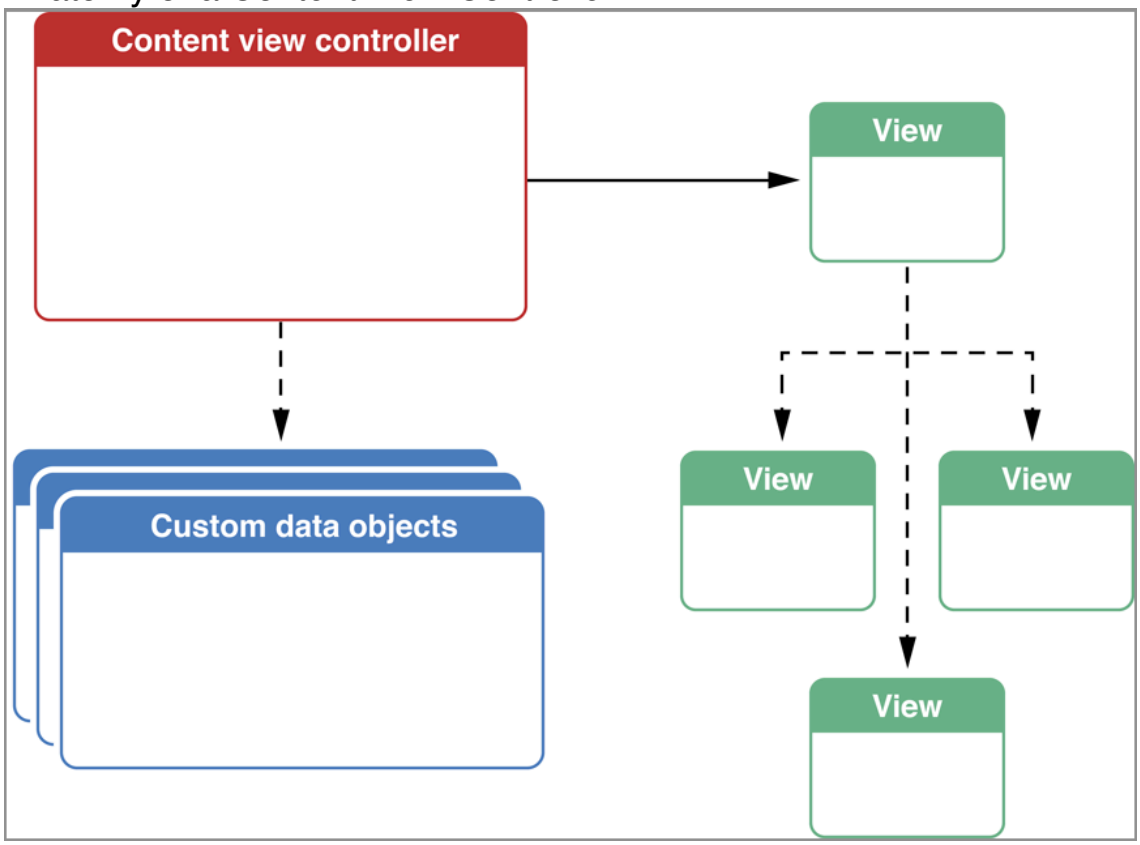
Creating Custom Content View Controllers

