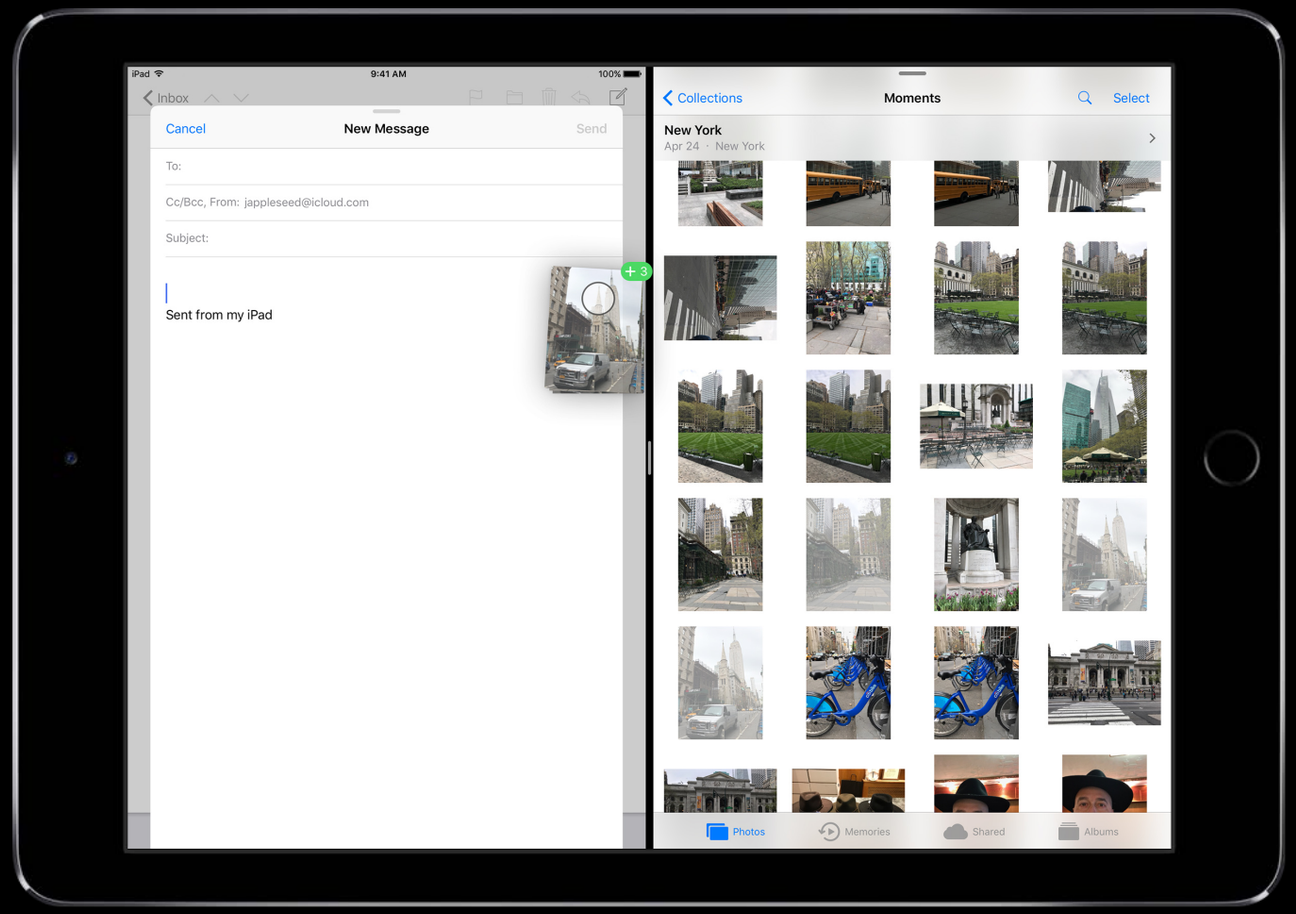
Drag and Drop

What is Drag and Drop?

A way to graphically move
or copy data



Drag and Drop on iOS

Goals

Drag and Drop on iOS

Goals

Responsive

Drag and Drop on iOS

Goals

Responsive

- On demand and asynchronous data delivery

Drag and Drop on iOS

Goals

Responsive

Secure

Drag and Drop on iOS

Goals

Responsive

Secure

- Data is only visible to destination

Drag and Drop on iOS

Goals

Responsive

Secure

- Data is only visible to destination
- Source may restrict access

Drag and Drop on iOS

Goals

Responsive

Secure

A great Multi-Touch experience

Drag and Drop on iOS

Goals

A great Multi-Touch experience

- The interface is live
- Deep integration with all of iOS
- Great visual feedback
- Hover to navigate
- Items can be added
- Transfer drags between fingers
- Multiple drag interactions

Drag and Drop on iOS

Phases of a drag session

Drag and Drop on iOS

Phases of a drag session

Lift 

Drag 

Set Down 

Data Transfer 

Drag and Drop on iOS

Phases of a drag session

Lift  Drag  Set Down  Data Transfer 

Long press

Lift preview

Drag and Drop on iOS

Phases of a drag session

Lift 

Drag 

Set Down 

Data Transfer 

Long press

Previews

Lift preview

Tap to add

Spring-loading

Drag and Drop on iOS

Phases of a drag session

Lift  Drag  Set Down  Data Transfer 

Long press	Previews	Cancel
Lift preview	Tap to add	Drop
	Spring-loading	Targeting

Drag and Drop on iOS

Phases of a drag session

Lift 

Drag 

Set Down 

Data Transfer 

Long press

Lift preview

Previews

Tap to add

Spring-loading

Cancel

Drop

Targeting

Representations

Lazy delivery

- Background
- By File Provider

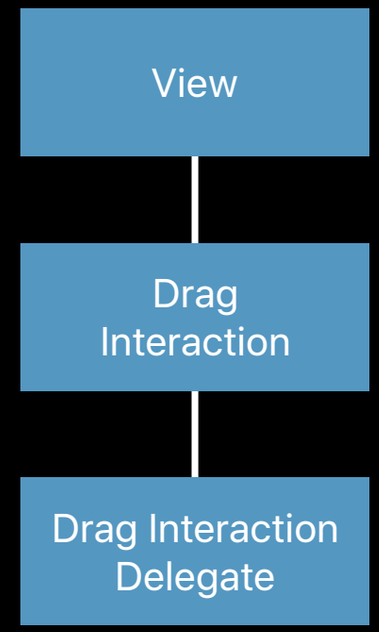
Progress

Enabling a Drag

Concepts - UIDragInteraction

A drag interaction is attached to a view

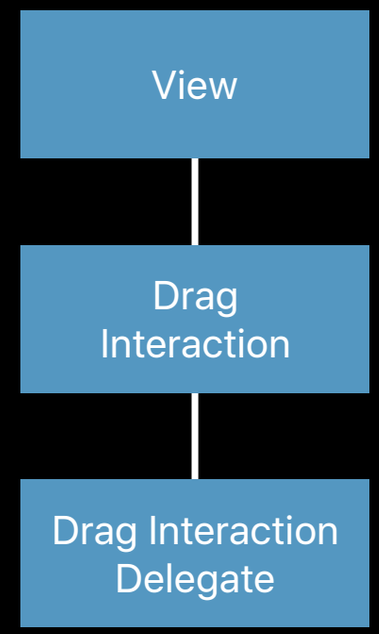
```
import UIKit

let view: UIView = ...
let delegate: UIDragInteractionDelegate = ...

let dragInteraction = UIDragInteraction(delegate: delegate)
view.addInteraction(dragInteraction)
```



