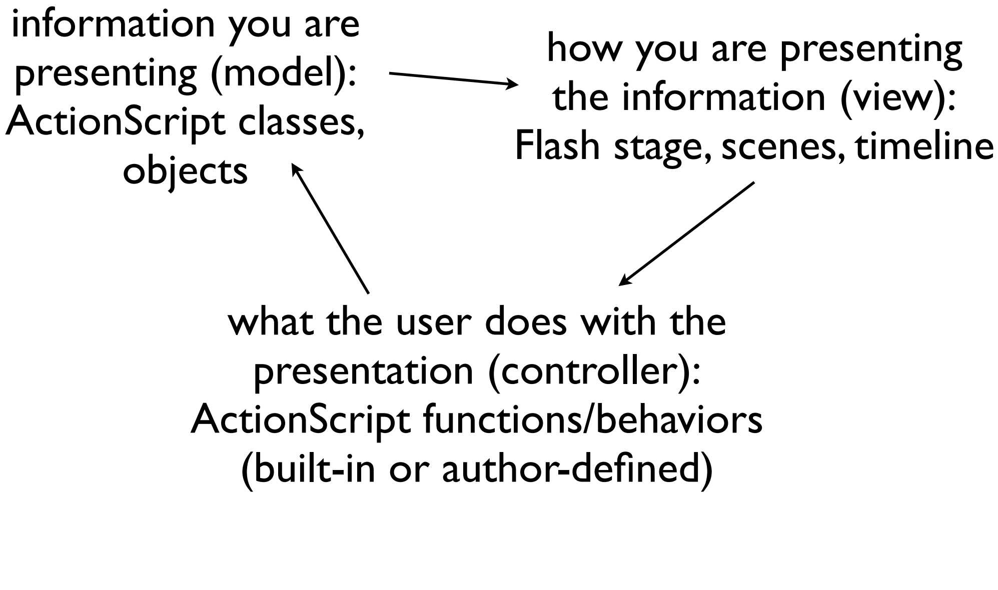
Accomplishing Real MVC in Flash

A review of Flash and MVC:

- Flash’s scene-oriented, stage/design approach blurs the line between what is content, presentation, and behavior
- Model/view tends to be indistinguishable, with controller being better separated thanks to ActionScript

ActionScript 2.0 Enables True MVC Separation



information you are presenting (model): ActionScript classes, objects

how you are presenting the information (view): Flash stage, scenes, timeline

what the user does with the presentation (controller): ActionScript functions/behaviors (built-in or author-defined)

From Your Brain to ActionScript Classes

What information is displayed by your application? What details or properties do these items possess?

- "I am displaying trivia questions that can have an image or audio file, with a multiple-choice answer"
- "My application is a movie collection with posters, a trailer, and cast/crew information"
- "I am displaying a catalog of my artwork; each work has a title, date, and image"

- The nouns in your application translate to ActionScript classes
- The properties of these nouns translate to variables within these classes

1. Determine a name for each class, and create a text file called (classname).AS in the same folder as the .FLA file
2. Define the class; its major components are its variables, constructor(s), and methods
3. Create *instances* of the class in your Flash document (.FLA file)
4. Create functions in your Flash document that assign these instances to the components and symbols that are on the Flash stage

Example Model: Family Tree in Manila 2004 Application

"I want to display a family tree, showing information about a person and allowing the user to see that person's ancestors and descendants."

- So the key class here is a Person; that Person will have information specific to that individual (name, birthdate, portrait, among others)
- Each Person object should also somehow lead to ancestors and descendants

```
class Person {
    // A Person object has a name, a list of parents, and a list of children.
    // Note that this is only a starter example --- in full bloom, you may want
    // to add more information such as birthdates, pictures, other relatives, etc.
    private var name: String;
    private var parents: Array;
    private var children: Array;

    // This is a "constructor" --- it "creates" a new Person.
    function Person(name: String) {
        this.name = name;
        this.parents = null;
        this.children = null;
    }

    // These are "methods" --- they define activities that you can "request" from a person.
    function toString(): String { return getName(); }

    // "Getter/setter methods" are a special category of method --- they provide ("get")
    // or modify ("set") the properties/attributes of the object.
    function getName(): String { return this.name; }
    function getParents(): Array { return this.parents; }
    function getChildren(): Array { return this.children; }

    function setName(newName: String): Void { this.name = newName; }
    function setParents(newParents: Array): Void { this.parents = newParents; }
    function setChildren(newChildren: Array): Void { this.children = newChildren; }
}
```

