SomaCore Diagram

Wire/Mediator

wires | mediators | wires

Wires and Mediators are the links, or "glue", between elements that shouldn't contain Framework code, such as Views and Models. Wires can register and send Commands, create other Wires, create Views and Models. Wires can have many roles and Mediators have a relation one-to-one with a view.

Model

data | data | data

View

components | components

Controller

commands | commands

Events caught by the Framework will trigger Commands. Commands can create Wires and control the Framework flow. Asynchronous Commands, Parallel Commands and Sequence Commands can be created as needed.
The Events used to execute Commands can be monitored from in the Wires, Mediators and Models. You can add listeners to commands. If the events cancellable, you can easily prevent them to occurs using the Event preventDefault method.

Event

events | events | events

Events with a bubble property set to true and mapped to a Command class become Commands. Because they are native Flash Events, they can be used in the Views and Models.

Soma ↔ Soma Injector
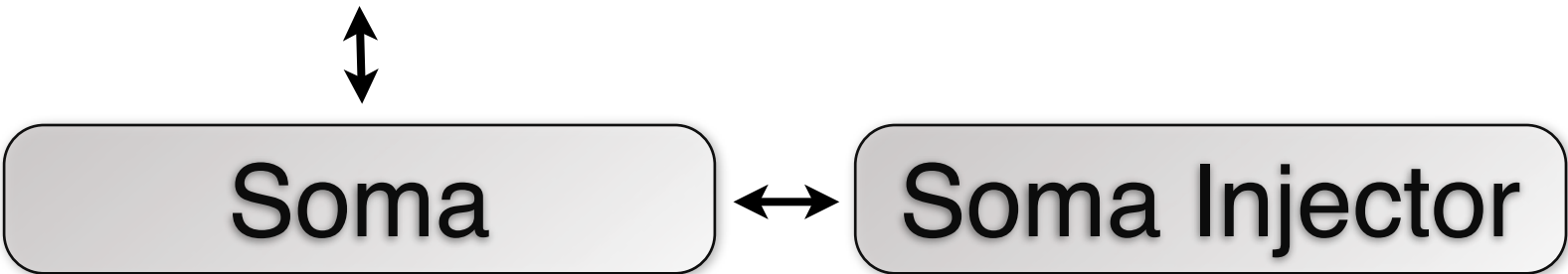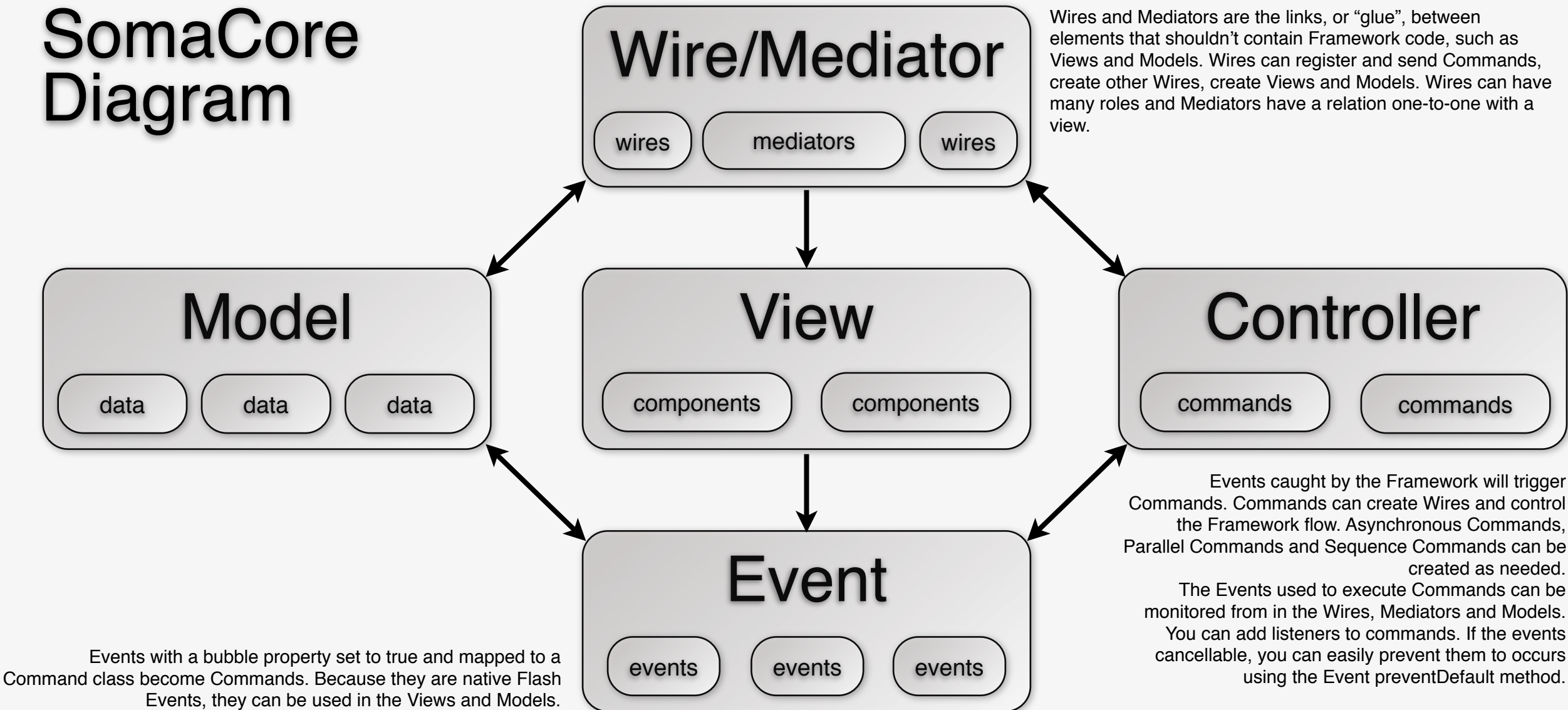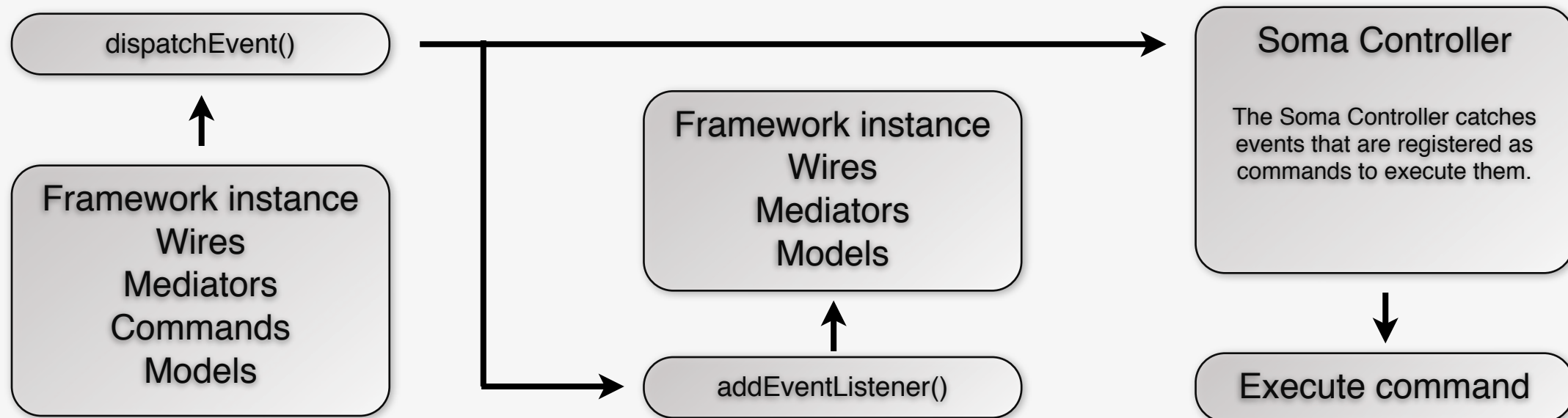
There is no Singleton in Soma,
You can extend the class Soma (and implement ISoma) to start to build your own application.
Your Soma Application will be the gate to talk to the Framework and is easily accessed from the Wires, the Mediators and the Commands. Your application can create Wires and Mediators, register Commands, instantiates Soma Plugins, initialise the application, and so on.

SomaCore
Commands workflow from framework elements

dispatchEvent()

Framework instance
Wires
Mediators
Commands
Models

Framework instance
Wires
Mediators
Models

addEventListener()

Soma Controller

The Soma Controller catches events that are registered as commands to execute them.

Execute command

Wires, Mediators, Commands or Models share the same EventDispatcher instance: the Framework instance itself.

Commands can be dispatched from all these Framework elements using the Flash built-in method:

dispatchEvent(MyEvent.DO_SOMETHING);

The Framework instance, Wires, Mediators and Models can listen to commands to monitor the application workflow using Flash built-in methods:

addEventListener(MyEvent.DO_SOMETHING, handler);

If the events are cancelable, the execution of the commands can be stopped before they occurred using the Flash built-in method:

event.preventDefault();

The Soma Controller intercepts the commands dispatched through the Framework to create the Command Class and call its execute method.

The execution of the commands will not occur if the event has been previously "default prevented".