Lift Phase

Concepts - UIDragInteraction

The delegate provides drag items
when the view lifts

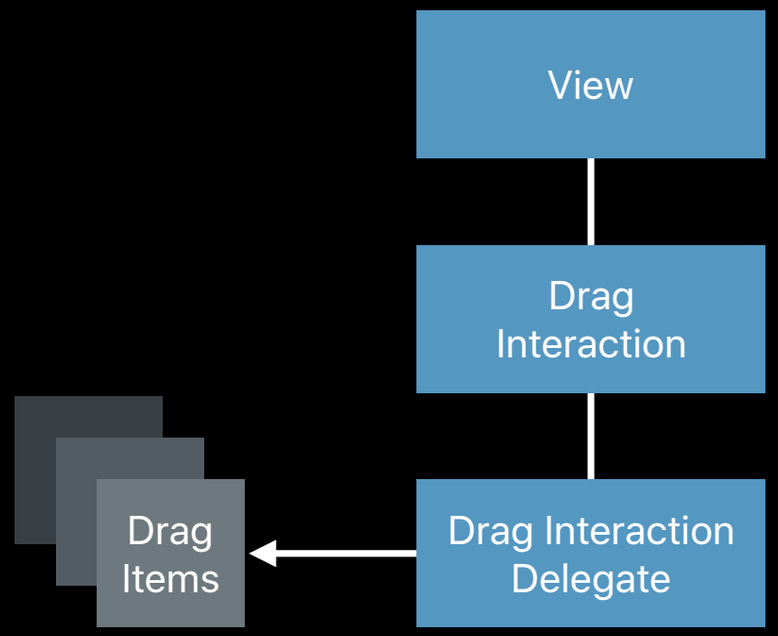


Lift Phase

Concepts - UIDragInteraction

The delegate provides drag items when the view lifts

No drag items -> drag gesture fails



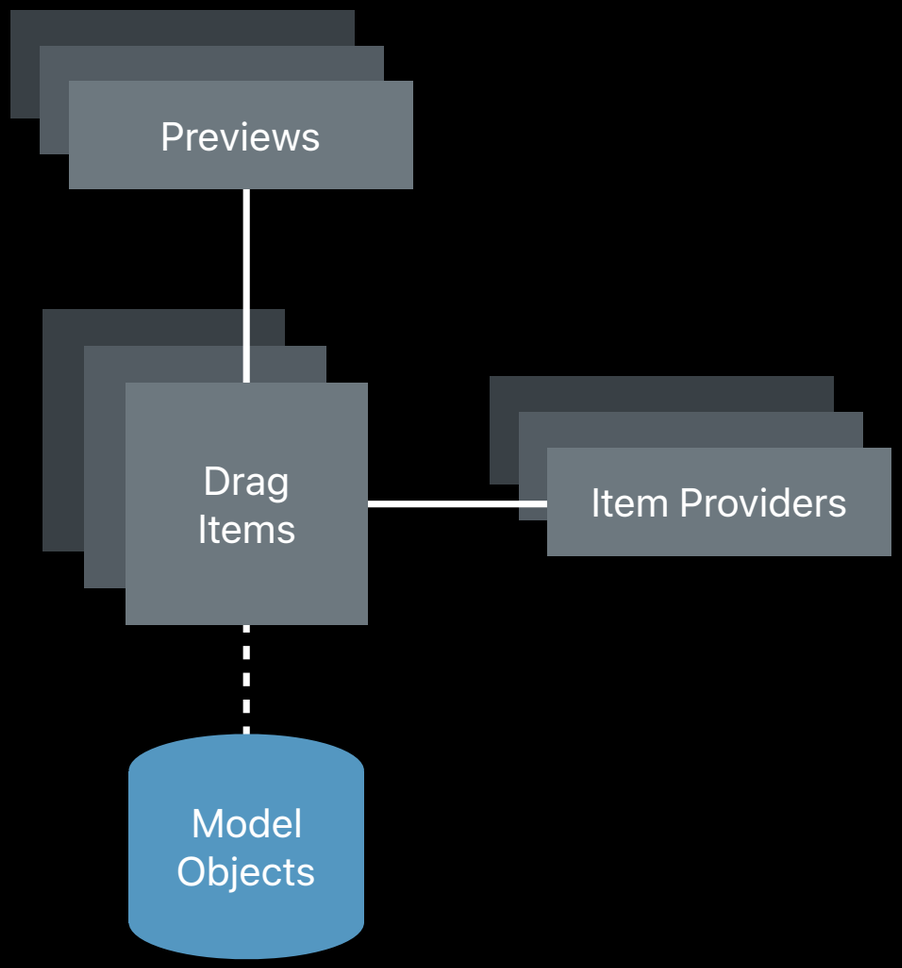
Drag Items

Concepts - UIDragItem

A drag item represents a model object

A drag item embodies

- Drag preview
- Item provider

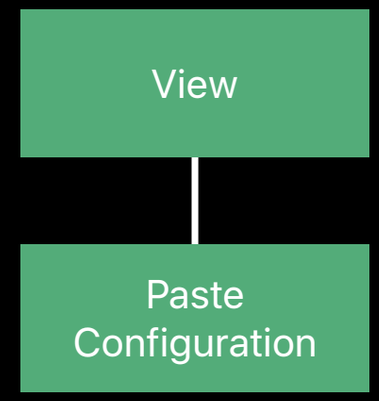


Enabling a Drop

Concepts - UIPasteConfiguration

UIResponders have a new paste configuration property

```
// Indicate you can accept or paste strings

let config = UIPasteConfiguration(typeIdentifiersForAcceptingClass:
                                  NSString.self)
view.pasteConfiguration = config
```

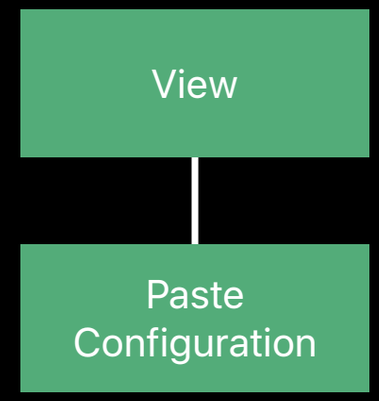


Enabling a Drop

Concepts - UIPasteConfiguration

UIResponders have a new paste configuration property

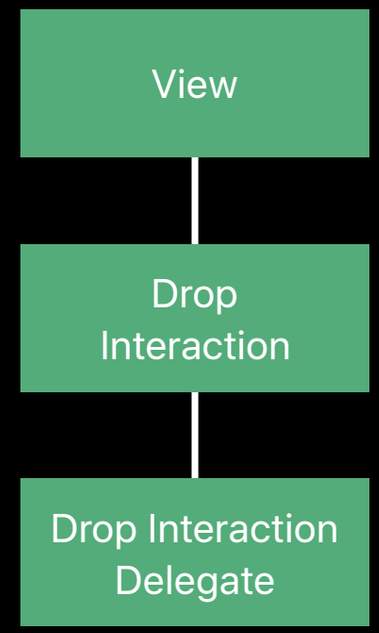
```
// Will be called for both Drag and Drop, and Copy/Paste

override func paste(itemProviders: [NSItemProvider]) {
}
```



Enabling a Drop

Concepts - UIDropInteraction

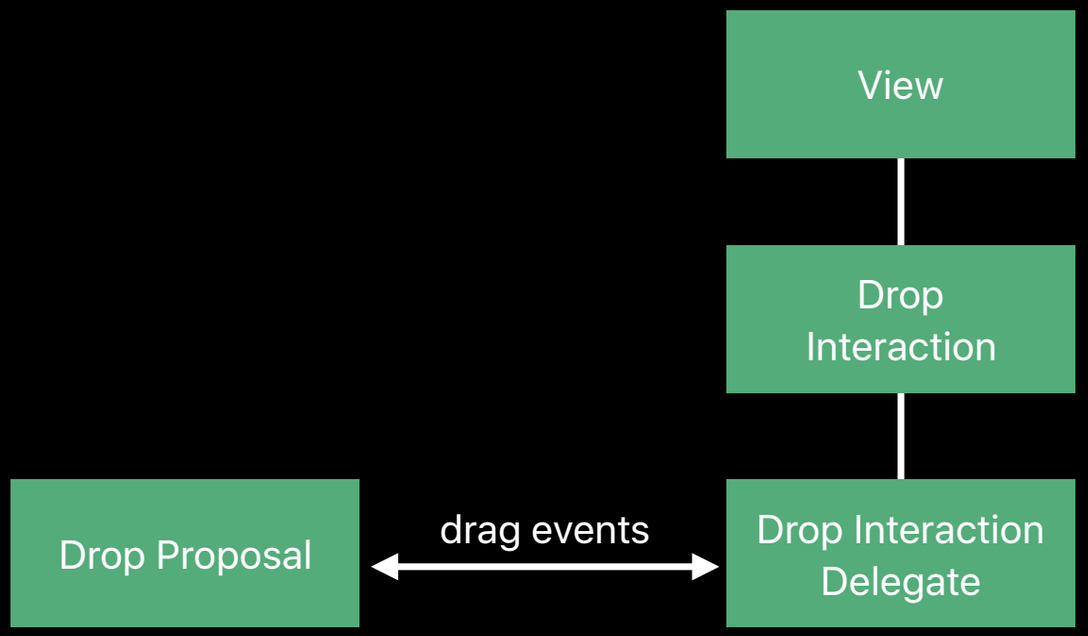
A drop interaction is attached to a view



