

Role of Developer }

Advanced Multimedia Technologies

Role of Developer

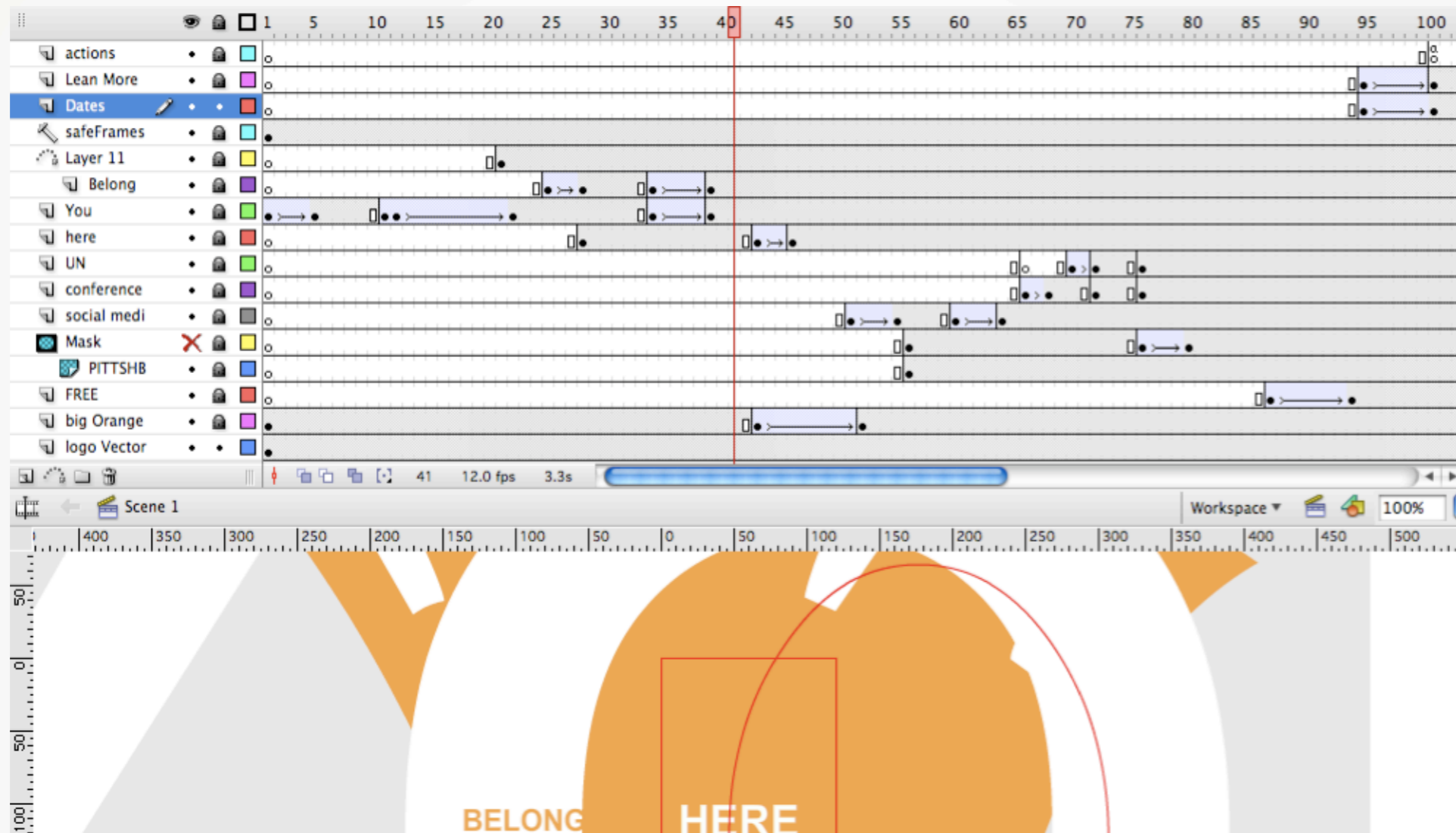
- { What You Know
 - { *One Document*
- { Chunk Out Project
 - { *MCV*
- { Getting Started
- { Reducing Development Time
- { Abstraction



What You Know

One Document

- { .fla files
- { action layer



Adding, Modifying, and
Deleting Content is tedious

One Document Problems

- Takes too long to update
- Searching for sections within the file
- Dangerous in case of loss of data



What's the alternative?

