

# Robotlegs AS3 Micro-Architecture

## Context or SignalContext

- initializes the framework
- provides an event bus
- not limited to a single *Context* (multiple *Contexts* for modular development)

## Actor

- is an abstract class for *Model* and *Service*
- eventDispatcher is injected
- provides a dispatch(event) method
- is for your convenience

## Model [M]

- extends *Actor*
- provide an API for data
- sits between application data and other actors
- should **not** listen for framework events
- dispatch framework events

## Service [S]

- extends *Actor*
- communicate with the outside world and provides an API to external services
- can parse results from external services (foreign data should be converted at the first opportunity)
- do **not** store data (data is stored on a *Model*)
- do **not** receive framework events
- dispatch framework events

## View [V]

- represented by your *View components* and their *Mediators*

## Mediator

- provide API for *View components* (to keep framework out)
- listen for *View components* events
- listen for framework events
- dispatch framework events
- **are** coupled to their *View components*
- can access *Service* and *Model* classes directly (but this couple the *Mediator* to the *Actor*)

## View components


- are **not** coupled to their *Mediators* (or any other framework class, period)

## Controller [C]

- represented by the *Command* class

## Command or SignalCommand

- are executed in response to framework events
- are stateless (they execute and die, performing a single unit of work)
- perform work on *Service* and *Model* classes (and occasionally *Mediators*)
- receive data from the events that trigger them
- dispatch framework events
- do **not** receive framework events (outside the event that triggers them, which is available for injection)

 Framework classes to be extended

 Framework classes to be extended, when using Signals instead of Events

 Framework packages

# [MVC+S]