Chapter 3

***Listing 3-1.*** A JavaScript Implementation of the Singleton Pattern var Logger = (function() {

// private variable to hold the only

// instance of Logger that will exist

var loggerInstance;

// Create the logger instance var createLogger = function() { var \_logWarning = function(message) {

// some complex work coud go here, but

// let's just fake it

return message.toUpperCase();

};

return {

logWarning: \_logWarning

};

};

return {

// Here is the crucial part. First we check

// to see if an instance already exists. If

// it does, we return it. If it does not, we // create it.

getInstance: function() { if (!loggerInstance) {

loggerInstance = createLogger();

}

return loggerInstance;

}

};

})();

***Listing 3-2.*** MVC in Action

function MyFirstCtrl($scope) {

// populate the employees variable with some model data

var employees = ['Christopher Grant', 'Monica Grant', 'Christopher Grant', 'Jennifer

Grant'];

// Now put this model data into the scope so it can be used in the view

$scope.ourEmployees = employees;

}

***Listing 3-3.*** A Complete MVC Example

<!DOCTYPE html>

<html ng-app>

<head>

<script src="js/angular.min.js"></script>

<script>

function MyFirstCtrl($scope) {

var employees = ['Catherine Grant', 'Monica Grant',

'Christopher Grant', 'Jennifer Grant'

];

$scope.ourEmployees = employees;

}

</script>

</head>

<body ng-controller='MyFirstCtrl'>

<h2>Number of Employees: {{ ourEmployees.length}}</h2>

</body>

</html>

***Listing 3-4.*** Displaying the Employee Names

<!DOCTYPE html>

<html ng-app>

<head>

<script src="js/angular.min.js"></script>

<script>

function MyFirstCtrl($scope) {

var employees = ['Catherine Grant', 'Monica Grant',

'Christopher Grant', 'Jennifer Grant'

];

$scope.ourEmployees = employees;

}

</script>

</head>

<body ng-controller='MyFirstCtrl'>

<h2>Number of Employees: {{ ourEmployees.length}}</h2> <p ng-repeat="employee in ourEmployees">{{employee}}</p>

</body>

</html>