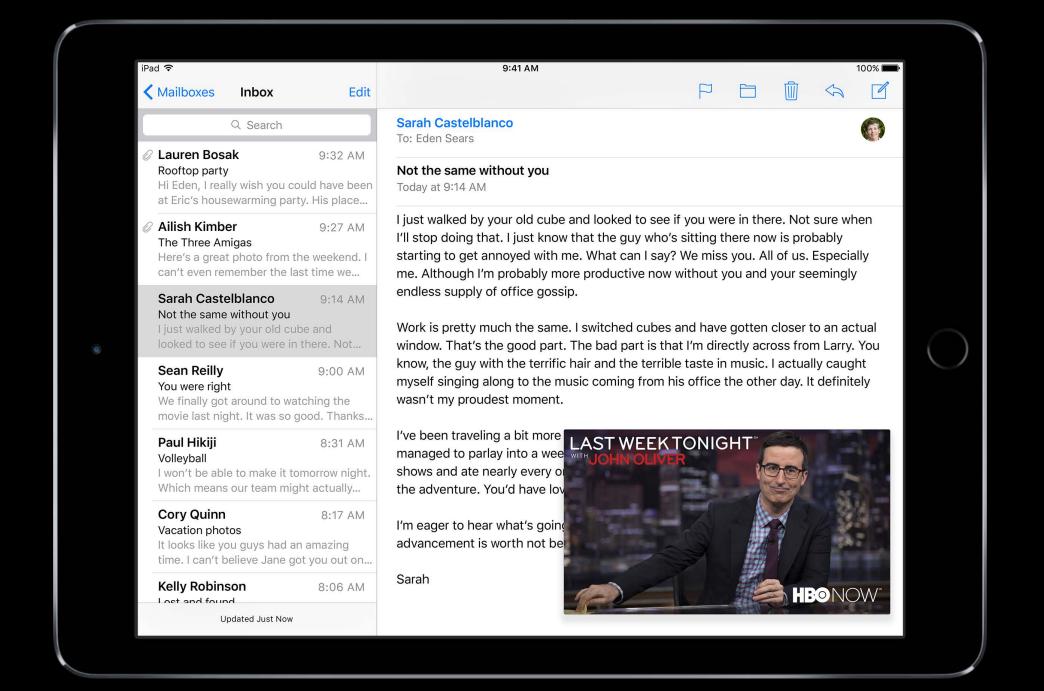


Use Instruments to identify and fix bugs

Prioritize your work appropriately and don't block the main thread

Identify and manage your working set

Use caches and respond to memory warnings



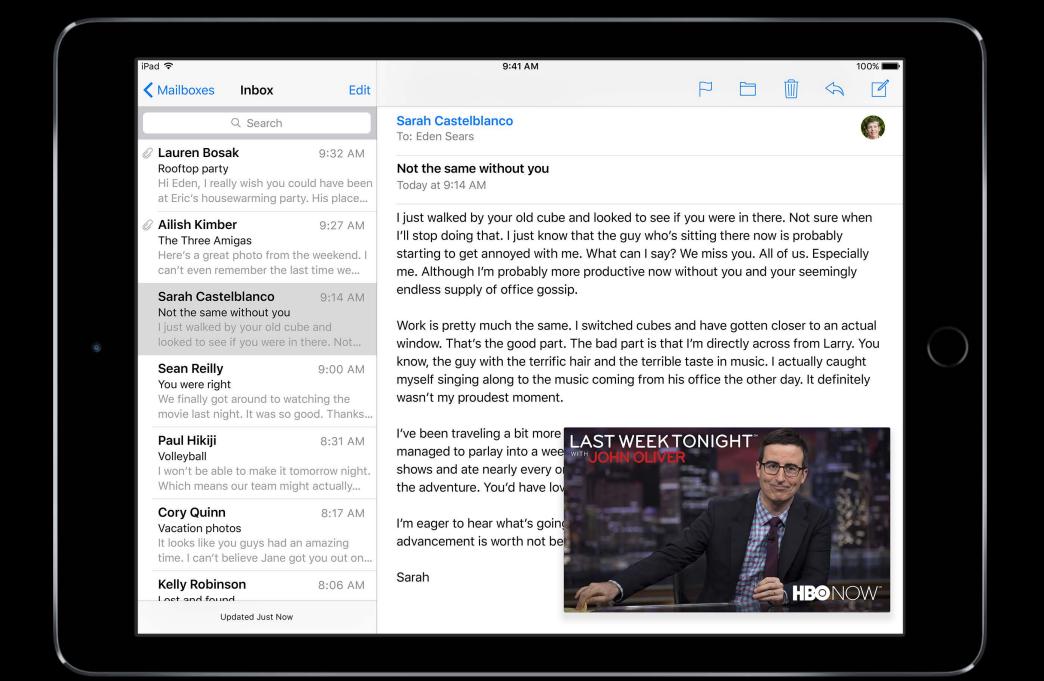
Use Instruments to identify and fix bugs

