|  |  |
| --- | --- |
|  |  |

BÁO CÁO MÔN HỌC

**LẬP TRÌNH WEB**

Đề tài:

**TÌM HIỂU JAVASCRIP FRAMEWORK**

Ngành: **CÔNG NGHỆ THÔNG TIN**

Giảng viên hướng dẫn :

Sinh viên thực hiện:

Nguyễn Văn Sĩ MSSV: 1911066559Lớp:19DTHE2

Hồ Sỹ Chiến MSSV: 1911066000Lớp:19DTHE2

Trần Trung Thuật MSSV: 1911066000Lớp:19DTHE2

Phùng Mộng Thảo Quyên MSSV: 1911066000Lớp:19DTHE2

# Phần 1: FRAMEWORK

1. Những thông tin cơ bản cần biết về framework là gì?
2. Framework là gì?

Trong việc lập trình ứng dụng, và bên cạnh đó là lập trình web sẽ luôn luôn xuất hiện rất nhiều vấn đề mà chúng ta gặp phải hết lần này đến lần khác, kể cả ở các công ty khác thì ai cũng gặp phải, khi xử lý cùng một vấn đề thì có cách nào để tiết kiệm thời gian và công sức không? Để cung cấp một môi trường trợ giúp cho việc lập trình ứng dụng nhanh chóng thì thuật ngữ Framework ra đời.

*“Framework là các đoạn code đã được viết sẵn, cấu thành nên một bộ khung và các thư viện lập trình được đóng gói. Chúng cung cấp các tính năng có sẵn như mô hình, API và các yếu tố khác để tối giản cho việc phát triển các ứng dụng web phong phú, năng động. Các framework giống như là chúng ta có khung nhà được làm sẵn nền móng cơ bản, bạn chỉ cần vào xây dựng và nội thất theo ý mình.”*



Hình 1.1.1.1: Những thông tin cơ bản cần biết về framework là gì

1. Tính năng chính của framework là gì

* *Giảm / loại bỏ các công việc lặp đi lặp lại.*
* *Tăng tính linh hoạt của ứng dụng thông qua tính trừu tượng.*
* *Sử dụng lại mã.*
* *Đơn giản hóa việc chúng ta sử dụng các công nghệ mới và quá phức tạp.*
* *Framework là gì kết nối một loạt các đối tượng / thành phần riêng lẻ thành một hệ thống hữu ích hơn.*
* *Bất kỳ ai cũng có thể dễ dàng kiểm tra và gỡ lỗi mã, ngay cả khi họ không tham gia vào dự án.*
* *Quy trình khép kín về thiết kế giao diện, mã và kiểm thử phần mềm.*

1. Cách thức hoạt động của các Framework là gì

Framework là gì được thiết kế để tiết kiệm thời gian và hỗ trợ phím tắt. Thực sự có thể tạo một trang web rất đơn giản hoặc ứng dụng web hiện đại mà không cần nó. Framework là gì tối ưu hóa quá trình phát triển và cho phép sử dụng, sửa đổi và tích hợp mã để đơn giản hóa công việc. Tất cả các lập trình viên chuyên nghiệp đều biết cách sử dụng một hoặc nhiều Framework là gì khác nhau.

Nói theo thống kê, sử dụng Framework là gì có thể hoàn thành nhiều chương trình hơn trong thời gian ngắn hơn, nhưng chất lượng cao hơn nhiều. Đó là bởi vì ngay cả một lập trình viên thiếu kinh nghiệm sử dụng một Framework là gì cũng có thể dễ dàng tích hợp những đoạn mã tuyệt vời vào chương trình của mình, và thông thường đây là những gì lập trình viên làm, bởi vì đã có rất nhiều mã với quy trình hoàn hảo. Tất cả những gì còn lại là tích hợp nó.

Giả sử dự án của bạn yêu cầu quy trình A + quy trình B + quy trình C. Các nhà phát triển tìm mã tốt nhất cho mỗi quy trình và hợp nhất chúng để hoạt động trong một khuôn khổ duy nhất.

Không giống như các thư viện, các Framework là gì cho phép đảo ngược điều khiển mã. Do đó, nếu bạn cần xác định các quy tắc hoặc phương pháp làm việc, hãy giả sử nó là một công cụ để xây dựng các ứng dụng.