Drag Phase

Concepts - UIDropInteraction

The delegate responds to drag events

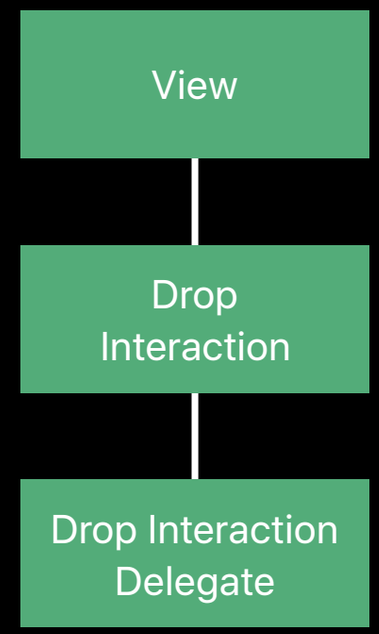


Set Down Phase

Concepts - UIDropInteraction

On touch up, the drag session may be cancelled

- The drag preview animates back

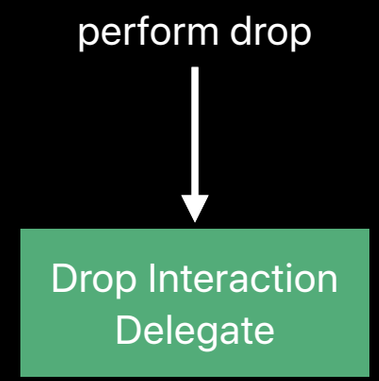


Set Down Phase

Concepts - UIDropInteraction

Or the drop is accepted

- The delegate is told to perform drop

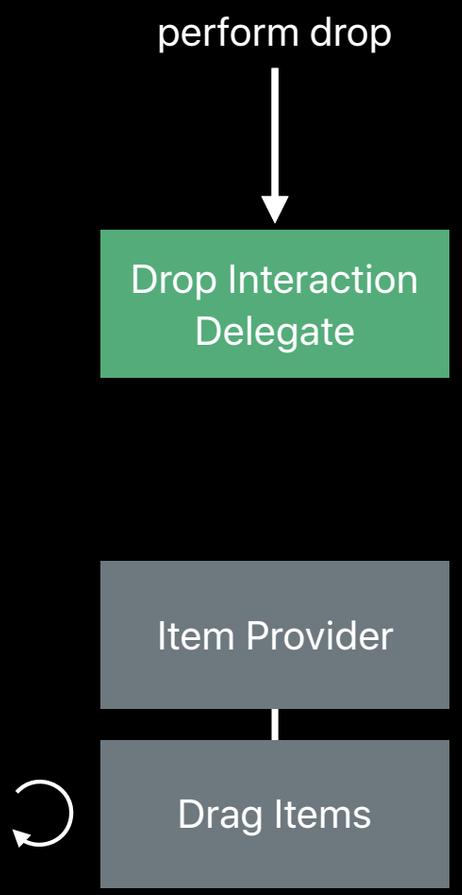


Data Transfer Phase

Concepts - UIDropInteraction

Or the drop is accepted

- The delegate is told to perform drop



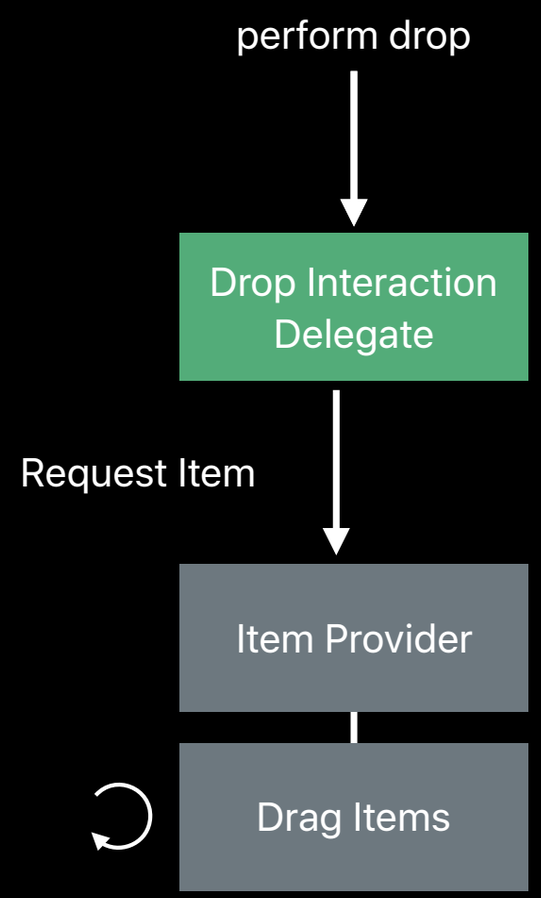
Data Transfer Phase