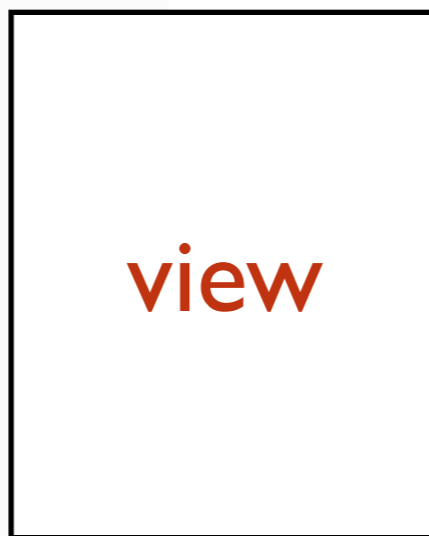
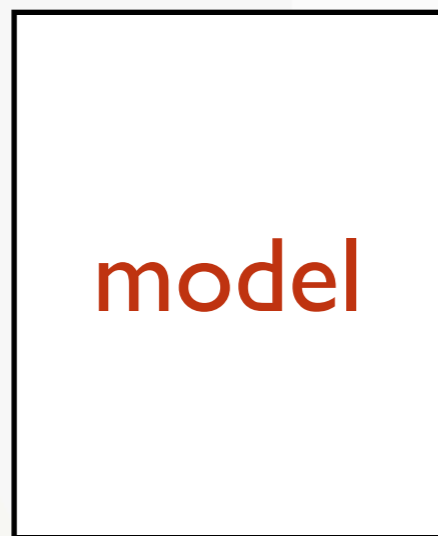