1. Ưu điểm của Framework là gì và nhược điểm của Framework là gì?
2. Ưu điểm của Framework là gì?

Framework là gì cung cấp nhiều lợi thế, chẳng hạn như:

* *Framework có các chức năng tích hợp chung cho các ứng dụng / phần mềm. Ví dụ: hầu hết tất cả các trang web thương mại điện tử đều yêu cầu đăng ký, đăng nhập, quản lý hồ sơ người dùng, v.v*
* *Giúp lập trình viên tiết kiệm thời gian và sức lực khi phát triển phần mềm / ứng dụng.*
* *Cho phép các sản phẩm ứng dụng kế thừa các tính năng và cấu trúc đã được tiêu chuẩn hóa. Điều này làm cho việc vận hành ứng dụng và bảo trì/khắc phục sự cố dễ dàng hơn.*
* *Cho phép người dùng mở rộng tùy ý theo nội dung mà framework cung cấp. Miễn là các tiêu chuẩn nhất định được tuân thủ, các nhà phát triển có thể mở rộng chức năng bằng cách ghi đè có chọn lọc các lớp hiện có hoặc viết các tính năng mới lên đầu khung.*

1. Nhược điểm của Framework là gì?

* *Phải mất rất nhiều thời gian và năng lượng để học cách làm chủ Framework là gì.*
* *Khi sử dụng Framework là gì, dung lượng của ứng dụng / phần mềm sẽ rất lớn. Trong nhiều trường hợp, ngay cả khi một trang web không chứa bất kỳ nội dung nào, kích thước mã của nó có thể lên tới hàng trăm MB.*
* *Khi lập trình viên viết mã, họ cần tuân thủ nghiêm ngặt các quy tắc do Framework là gì đặt ra.*
* *Các Framework là gì thường lớn, vì vậy chúng không thích hợp để phát triển các ứng dụng quá nhỏ. Ví dụ, bạn muốn xây dựng một ứng dụng nhỏ cập nhật thông tin của Festival Hoa Đà Lạt năm 2022 thì không cần sử dụng Framework. Ngược lại, viết mã của riêng bạn sẽ tiết kiệm rất nhiều thời gian.*

# Phần 2: ENTITY FRAMEWORK

1. Entity Framework là gì?

Entity Framework là một ánh xạ quan hệ đối tượng (ORM), là một loại công cụ giúp đơn giản hóa ánh xạ giữa các đối tượng trong phần mềm của bạn với các bảng và cột của cơ sở dữ liệu quan hệ.

* Entity Framework (EF) là một framework ORM mã nguồn mở độc lập với .NET Framework.
* Entity Framework đảm nhiệm việc tạo các kết nối cơ sở dữ liệu và thực thi các lệnh, cũng như lấy kết quả truy vấn và tự động ánh xạ các kết quả đó thành các đối tượng trong ứng dụng của bạn.
* Entity Framework giúp theo dõi các thay đổi của đối tượng và cập nhật các thay đổi đó trở lại cơ sở dữ liệu cho bạn.

1. Sơ lược về lịch sử Entity Framework

Entity Framework được Microsoft chính thức phát triển từ năm 2008, tính đến nay cũng được trên 12 năm và hiện nay thì Entity Framework đã được Microsoft khuyến nghị sử dụng. Ra đời cùng với .NET 3.5 và phiên bản mới nhất là 6.0.

* Đối với phiên bản Entity Framework đầu thì nó chỉ hỗ trợ người dùng tiếp cận Database first. Tức là EF khi đó chỉ có thể làm việc cùng với một cơ sở dữ liệu đã có sẵn và nó giúp sinh ra các lớp code, model bằng C# hoặc là VB.NET.
* Phiên bản Entity Framework 4 xuất hiện hướng tiếp cận Model first, giúp cho người dùng thiết kế các lớp thực tế trước sử dụng giao diện đồ họa.
* Đến phiên bản 4.1 thì đã đưa thêm được hướng tiếp cận của code first. Tức là mọi thứ đều có thể viết bằng code VB.NET hoặc là C# từ đầu. Có thể nói hướng tiếp cận này vô cùng được nhiều lập trình viên hứng thú và sử dụng một cách rộng rãi.