Concepts - UIDropInteraction

Or the drop is accepted

- The delegate is told to perform drop

Delegate requests data representation of items



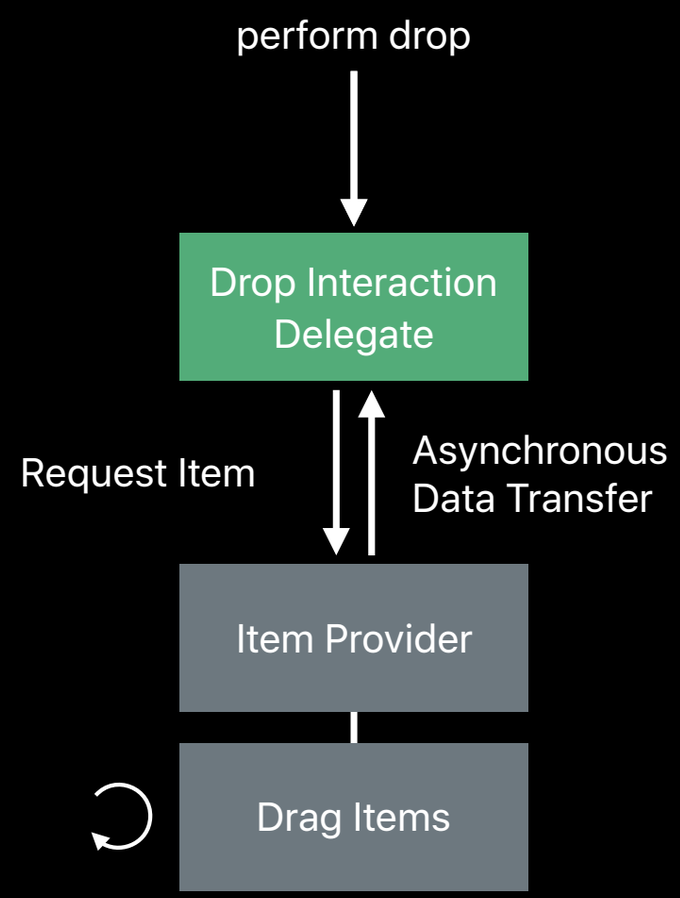
Data Transfer Phase

Concepts - UIDropInteraction

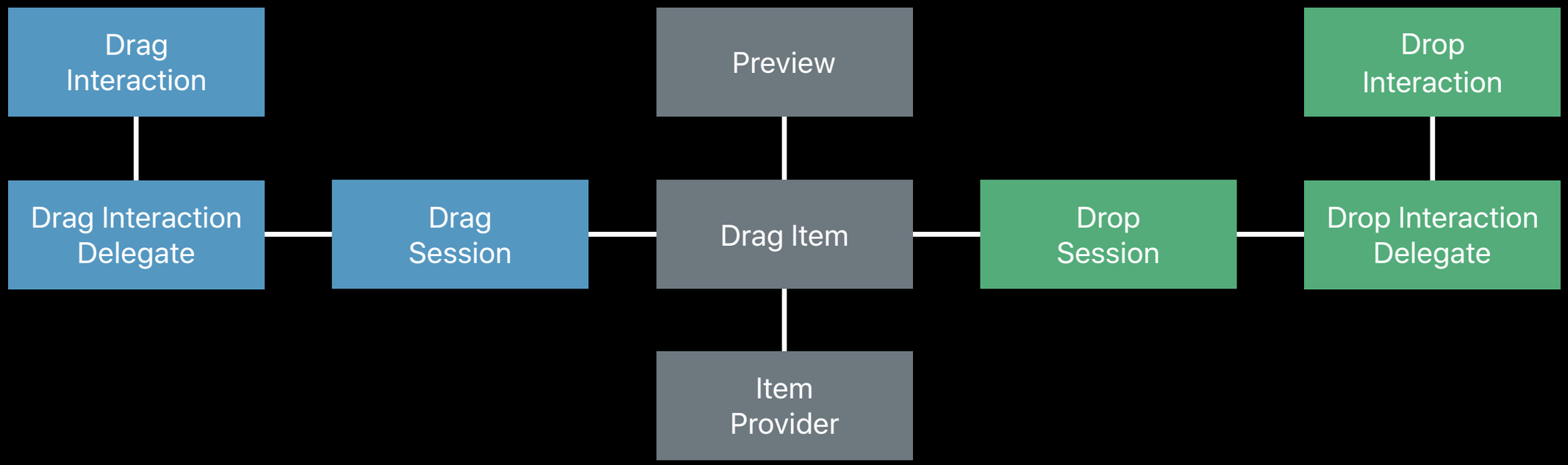
Or the drop is accepted

- The delegate is told to perform drop

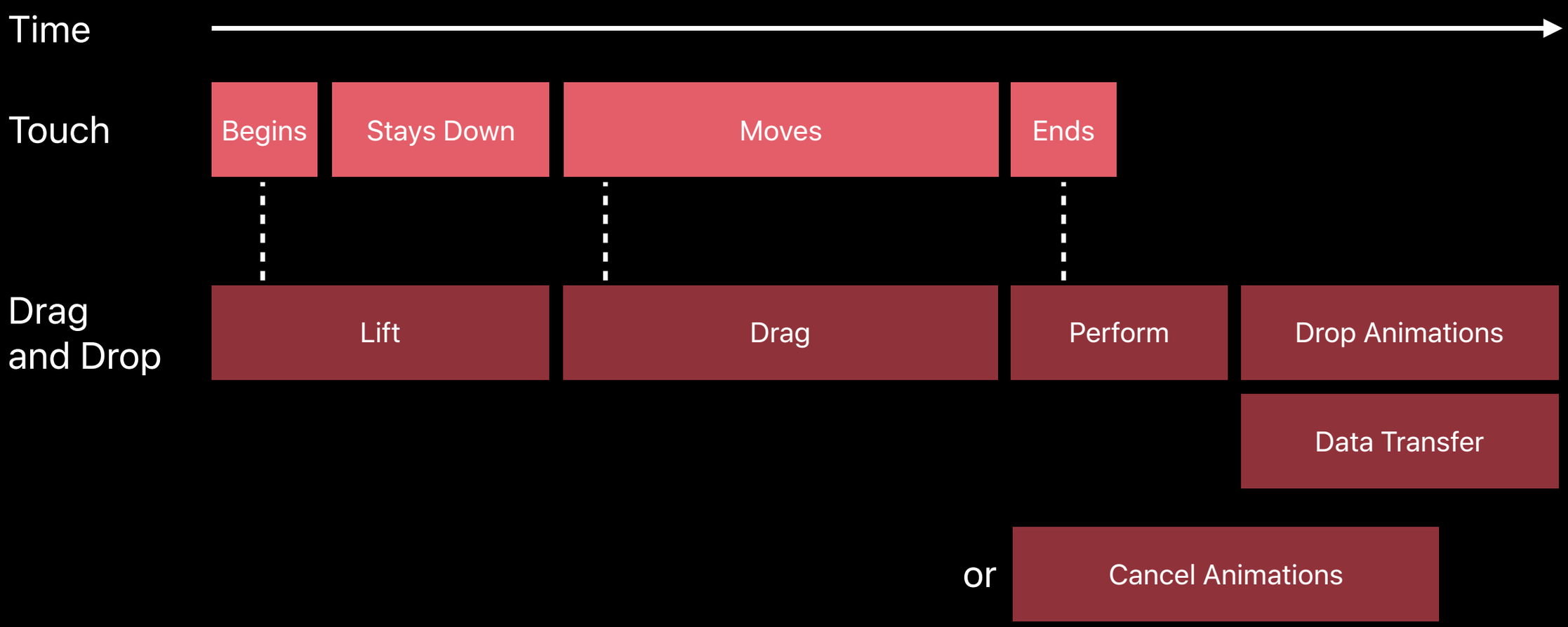
Delegate requests data representation of items



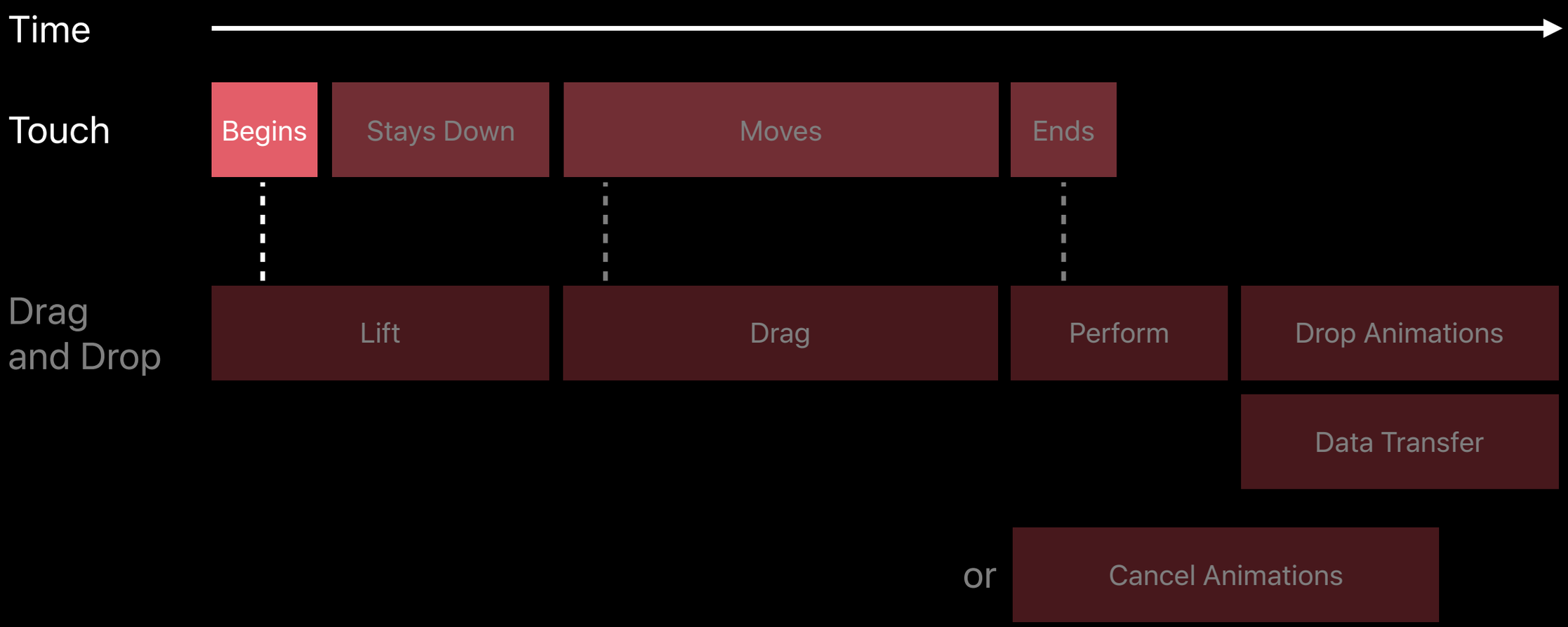
API Roadmap


Drag Interaction
Drag Interaction Delegate
Drag Session
Preview
Drag Item
Item Provider
Drop Session
Drop Interaction
Drop Interaction Delegate