Chunk Out
the Project

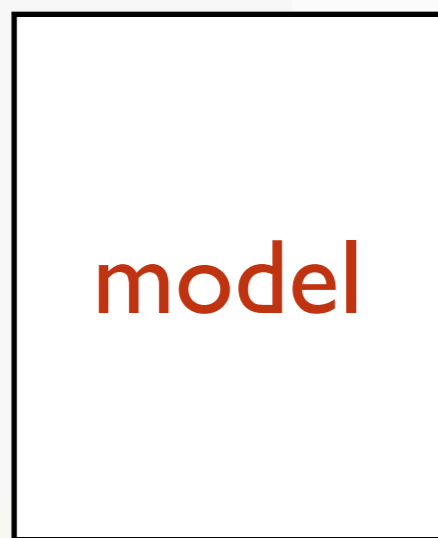
M - V - C

Model View Control

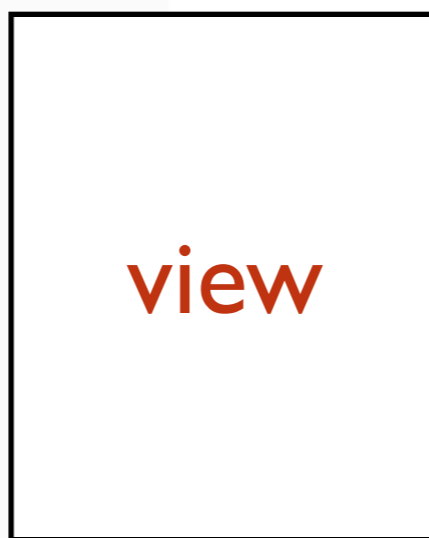
M - V - C



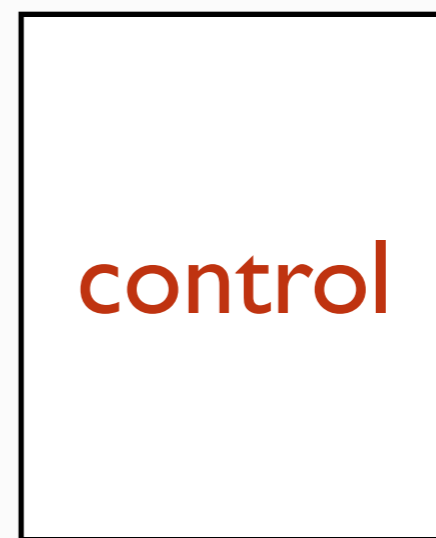
M - V - C



data



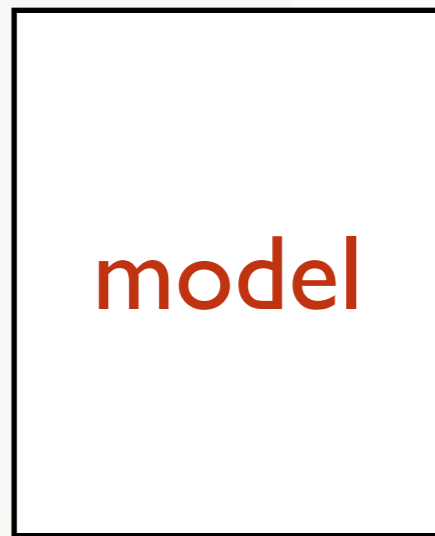
layout



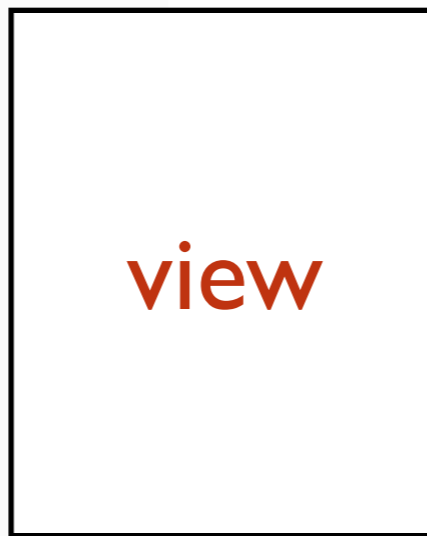
interaction



M - V - C



data



layout



interaction

Each section can be swapped out and it won't interfere with the project

MVC allows for flexibility
in development as
well as maintenance.

This is a design pattern that
you as a developer control.

There aren't any MVC buttons. It's a development philosophy that is carried out throughout the project



Getting Started

Baseline

The basics you need to know in order to get started with higher level ActionScript development.

Baseline

- { Objects
- { Programing Basics
- { Animation

Objects

- { MovieClips are objects
- { Instance names allow communication with through code
- { Text, Sound, Video, and XML are other objects
- { Objects aren't always visual
- { Objects are instances from a class

Objects

- { MovieClips are objects
- { Instance names are created through code
- { Text, Sound, Video objects
- { Objects aren't always visual
- { Objects are instances from a **class**

classes are blueprints of an objects properties, methods, and general purpose

Programming Basics

- { Variables
- { Functions and Arguments
- { trace and comments
- { conditionals
- { arrays
- { loops

Animation

— { Keys

— { Timing

— { Tweens



Reducing Development Time

Goal

Work smarter. Time is money. Many tasks are repetitive. Develop systems to make these tasks more efficient.

Repetitive Tasks

- { Transitions
- { Loading bars
- { Loading / displaying content
- { Screens / page
- { Playing audio and video

Repetitive Tasks

- { Transitions
- { Loading bars
- { Loading / display
- { Screens / page
- { Playing audio and video

Pull this functionality out from other projects. Reuse what you have solved



Don't reinvent the wheel!



Don't solve the same
problems!

Use a proof of concept library

By Abstracting

*We are creating re-usable elements,
templates, code, components etc...*

At this level we are in the
business of **building systems**
not pages, screens etc...

The Difference

Sr. Developer

Systems Creation

Jr. Developer

**Asset and Data
Population**

The Difference

Sr. Developer

Systems Creation

Jr. Developer

**Asset and Data
Population**

**Both must understand each others role
in order for the project to be successful**

The Difference

Sr. Developer

Systems Creation

Jr. Developer

**Asset and Data
Population**

But each role is clearly defined

The Next Level

It's now time to think less about project details and to think more about overarching functionality.



How do you do that?



Abstraction

Abstraction

- { Generalize
- { Categorize
- { Group like concepts
- { Abstract

Classes

The blueprints of functionality

All possibilities are
accounted for in the class

Initialization

*Requires only a few lines vs
hundreds of lines of code.*

Usage

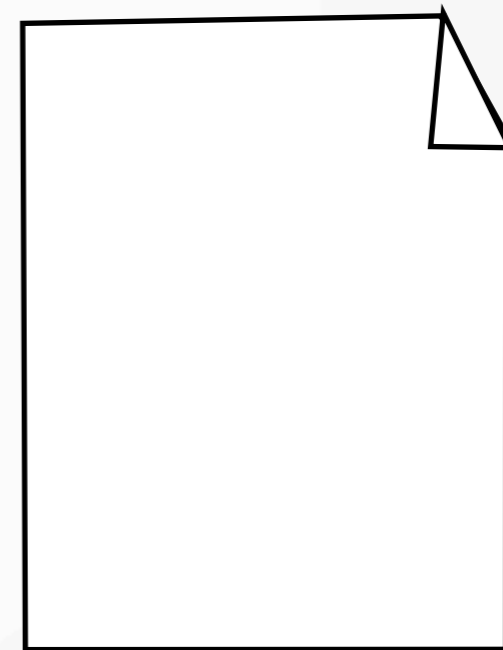
The class can be instantiated as many times as you want, for as many projects as you have.