1. Đặc điểm của Entity Framework

Để hiểu rõ hơn Entity Framework thì bạn cần phải tìm hiểu về đặc điểm của nó như thế nào, sau đó mới có thể áp dụng linh hoạt trong công việc. Entity Framework sẽ có 3 đặc điểm nổi bật như sau:

* Thứ nhất là tất cả cơ sở dữ liệu đều được thể hiện bằng một lớp con của DBContext. Mỗi bảng dữ liệu sẽ được thể hiện bằng một object của Dbset. Mỗi hàng trong bảng đó sẽ thể hiện một object của lớp thực tế mà bạn đã xây dựng. Còn mỗi cột sẽ được thể hiện bằng thuộc tính của object. Đối với những thao tác này thì hoàn toàn được thực hiện một cách tự động, hoặc là cũng có thể can thiệp vào chúng theo ý của mình.
* Thứ hai chính là bạn có thể sử dụng để truy vấn dữ liệu sử dụng LINQ mà không phải sử dụng SQL. Người dùng có thể đơn giản hơn khi thực hiện các truy vấn CRUD từ code C# với các class mà không cần phải viết bất kỳ một dòng SQL nào.
* Thứ ba chính là có thể thực hiện thao tác với các cấu trúc dữ liệu như: Tạo ra cơ sở dữ liệu, đổi cấu trúc bảng, tạo bảng,.. một cách đơn giản và nhanh chóng với công cụ Migration mà cũng sẽ không mất dữ liệu.

Đó chính là 3 đặc điểm nổi bật, cơ bản nhất mà một người lập trình viên khi sử dụng. Entity Framework cần phải nhớ rõ. Khi nhớ được những đặc điểm này của nó thì quá trình sử dụng, thay thế khi lập trình cũng được đơn giản hóa hơn.

1. Lợi ích khi sử dụng Entity Framework

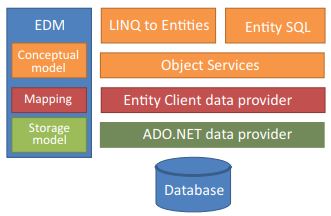
Entity Framework được Microsoft phát triển và khuyến nghị sử dụng, bên cạnh đó thì nó cũng được rất nhiều lập trình viên ưa chuộng sử dụng. Chắc chắn Entity Framework sẽ phải có những lợi ích tích cực thì mới được ưa chuộng đến như vậy, trong phần này chúng ta cùng tìm hiểu về lợi ích khi sử dụng Entity Framework nhé!

Ưu điểm, lợi ích của Entity Framework bao gồm như sau:

* Entity Framework sẽ giúp cho người sử dụng có thể tăng năng suất thông qua giảm số lượng code của mình. Trên thực tế đã cho thấy có đến khoảng 35% code của các chương trình thông thường là sẽ làm việc cùng với cơ sở dữ liệu. Tuy nhiên nếu như bạn sử dụng một Entity Framework chất lượng thì nó có thể giúp bạn giảm con số này xuống khoảng 15-20%, cũng có những trường hợp giảm chỉ còn 5%. Đây là con số không hề nhỏ chút nào, nó giúp cho người sử dụng tiết kiệm được nhiều thời gian, công sức và chất xám của mình trong quá trình làm việc. Thời gian tiết kiệm đó sẽ giúp họ làm được nhiều các công việc khác.
* Entity Framework làm giảm sự phức tạp hóa khi cần bảo trì phần mềm. Bạn có thể so sánh đơn giản, việc bảo trì 400 dòng code và bảo trì khoảng 1000 dòng code thì cái nào sẽ đơn giản hơn. Khi số lượng code ít thì bảo trì, kiểm tra lại cũng sẽ đơn giản hơn. Đặc biệt là khi bạn phải bảo trì thường xuyên, dài hạn. Đối với bảo trì có thể thực hiện ở trên phía cơ sở dữ liệu và ứng dụng mà không khiến người dùng phải code lại.
* Entity Framework sẽ được đặt trong .NET Framewwork, được tích hợp trong cả Visual studio. Yếu tố này sẽ giúp cho người dùng làm việc với EF đơn giản hơn và tiện lợi hơn nhiều. Có thể đây sẽ là lý do chính khiến cho nhiều người từ sử dụng Nhibernata mà sử dụng thành EF. Còn một ORM cũng khá phổ biến và dễ sử dụng cho .NET. Entity frameword, nó cũng là ORM mà được nhà phát triển khuyến nghị sử dụng. Cũng có rất nhiều các công ty lớn đầu tư vào ORM này.