Drag and Drop Timeline



API Essentials—1

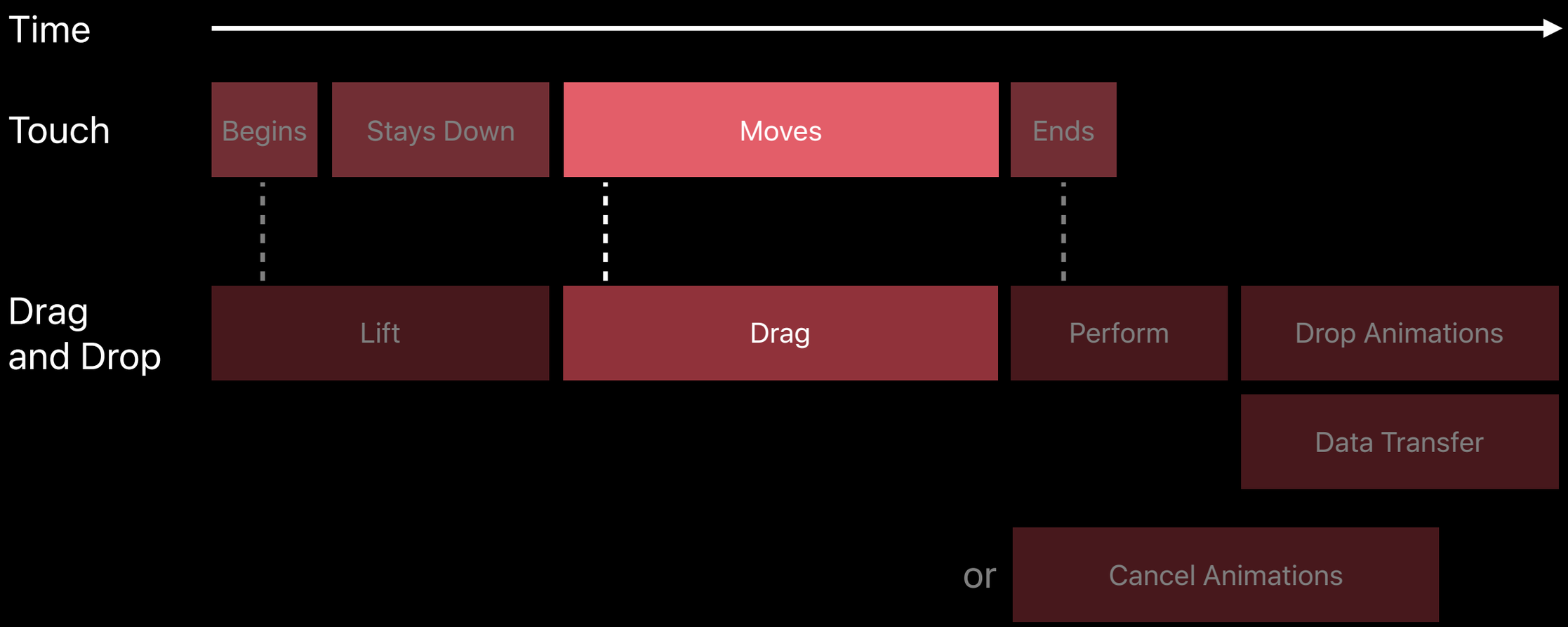


API Essentials—1

Get the items to drag

```
func dragInteraction(_ interaction: UIDragInteraction,
                     itemsForBeginning session: UIDragSession) -> [UIDragItem] {
    let itemProvider = NSItemProvider(object: "Hello World" as NSString)
    let dragItem = UIDragItem(itemProvider: itemProvider)
    return [ dragItem ]
}
```

API Essentials—2



API Essentials—2

Get the drop proposal

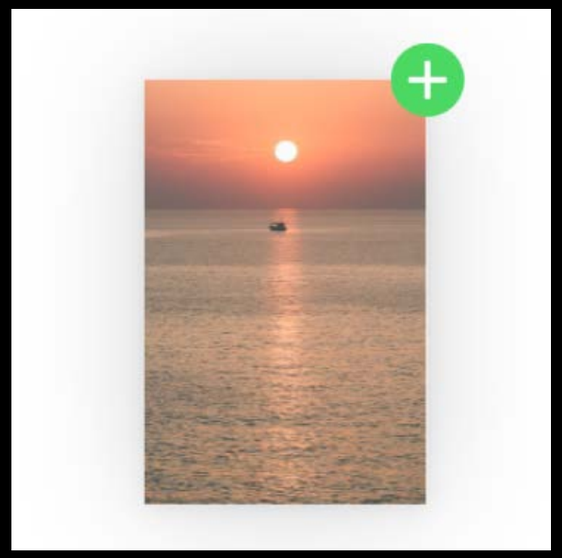
```
func dropInteraction(_ interaction: UIDropInteraction,
                     sessionDidUpdate session: UIDropSession) -> UIDropProposal {
    return UIDropProposal(operation: UIDropOperation)
}
```