Case Study

I - Project

75 Hours

- { 20 Screens with transitions
- { Sound track and sound interactions
- { Pre-loader
- { Navigation

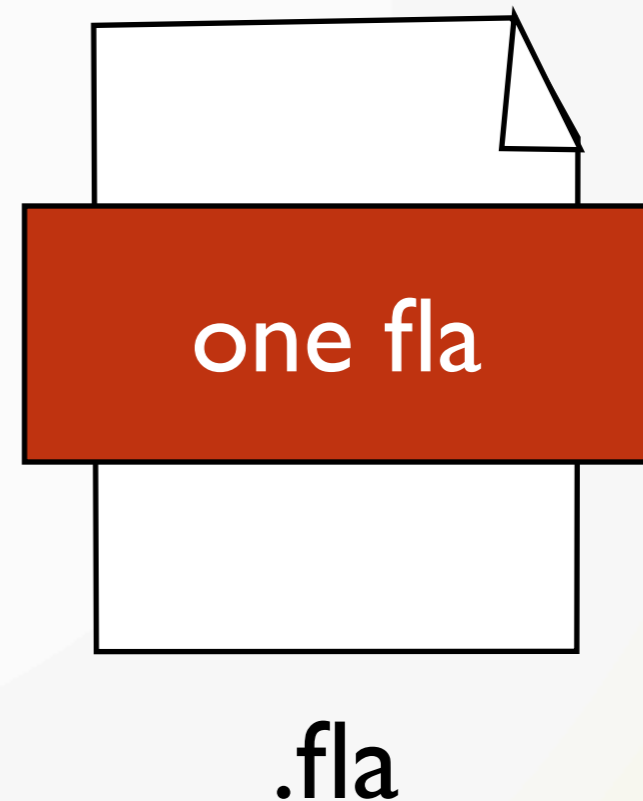


.fla

I - Project

- { 20 Screens with transitions
- { Sound track and sound interactions
- { Pre-loader
- { Navigation