Đó chính là những lợi ích mà một Entity Framework đem lại. Thế nhưng bên cạnh những lợi ích đó thì Entity Framework cũng đã gây rất nhiều tranh cãi về mặt hạn chế. Không có công cụ nào ra đời mà lại hoàn hảo ở mức tuyệt đối, đôi khi nó cũng có những điểm hạn chế của mình. Đối với Entity Framework cũng vậy, khi sử dụng nó thì sẽ gây ra nhược điểm hiệu suất khá lớn. Bởi vì ORM phức tạp, khi sử dụng sẽ dẫn đến giảm hiệu suất hơn khi sử dụng ADO.NET. Song, hạn một hạn chế này sẽ khiến cho nhiều lập trình viên phải so sánh với những lợi ích mà nó đem lại cho mình trong quá trình sử dụng.

1. Kiến trúc của một Entity Framework



*Hình 2.5.1:Kiến trúc của một Entity Framework*

Như trong sơ đồ trên bạn thấy Entity Framework phân chia làm các thành phần: Entity Data Model (EDM), Entity Client data provider, Object Services, LINQ to Entities và Entity SQL.

1. Entity Data Model (EDM)

EDM là thành phần lưu trữ thông tin ánh xạ (mapping) giữa các class và cơ sở dữ liệu. Nó đóng vai trò cầu nối giữa các lớp model và cơ sở dữ liệu nhưng lại giúp chương trình hoàn toàn tách biệt với cơ sở dữ liệu.

Cơ sở dữ liệu và các lớp model của chương trình là hai thứ khác nhau hoàn toàn về bảnchất và cấu trúc. Nhờ có EDM, hai thứ khác nhau này được liên kết với nhau. Nếu bạn thay một cơ sở dữ liệu khác, EDM lại giúp bạn tạo cầu nối mới mà không phải thay đổi các lớp model. Như vậy chương trình không phụ thuộc vào cơ s ở dữ liệu.



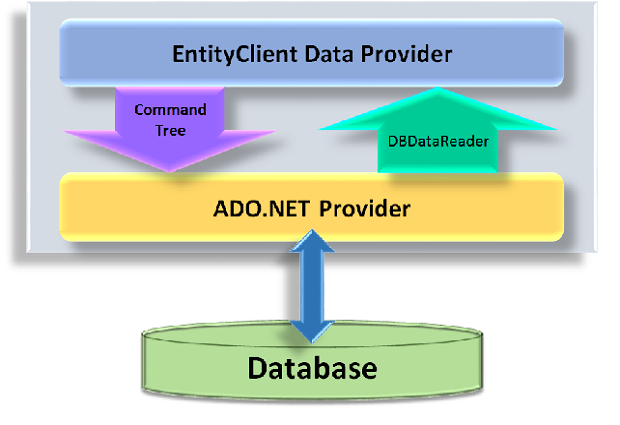
*Hình 2.5.1.1:Entity data model*

EDM có 3 thành phần (thực ra chỉ là 3 file xml):

* Conceptual model (đuôi csdl): chịu trách nhiệm lưu thông tin về các lớp model và quan hệ giữa chúng;
* Storage model (đuôi ssdl): lưu thông tin về cơ sở dữ liệu, bao gồm các bảng, view, stored procedure, khóa, quan hệ;
* Mapping model (đuôi msl): lưu thông tin ánh xạ giữa conceptual và storage model.

Thông tin từ các file này được sử dụng để cung cấp metadata cho các thành phần còn lại.Một điều vô cùng may mắn là EDM hầu như được tạo tự động hết.