API Essentials—2

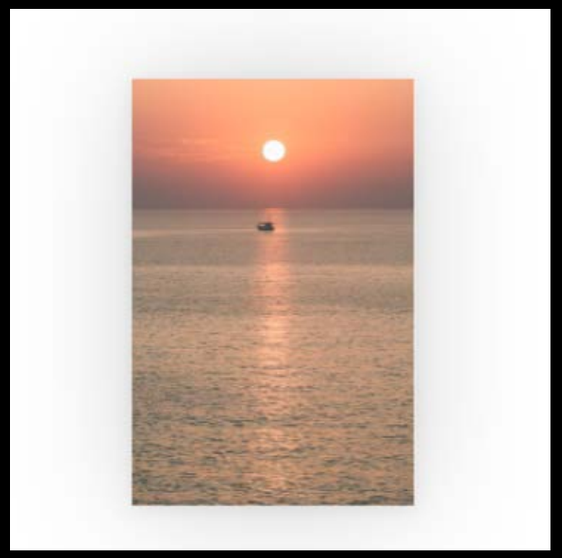
UIDropOperation

`.cancel`

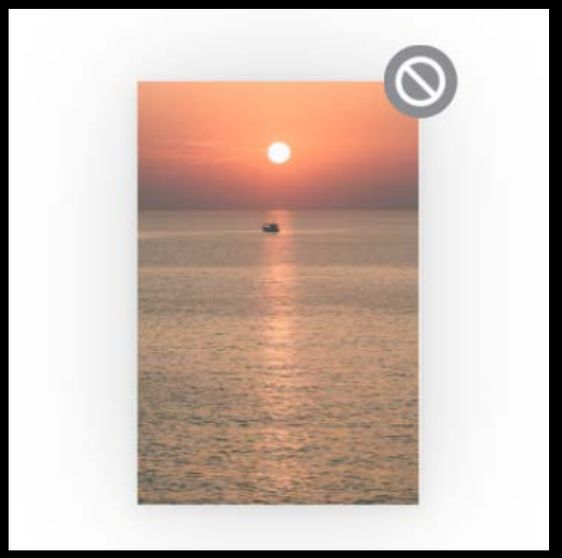
`.copy`



`.move`



`.forbidden`

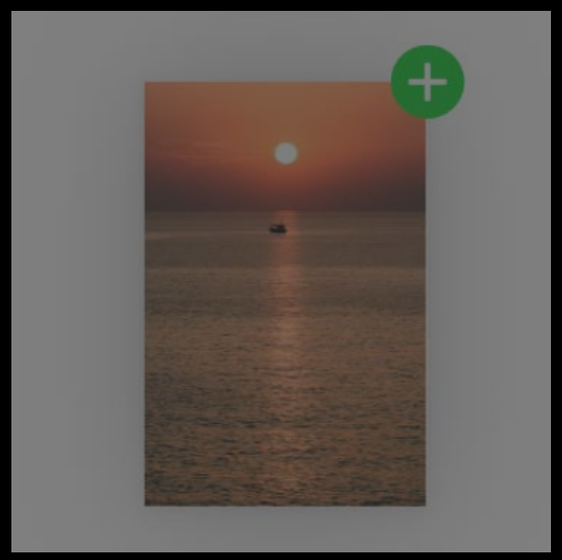


API Essentials—2

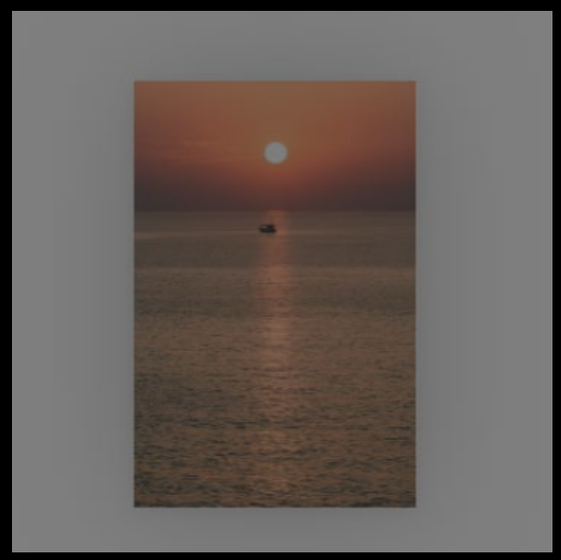
UIDropOperation

`.cancel`

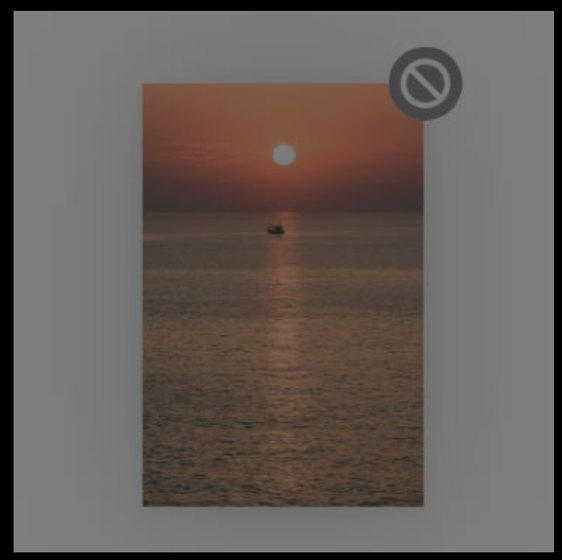
`.copy`



`.move`



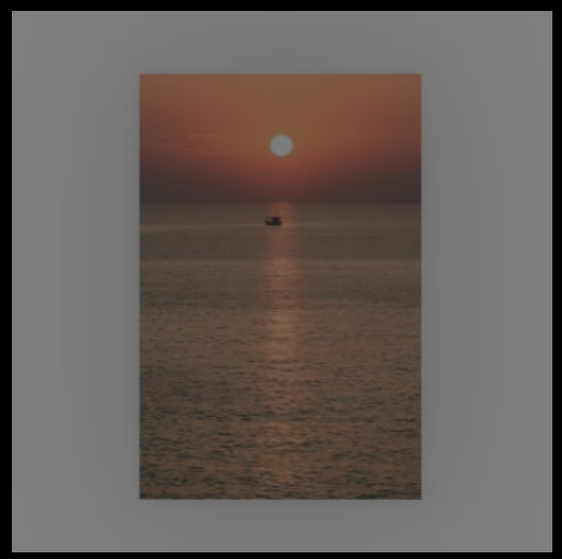
`.forbidden`



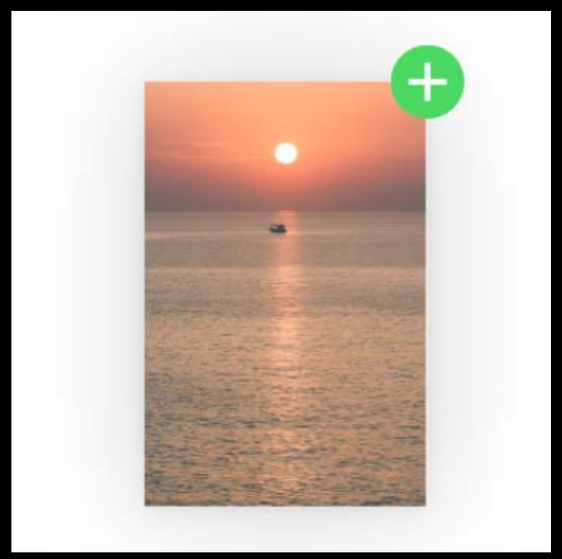
API Essentials—2

UIDropOperation

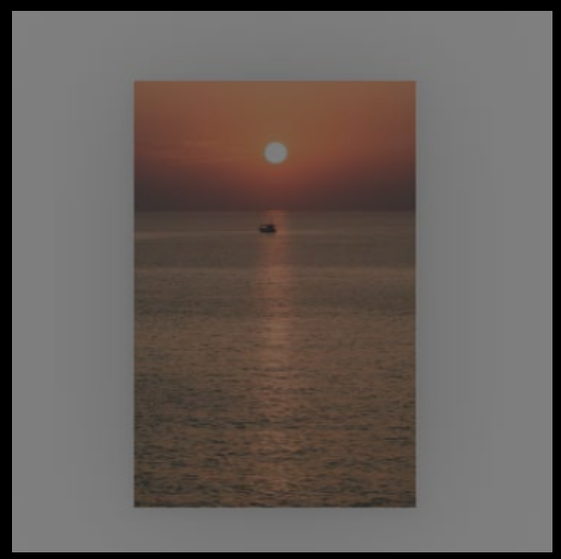
`.cancel`



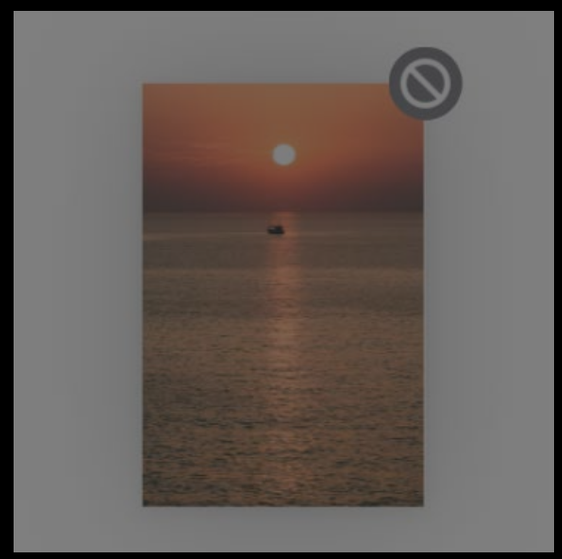
`.copy`



`.move`



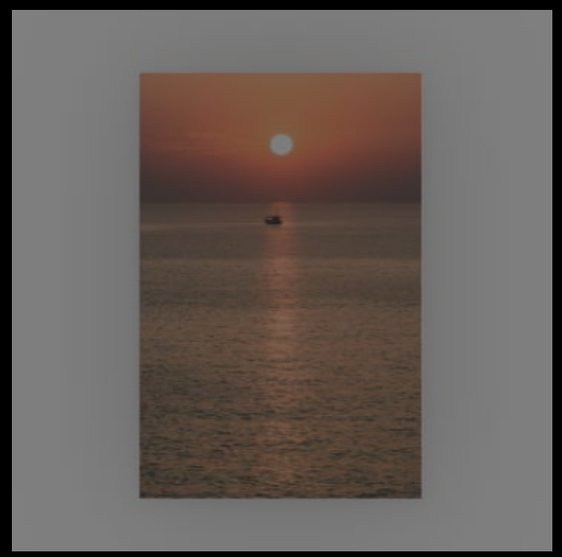
`.forbidden`



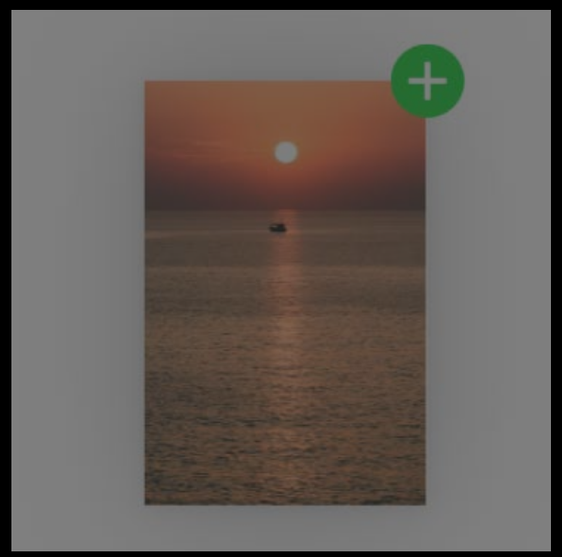
API Essentials—2

UIDropOperation

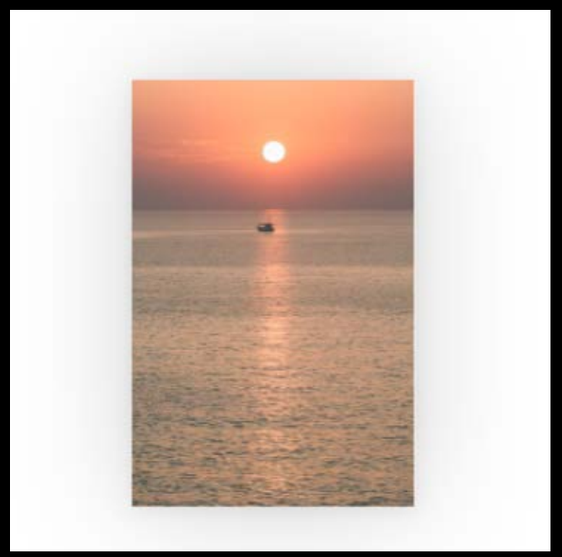
`.cancel`



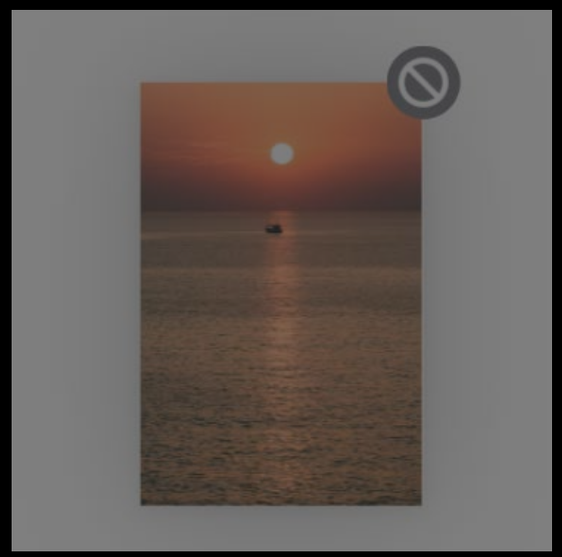
`.copy`



`.move`



`.forbidden`



API Essentials—2

UIDropOperation