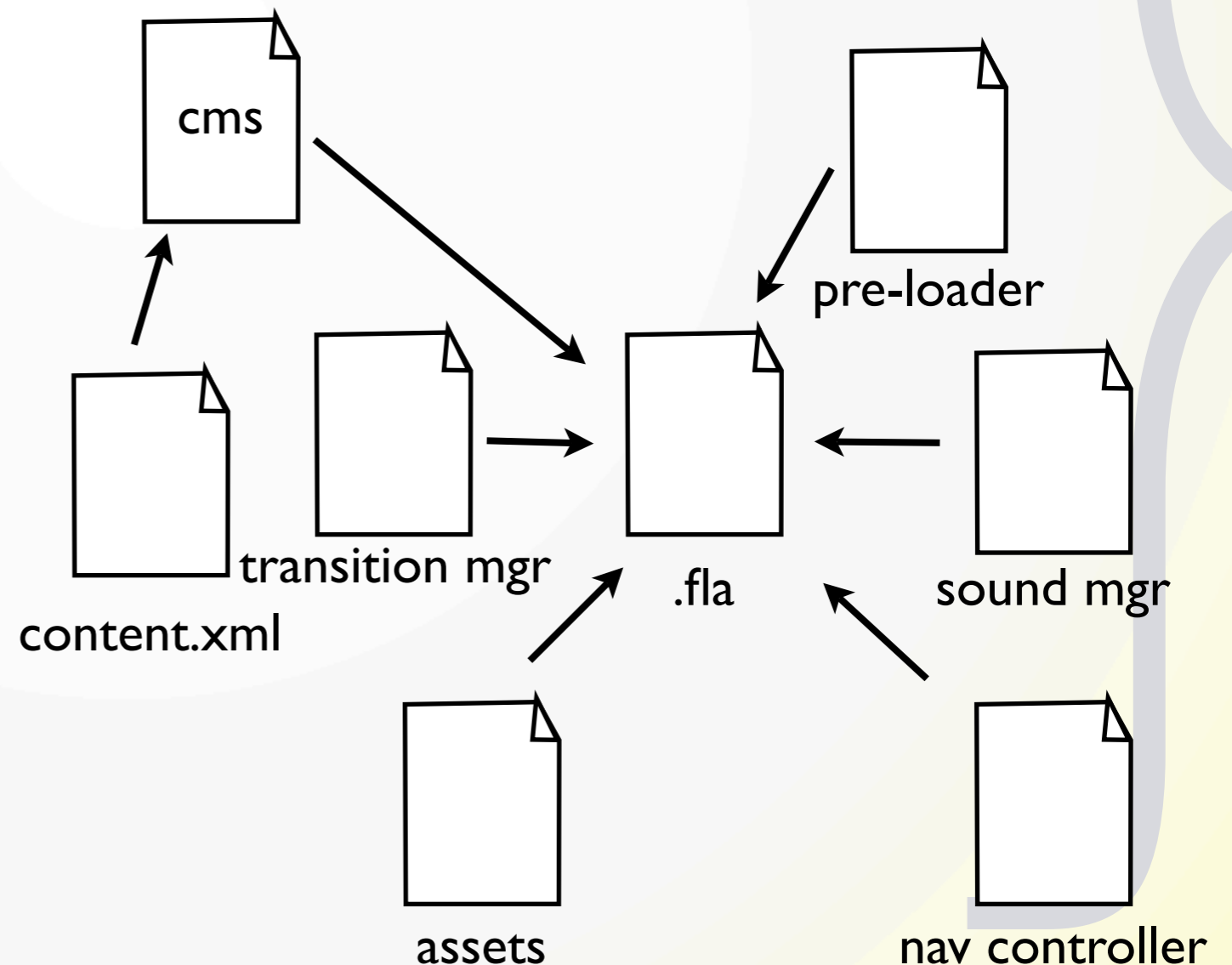
75 Hours



I - Project

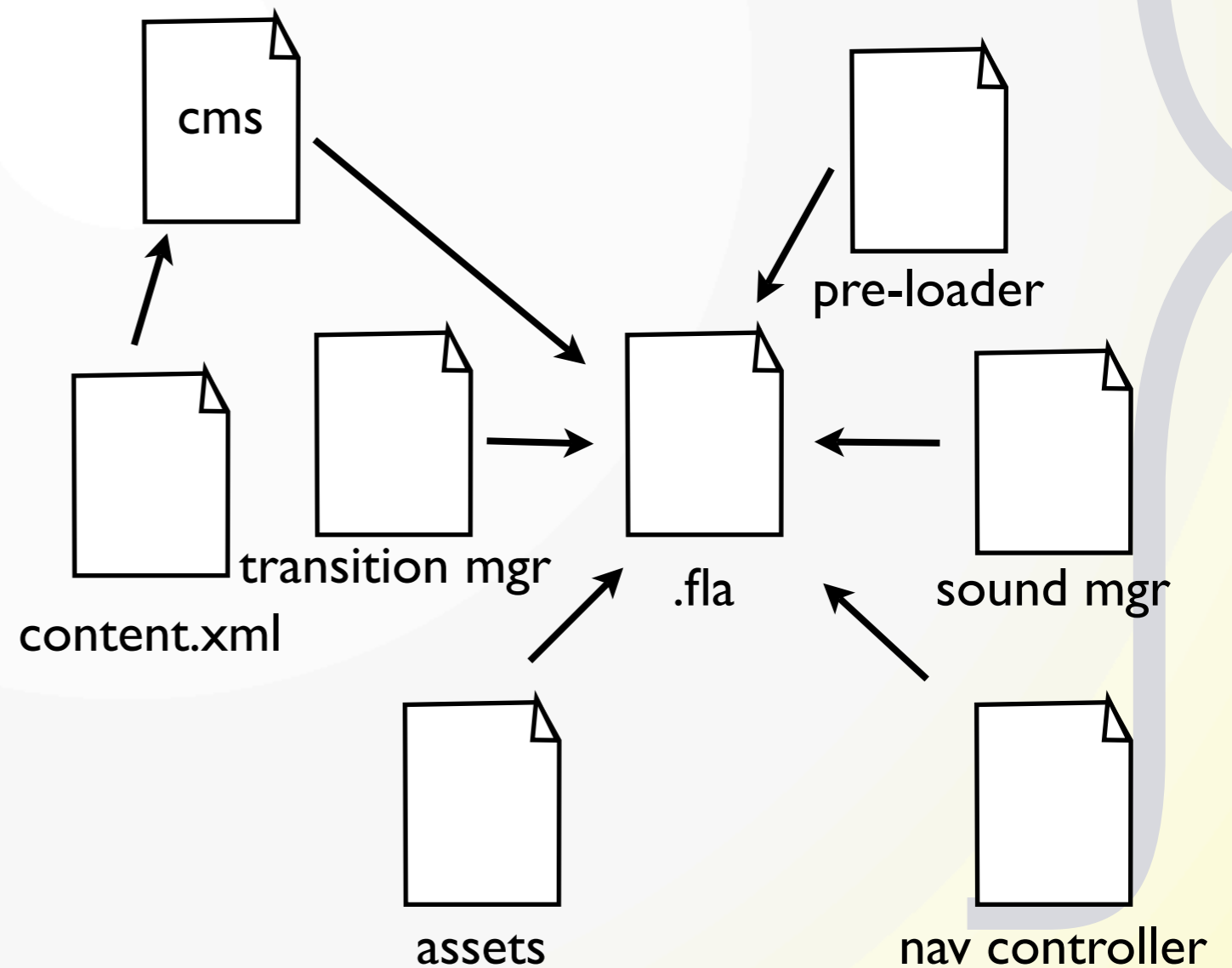
- Navigation controller
- Content Management System
- Transition Mgr
- Sound Mgr