1. Entity Client data provider

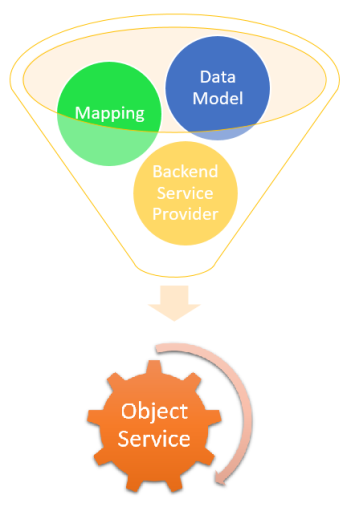


*Hình 2.5.2.1: Entity Client data provider*

Tầng này chịu trách nhiệm tương tác với Data provider của ADO.NET. Tầng này thực hiện chuyển đổi truy vấn LINQ to Entities hoặc Entity SQL về truy vấn SQL. Sau đó Data provider sẽ thực thi truy vấn SQL. Như bạn đã biết, Data provider chịu trách nhiệm tương tác về mặt vật lý với cơ sở dữ liệu.

Một nhiệm vụ khác của Entity Client là chuyển đổi kết quả thu được từ truy vấn SQL về một dạng trung gian (model tabular structure). Kết quả trung gian này chuyển lên cho Object Services xử lý.

1. Object Services

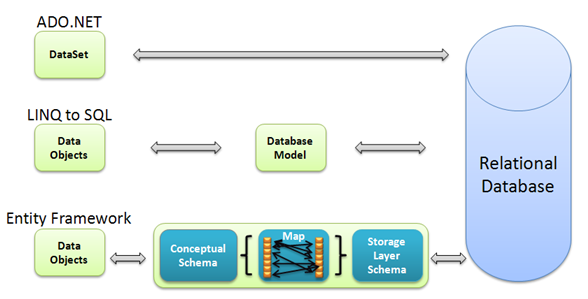


*Hình 2.5.3.1: Object Services*

Đây là cánh cổng để truy xuất dữ liệu. Tầng này thực hiện một quá trình gọi là **materialization** – quá trình chuyển đổi dữ liệu dạng bảng nhận từ tầng dưới (Entity Client data provider) về dạng object. Lưu ý rằng, dữ liệu dạng bảng nhận từ tầng dưới không phải là dữ liệu dạng bảng thu trực tiếp từ cơ sở dữ liệu (gọi là database tabular structure) mà là một dạng trung gian có tên gọi là model tabular structure.

Ngoài ra tầng này cũng đảm nhiệm việc quản lý trạng thái (state management). Hiểu một cách đơn giản, đây là việc theo dõi các thay đổi (change tracking) của từng object. Quản lý trạng thái có vai trò đặc biệt quan trọng khi thực hiện các thao tác thêm mới, cập nhật, xóa bỏ dữ liệu. Nhờ có nó bạn thoải mái thực hiện các thao tác trên dữ liệu và lưu lại tất cả thay đổi chỉ với một dòng code!

1. LINQ to Entities và Entity SQL.



*Hình 2.5.4.1: LINQ to Entities và Entity SQL.*

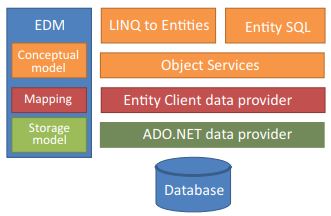
Đây là hai ngôn ngữ dùng để truy vấn dữ liệu.Sở dĩ có 2 ngôn ngữ truy vấn là vì Entity Framework đã xây dựng trước. Entity SQL được tạo ra làm ngôn ngữ truy vấn.

Sau đó LINQ mới ra đời và đội ngũ phát triển Entity Framework đưa LINQ to Entities vào EF.Nếu đã quen thuộc với LINQ thì bạn tiếp có thể tục sử dụng nó cho LINQ to Entities vì nó rất giống nhau.

Bạn không cần học Entity SQL làm gì. Entity SQL có cú pháp nhìn hơi khó chịu.

1. Làm việc với Entity Framework
2. Các giai đoạn làm việc

Nếu nhớ lại kiến trúc Entity Framework vừa trình bày ở phần trên bạn sẽ thấy nó phân chia làm hai khối rõ ràng:



*Hình 2.6.1.1: Kiến trúc Entity Framework*

Hai khối này liên quan đến hai giai đoạn làm việc với Entity Framework:

* Xây dựng Entity Data Model;
* Truy vấn và xử lý dữ liệu.