Next, Define the Data

- Somewhere in the .FLA file, write some ActionScript that creates the set of person objects to be displayed
- In this example, we are creating the data directly in the script; in other applications, you potentially read this information from a separate file or a database

```
// Set up our "database" of people.
var dondi = new Person("Dondi Dionisio");
var meilyn = new Person("Mei Lyn Dionisio");
var aidan = new Person("Aidan Dionisio");
var anton = new Person("Anton Dionisio");

var parents = [ dondi, meilyn ];
var children = [ aidan, anton ];

dondi.setChildren(children);
meilyn.setChildren(children);
aidan.setParents(parents);
anton.setParents(parents);
```

Typical place for this would be the Actions layer of the first frame of your timeline, or the first frame of the scene or movie clip that uses this information…

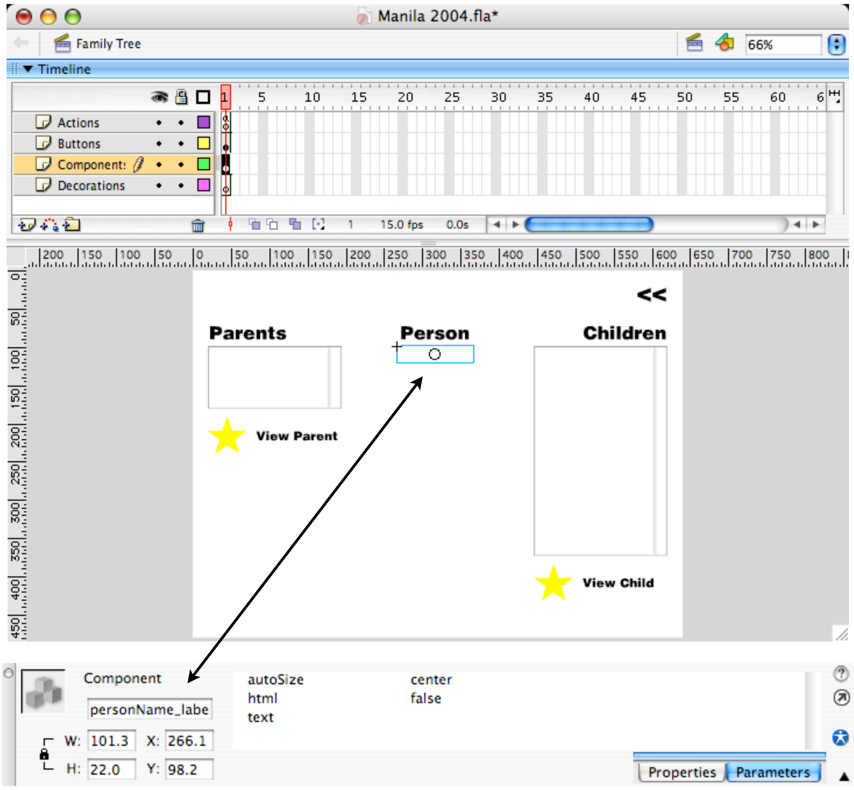
If we do things right, adding more data to your Flash application will merely entail changes to this section of your ActionScript.

Set Up Your Stage (View)

- Decide what components (labels, lists, date choosers — Flash has quite a few built in) will display what properties of your data
- Put these components on the Flash stage
- Assign instance names to these components

Set Up Controller Elements

- Frequently buttons, but may consist of other components (sliders, date choosers, or anything else that accepts user input or manipulation)
- Assign instance names to the controller elements, too
- It helps to have a spec here, so you know what gets connected to where



