The Model Layer

What Is the Model?

In an MVC application, the Model layer is typically the largest and most important piece. The Model is designed to house the business logic and data access code; in other words, the Model consists of the core of the application. For example, if an application computes the average sell-through rate of a product, the Model layer performs that computation. For an application that maintains a database of employees, complete with salary and tax information, the Model handles the maintenance task. Thus, it is the Model that defines what the application does. The View and Controller interact with the Model and provide a user interface to it.

The MVC architecture dictates that the Model layer should be self contained and function independently from the View and Control layers. That way, the core application code can be used over and over again with multiple user interfaces. For example, you could have a Web interface for the application as well as a stand-alone or wireless interface. Each interface (Web, stand-alone, and so on) would have its own code for the user interface (View), but would reuse the core application (Model) code. This is the basis for the MVC architecture: having a clean separation of responsibilities and reducing coupling between application layers.

Model Layer Breakdown

The typical Model layer of a correctly designed MVC application can be broken down into three conceptual sublayers. Each sublayer can be thought of as a component or responsibility of the Model. Figure 3-1 illustrates this breakdown.

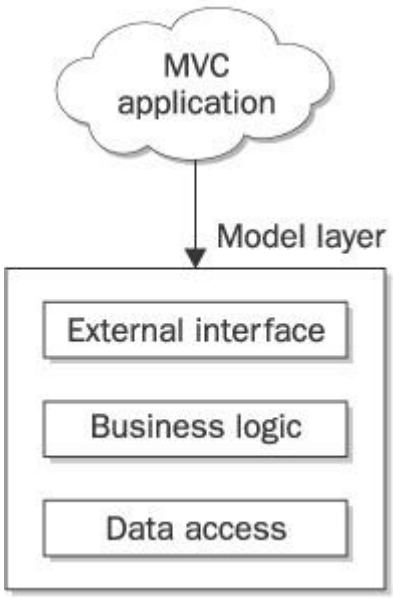

Figure 3-1: Model layer breakdown

Each sublayer does not necessarily represent a separate set of classes, but rather the Model's set of responsibilities. You may choose to house a specific function's code for all layers in one large class, or you may break down each sublayer into fine-grained objects. The level of object granularity is up to you and what's best and/or necessary really depends on the size and complexity of your application. The following are the three sublayers:

- **External interface** Composed of code that provides an interface that external code uses to interact with the Model.
- **Business logic** Encompasses the bulk of the Model code and provides the business functionality for an application.
- **Data access** Composed of code for communicating with an application's data sources such as a database.