Ở giai đoạn xây dựng EDM bạn có bốn cách tiếp cận khác nhau: Database-first, Model-first, Code-first, Code-first from Database. Lựa chọn cách nào phụ thuộc vào từng điều kiện cụ thể. Dù chọn cách nào, cuối cùng bạn cũng thu được đủ bộ Entity Data Model (EDM). Tuy nhiên cái bạn trực tiếp sử dụng trong chương trình chỉ là thành phần Conceptual Model.

Việc lựa chọn hướng tiếp cận xây dựng EDM không có ảnh hưởng gì đến giai đoạn xử lý.

Ở giai đoạn xử lý dữ liệu, bạn sử dụng LINQ to Entities và Object Services để làm việc với dữ liệu dựa trên EDM đã xây dựng. Trong đó: LINQ to Entities giúp truy vấn (đọc dữ liệu); Object Services giúp thực hiện các thao tác theo dõi và ghi dữ liệu. Bạn cũng không bao giờ phải tương tác các tầng bên dưới.

Nhìn lại sơ đồ kiến trúc trên, cái mà bạn trực tiếp làm việc cùng (Conceptual Model, LINQ to Entities, Object Services) là lớp trên cùng và đều được tô màu vàng.

1. Các hướng tiếp cận khi sử dụng EMD
   1. . Database-first

Hướng tiếp cận này sẽ giúp sinh ra model từ cơ sở dữ liệu có sẵn, từ stored procedures hoặc views.

Bạn chỉ cần tinh chỉnh lại model (domain classes) theo nhu cầu của mình là xong. Việc tinh chỉnh này thực hiện qua một giao diện đồ họa (visual designer) rất tiện lợi. Nói cách khác, bạn xuất phát từ Storage Model có sẵn, EF sẽ giúp bạn sinh ra Conceptual Model và Mapping.

* 1. . Model-first

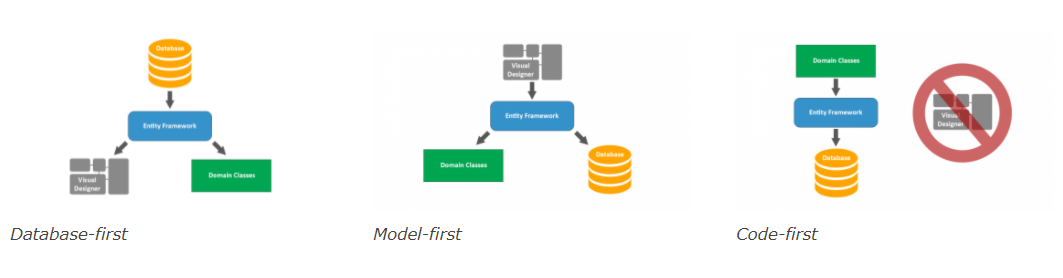
Đây là hướng tiếp cận thứ hai xuất hiện từ EF4. Hướng tiếp cận này cho phép bạn sử dụng một giao diện đồ họa để thiết kế model trước. Sau đó trình thiết kế này sẽ sinh ra các lớp model tương ứng cũng như mã SQL để tạo cơ sở dữ liệu. Nói cách khác, bạn xuất phát từ một giao diện thiết kế riêng biệt. EF sẽ giúp bạn sinh ra cả ba thành phần của EDM.

* 1. . Code-first

Đây là hướng tiếp cận được lập trình viên hoan nghênh nhất. Code-first xuất hiện từ EF4.1.

Với vai trò lập trình viên, bạn không cần quan tâm đến SQL hay giao diện thiết kế đồ họa nữa. Tất cả những gì cần làm là xây dựng các lớp model (domain classes) theo đúng kiểu hướng đối tượng mà bạn quen thuộc nhất. Entity Framework sẽ làm tất cả những gì còn lại để bạn có một cơ sở dữ liệu phù hợp. Nói theo kiểu khác, bạn tự mình code Conceptual Model, EF sẽ giúp bạn sinh ra Mapping và Storage Model.

Code-first cũng là hướng tiếp cận chúng ta sẽ tiếp cận đầu tiên và tập trung nhiều thời gian nhất.



*Hình 2.6.2.1: So sánh sơ đồ 3 cách tiếp cận*