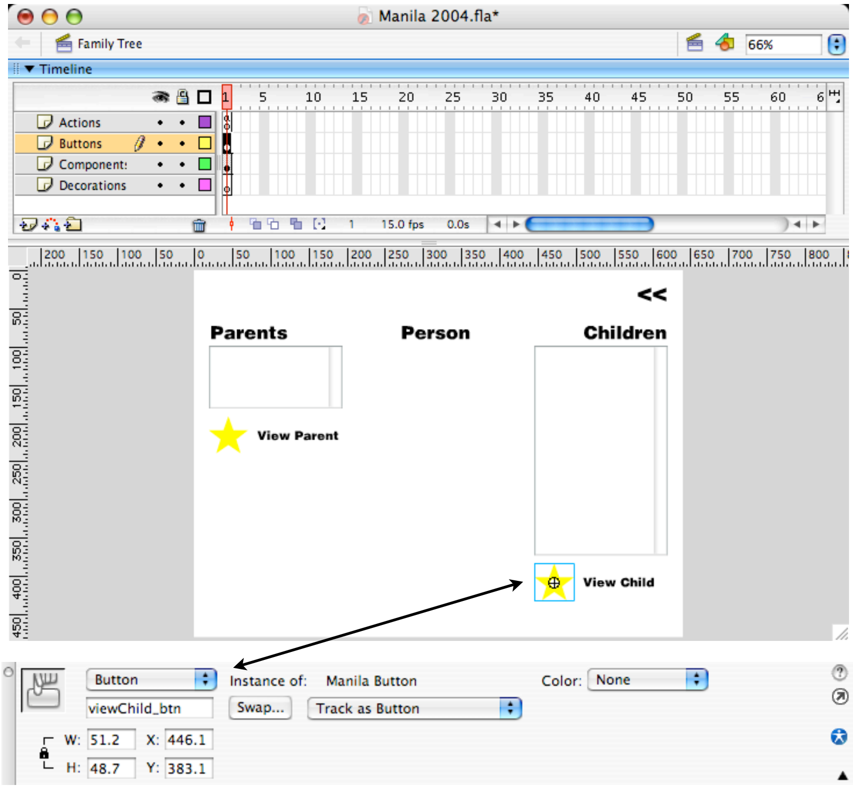
Component

personName_labe

autoSize	center
html	false
text	

W: 101.3 X: 266.1

H: 22.0 Y: 98.2



Button

viewChild_btn

Instance of: Manila Button

Color: None

Track as Button

W: 51.2 X: 446.1

H: 48.7 Y: 383.1

Connect the M to V, then V to C, then C to M

This is typically ActionScript in the following forms:

- Helper functions that take an object and makes sure that view elements are showing information from that object (model to view)
- Functions assigned to components (such as button *onRelease*), which describe what will happen when user interacts with these components — the functions themselves are controller-to-model, while assigning the functions to the buttons comprise view-to-controller

```
// Define a function that sets the current person.
function setPerson(newPerson: Person) {
    personName_label.text = newPerson.getName();
    parents_list.dataProvider = newPerson.getParents();
    children_list.dataProvider = newPerson.getChildren();
}

/*
   Set up button behavior.
*/
// These are new --- these buttons just move from person to person.
viewParent_btn.onRelease = function() {
    if (parents_list.selectedItem != null) {
        setPerson(parents_list.selectedItem);
    }
};

viewChild_btn.onRelease = function() {
    if (children_list.selectedItem != null) {
        setPerson(children_list.selectedItem);
    }
};

// Set up the lists to display the right information about
// the objects that they are displaying.
function getPersonListLabel(p: Person): String {
    return p.getName();
}

parents_list.labelFunction = getPersonListLabel;
children_list.labelFunction = getPersonListLabel;

// Finally, set the "first" person, then stop the scene.
setPerson(dondi);
stop();
```

These parts make specific use of Flash's Label and List components — learning about them is a cycle of looking them up, testing, or looking for examples on the Internet. And there's always the venerable short cut — find someone who knows and ask them directly.

Scripting Survival Tips

Get comfortable with looking things up! It's nearly impossible to keep an entire scripting language and its libraries in your head

- ActionScript Language Reference in Flash Help
- Reference section in upper-left corner of Actions panel

Learn how to "learn by example" — look at other people's work (Macromedia has tutorials of their own, and many Flash folks post their source documents)

If things don't work the way you expect, learn how to identify the kind of problem that is occurring, and where/when the problem takes place

- Is the script written correctly?
- Are you using the correct instance names?
- If Flash reports an error, what kind of error is it? If Flash says that the error occurred at a specific line in your script, look before and after that line as well — sometimes, the true error is not exactly at the line that Flash identifies.