40 Hours



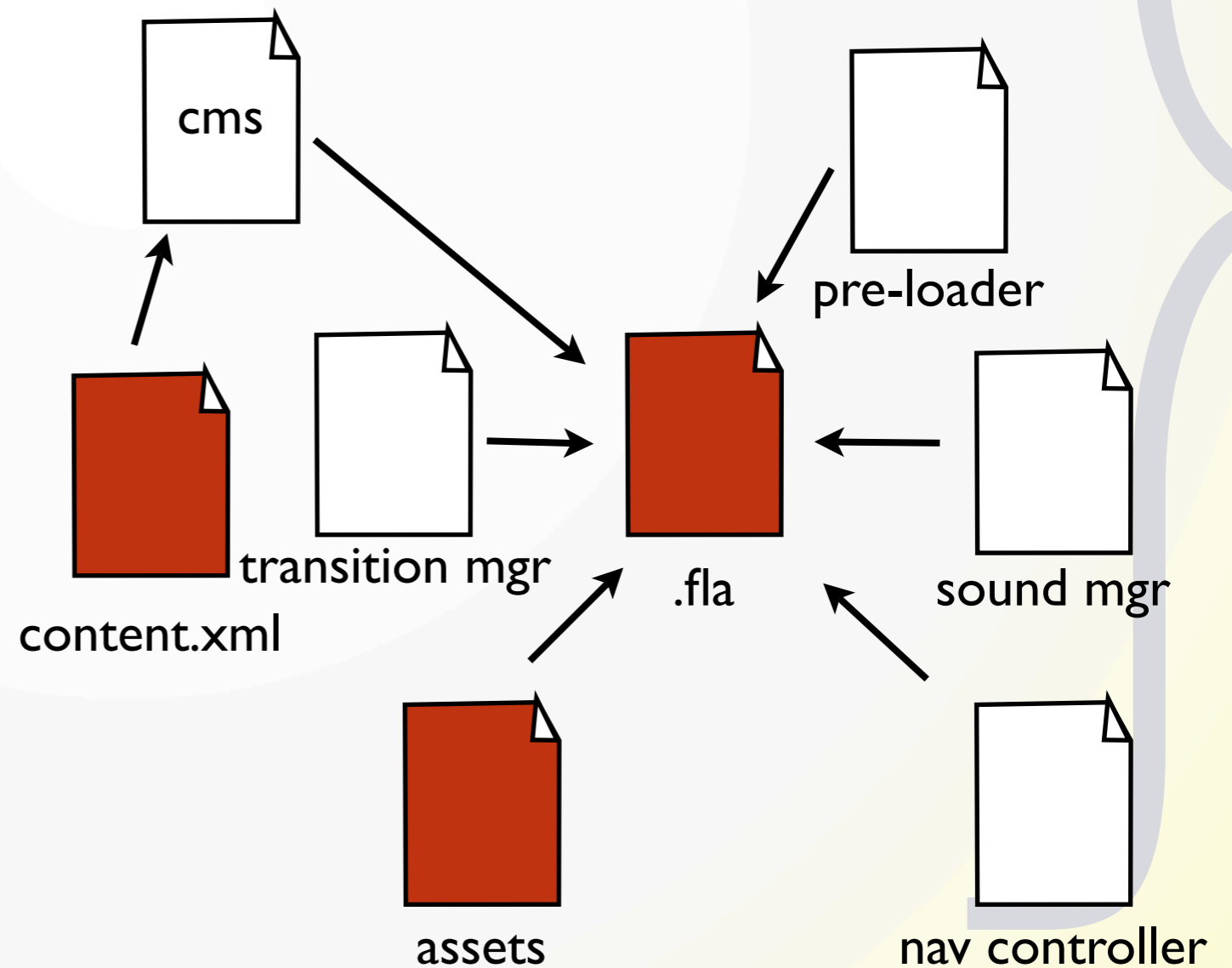
Projects 2-3-4-5-6-7-8-9

- Navigation controller
- Content Management System
- Transition Mgr
- Sound Mgr



Projects 2-3-4-5-6-7-8-9

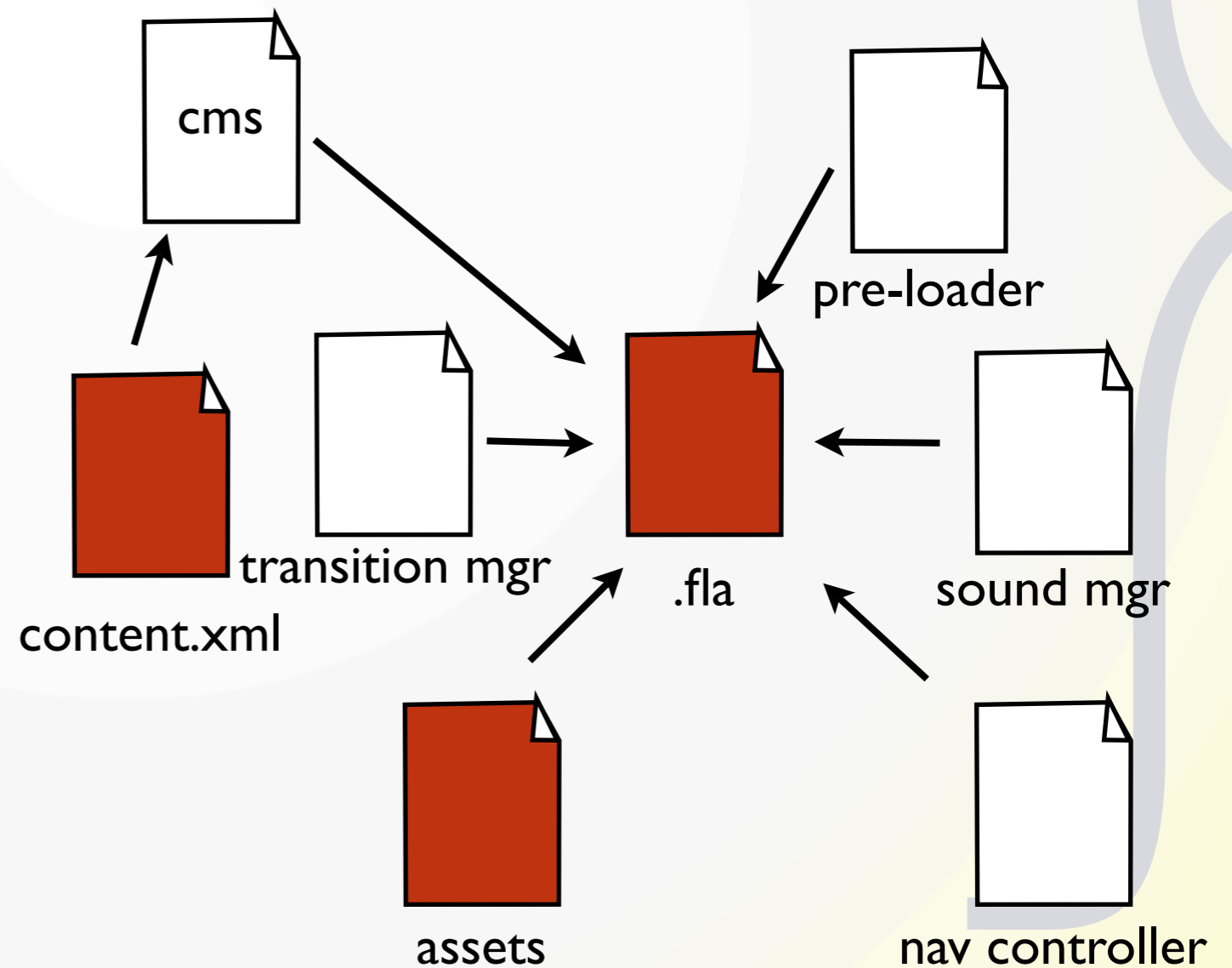
— { Change only what's different



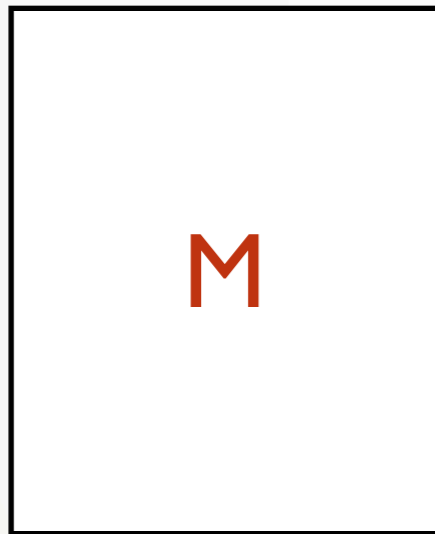
Projects 2-3-4-5-6-7-8-9