Prioritize your work appropriately and don't block the main thread

Identify and manage your working set

Use caches and respond to memory warnings

Leverage the virtual memory system to increase reclaimable memory



Use Instruments to identify and fix bugs

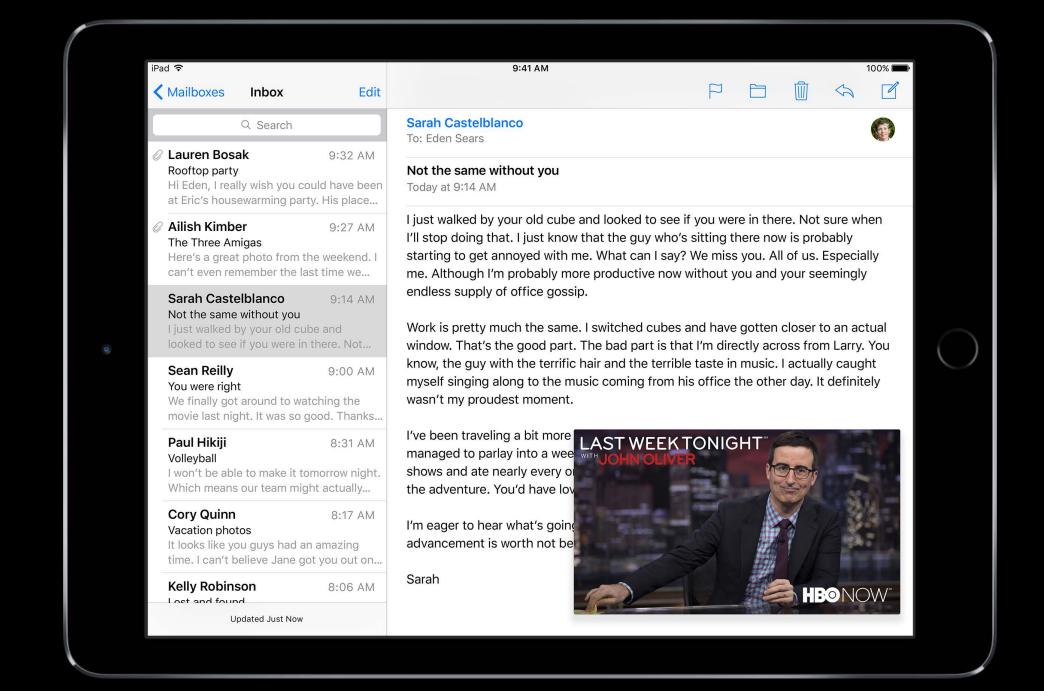
Prioritize your work appropriately and don't block the main thread

Identify and manage your working set

Use caches and respond to memory warnings

Leverage the virtual memory system to increase reclaimable memory

Great performance requires tradeoffs



"It is not the strongest or the most intelligent who survive, but those who can best manage change."

Charles Darwin (apocryphal)

More Information

Documentation
Adopting Multitasking
Enhancements on iPad
Performance Overview
Instruments User Guide
http://developer.apple.com/iOS

Technical Support

Apple Developer Forums

http://developer.apple.com/forums

Developer Technical Support http://developer.apple.com/support/ technical

General Inquiries

Curt Rothert, App Frameworks Evangelist rothert@apple.com

Related Sessions

Getting Started with Multitasking on iPad in iOS 9	Presidio	Tuesday 4:30PM
Multitasking Essentials for Media-Based Apps on iPad in iOS 9	Pacific Heights	Wednesday 2:30PM
Performance on iOS and watchOS	Presidio	Friday 11:00AM
Building Responsive and Efficient Apps with GCD	Nob Hill	Friday 10:00AM

Labs

Cocoa Touch and Multitasking Lab	Frameworks Lab B	Wednesday 9:00AM
Power and Performance Lab	Frameworks Lab C	Friday 12:00PM

ÓWWDC15