Delegates must cooperate
to make it look like a move

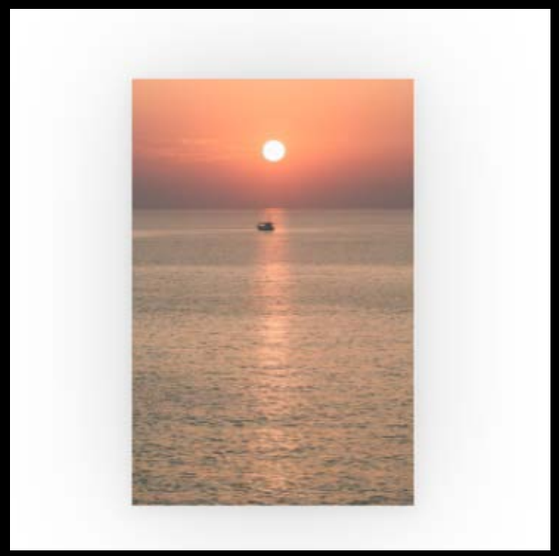
Only within a single app

Drag interaction delegate
must allow moves

Drop interaction delegate checks

`UIDropSession allowsMoveOperation`

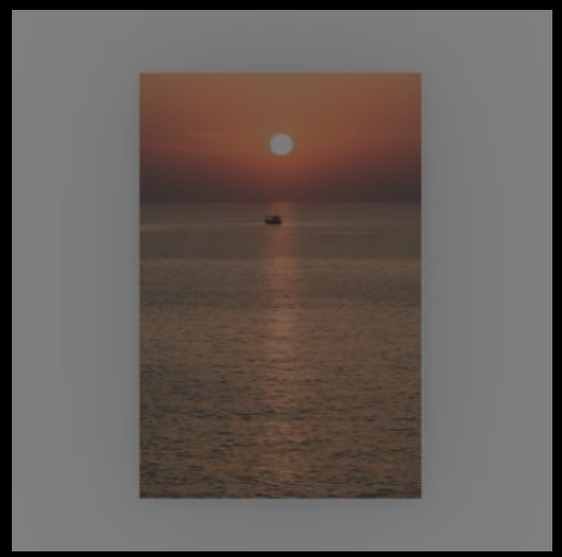
`.move`



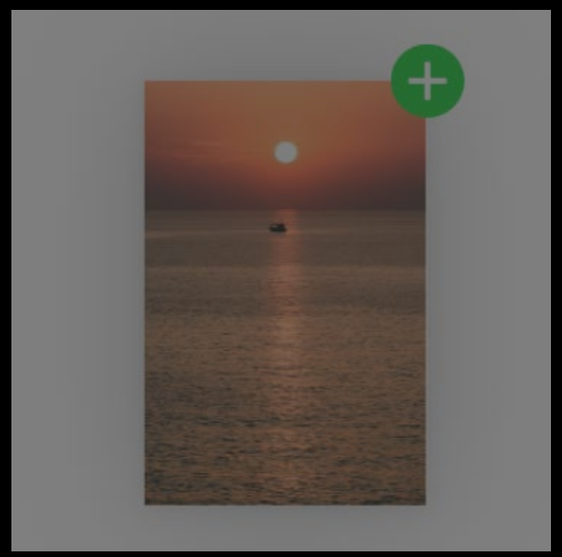
API Essentials—2

UIDropOperation

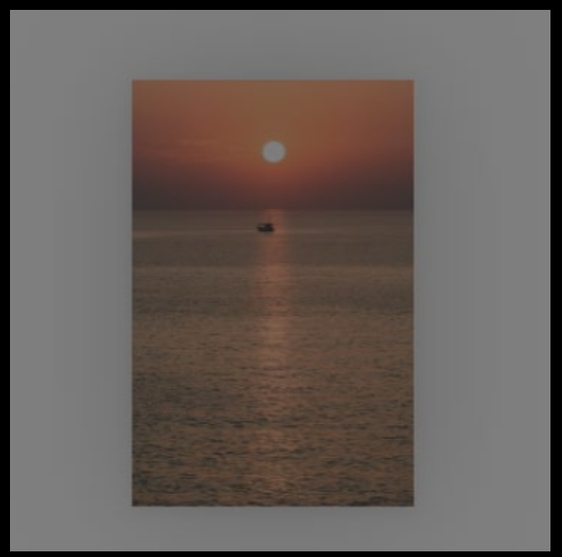
`.cancel`



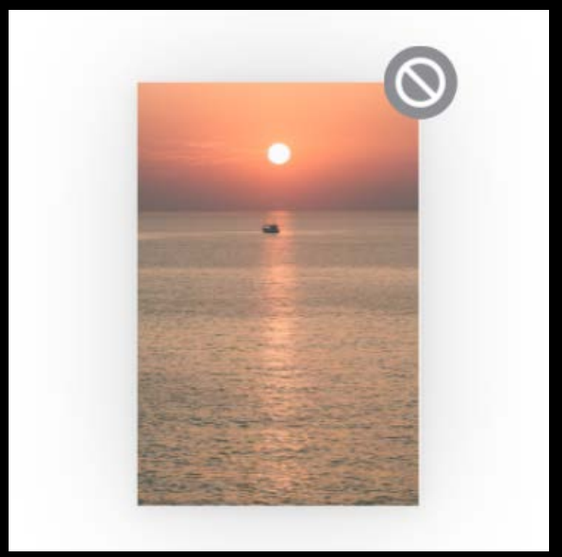
`.copy`



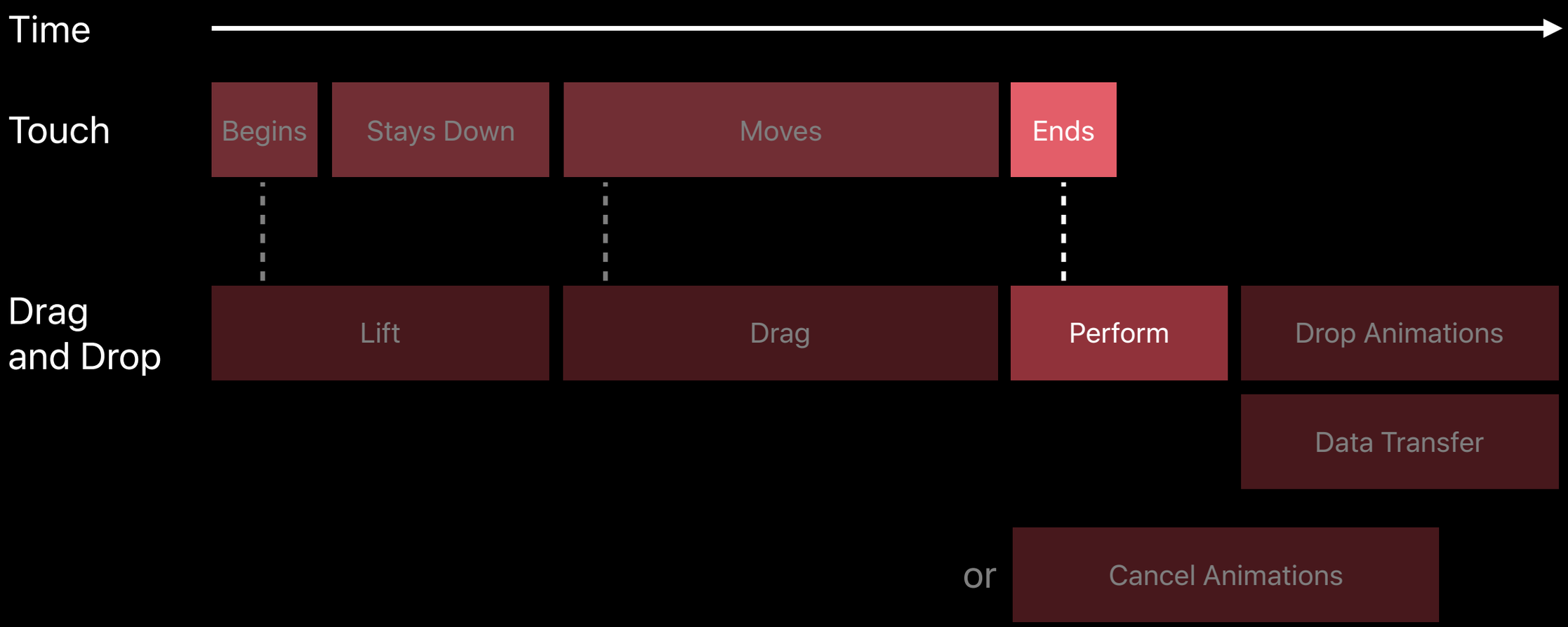
`.move`



`.forbidden`



API Essentials—3



API Essentials—3

Perform the drop

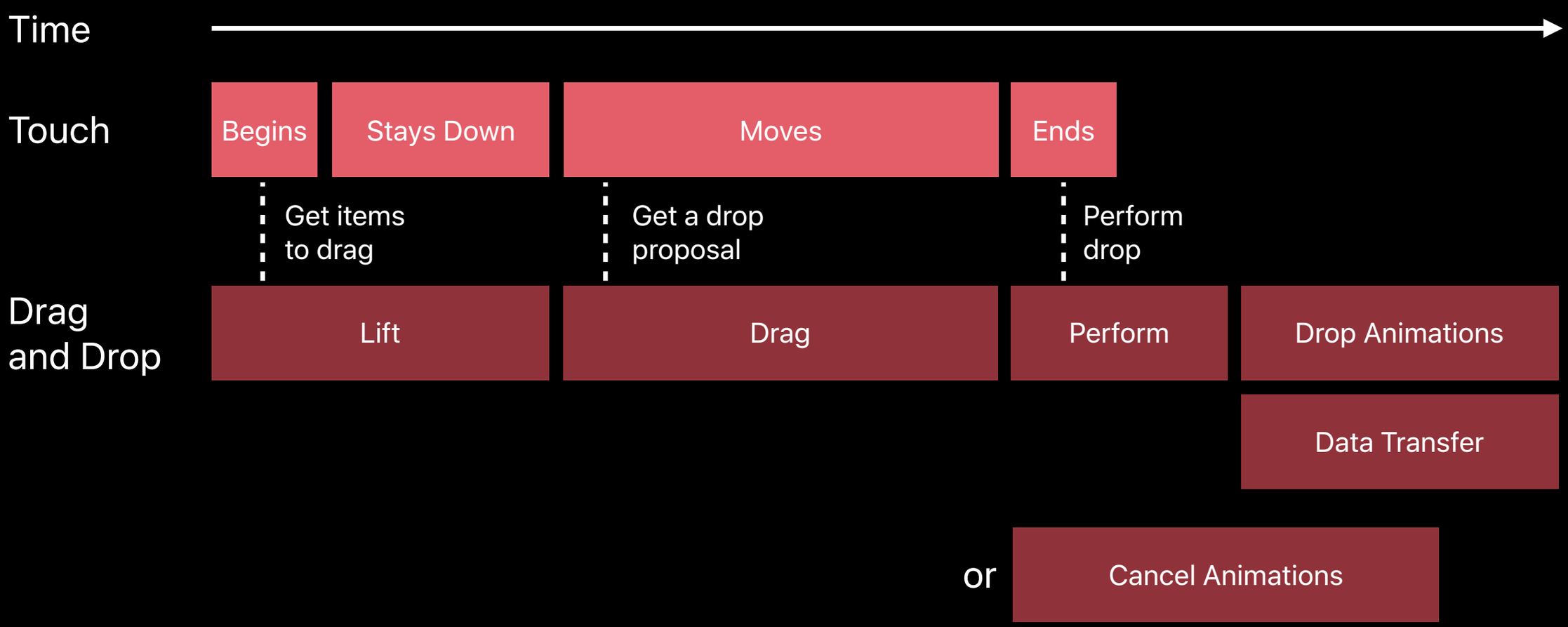
```
func dropInteraction(_ interaction: UIDropInteraction, performDrop session: UIDropSession) {
    session.loadObjects(ofClass: UIImage.self) { objects in
        for image in objects as! [UIImage] {
            self.imageView.image = image
        }
    }
}
```

API Essentials—3

Perform the drop

```
func dropInteraction(_ interaction: UIDropInteraction, performDrop session: UIDropSession) {
    for item in session.items {
        item.itemProvider.loadObject(ofClass: UIImage.self) { (object, error) in
            if object != nil {
                DispatchQueue.main.async {
                    self.imageView.image = (object as! UIImage)
                }
            }
            else {
                // Handle the error
            }
        }
    }
```

API Essentials



Demo

References

Introduction to Drag and Drop - https://developer.apple.com/videos/play/wwdc2017/203/

Mastering Drag and Drop - https://developer.apple.com/videos/play/wwdc2017/213/

Drag and Drop with collection and tableview - https://developer.apple.com/videos/play/wwdc2017/223/

Data delivery with drag and drop - https://developer.apple.com/videos/play/wwdc2017/227/