Anatomy of a Content View Controller



> View Controllers Manage Resources

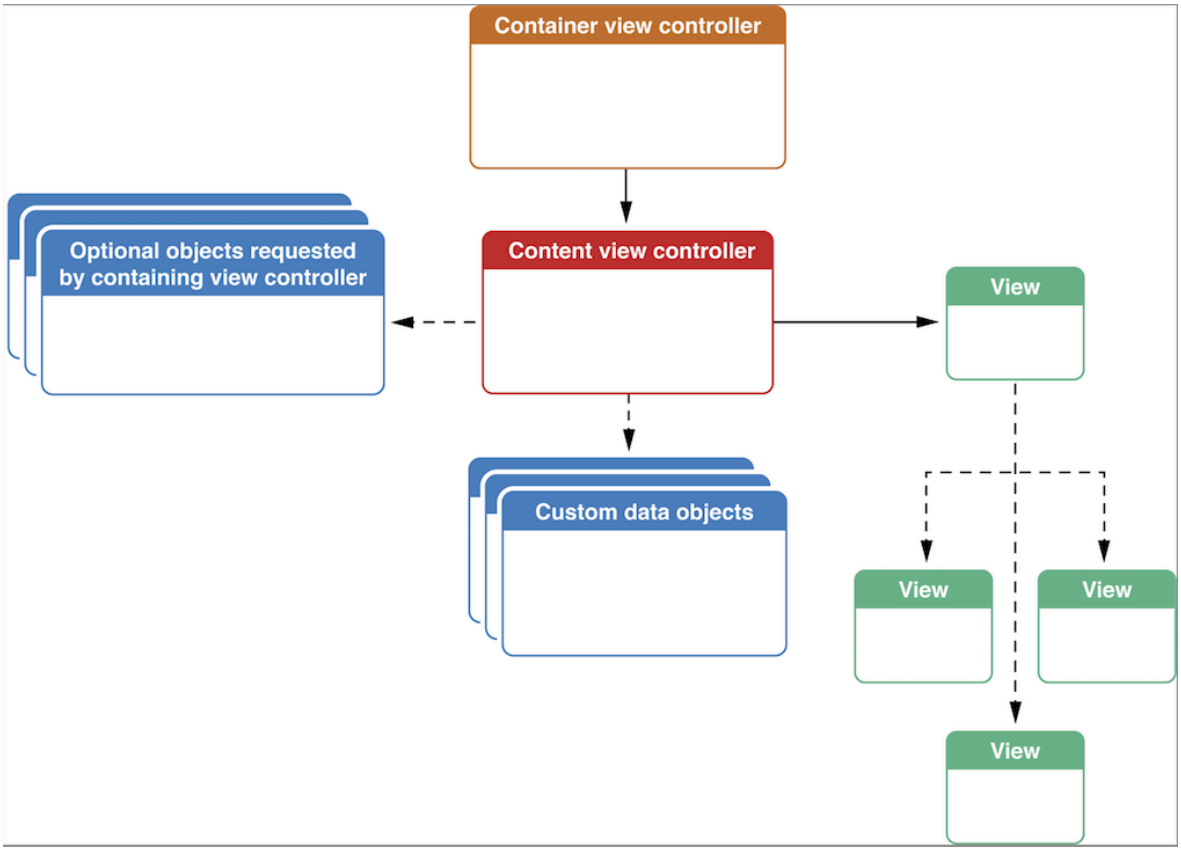
> View Controllers Manage Views

> View Controllers Respond to Events

> View Controllers Coordinate with Other Controllers

> View Controllers Often Work with Containers

1. If your view controller is placed inside a container view controller, the container imposes an additional constraints.



> View Controllers May Be Presented by Other View Controllers
1. There are several reasons you might present a view controller:
=> To gather information from the user immediately.
=> To present some content temporarily.
=> To change work modes temporarily.
=> To implement alternate interfaces for different device orientations.
=> To present a new view hierarchy with a specific type of animated transition (or no transition).

2. In almost all cases, the presented view controller implements a delegate. The presented view controller uses the delegate to communicate with the presenting view controller.

Designing Your Content View Controller
1. Before writing any code in your view controller, you should be able to answer some basic questions about how you intend to use it.
=> Are you using a storyboard to implement the view controller?
=> When is it instantiated?
=> What data does it show?
=> What tasks does it perform?
=> How is its view displayed onscreen?

=> How does it collaborate with other view controllers?

> Use a Storyboard to Implement Your View Controller
1. You always use a storyboard unless you have a strong reason not to.

2. When you use storybooks:
=> iOS usually instantiates your view controller for you automatically.
=> To finish instantiating it, you override its awakeFromNib method.
=> Other objects configure it through its properties.
=> You create its view hierarchy and other related objects in Interface Builder.
=> Relationships with other view controllers are created in the storyboard.

3. If you design your view controller to be used programmatically:
=> The view controller is instantiated by allocating and initializing it.
=> You create an custom initialization method to initialize the view controller.
=> Other objects configure the view controller using its initialization method and by configuring its properties.
=> You override the loadView method to programmatically create and configure its view hierarchy.
=> Relationships with other view controllers are created by writing code.

> Know When Your Controller Is Instantiated

> Know What Data Your View Controller Shows and Returns

> Know What Tasks Your Controller Allows the User to Perform

> Know How Your View Controller Is Displayed Onscreen

> Know How Your Controller Collaborates with Other Controllers

Examples of Common View Controller Designs

Implementation Checklist for Custom Content View Controllers