— { Change only
what's different

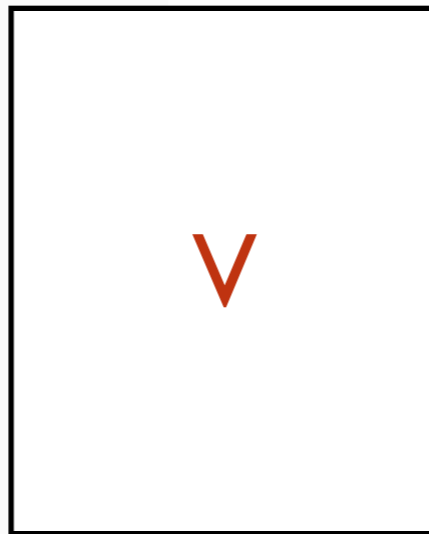
**10 Hours
per project**



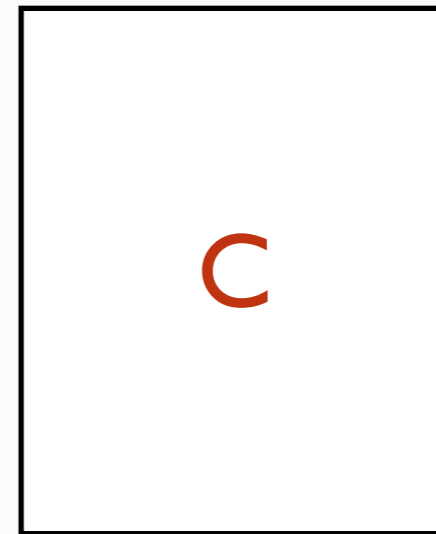
Abstract and Categorize



data



layout

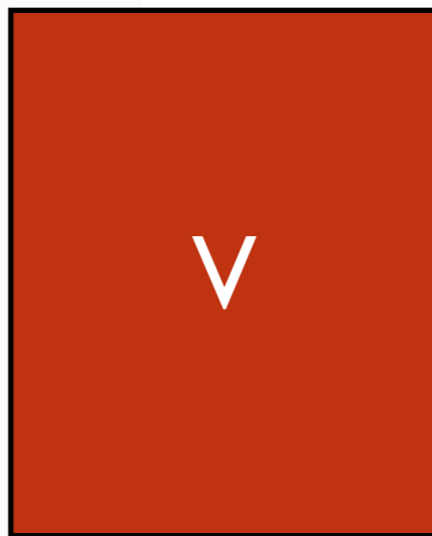


interaction

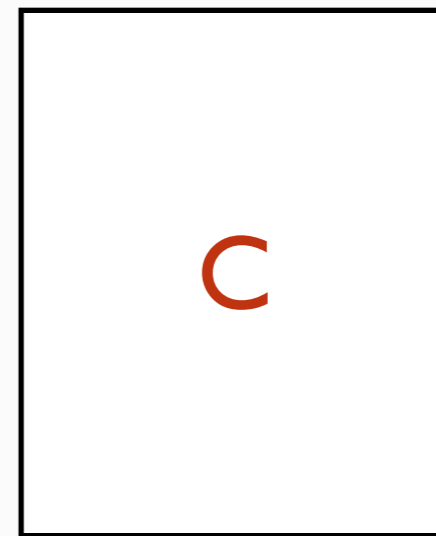
Abstract and Categorize



data



layout



interaction

Swap these out

You've got a **framework** to
reference for all future projects

The ground floor towards
harnessing these strategies is
class development.

Summary: Role of Developer

- { What You Know
 - { *One Document*
- { Chunk Out Project
 - { *MCV*
- { Getting Started
- { Reducing Development Time
- { Abstraction