Getting Started with Entity Framework 6

Overview of Entity Framework 6

Julie Lerman

thedatafarm.com | @julielerman

This Course

High level view of EF6

Creating a model with code

Use EF's DbContext between model & DB

Your model & EF in client & server apps

This Module

What is Entity Framework?

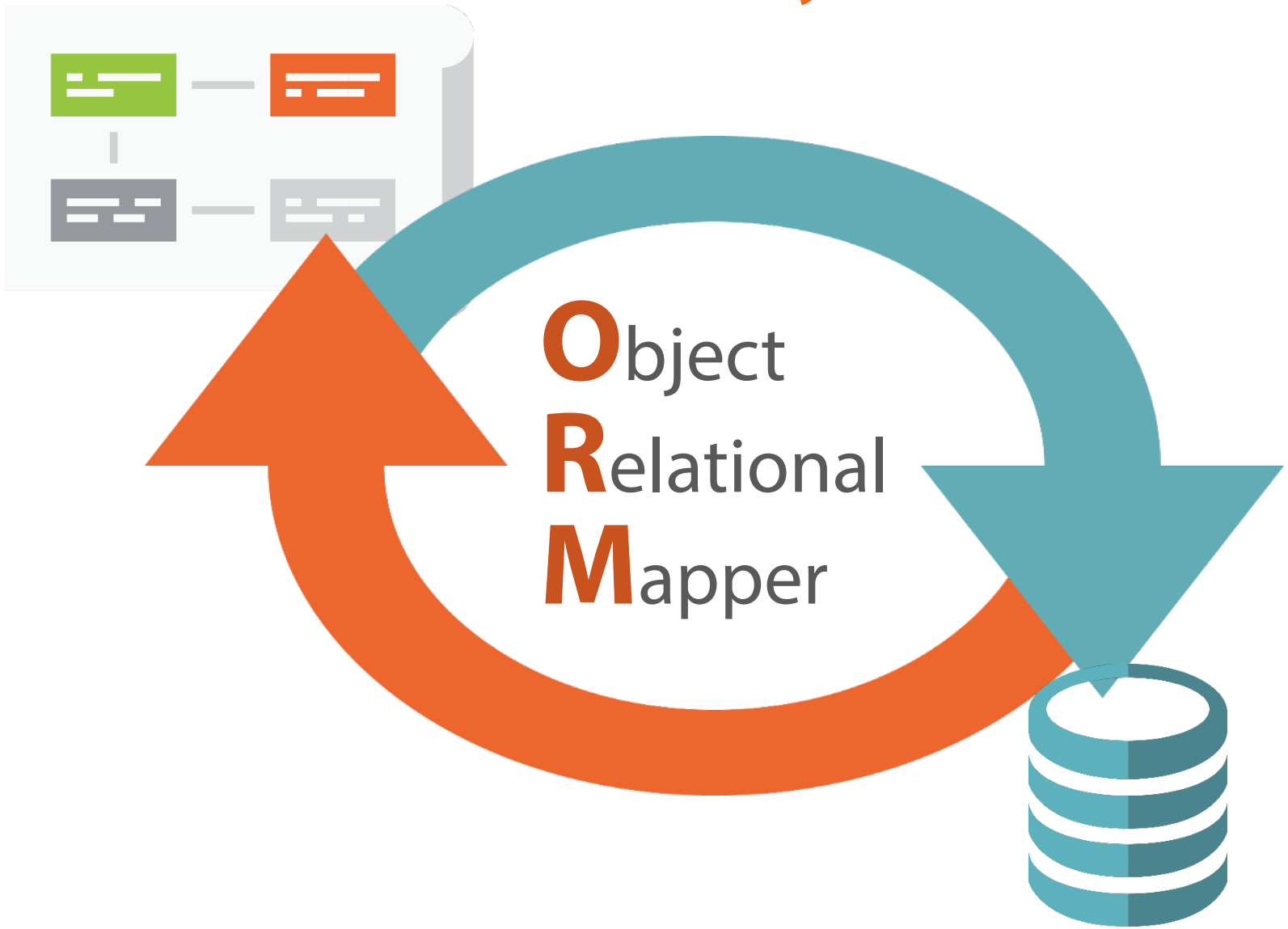
Why use Entity Framework?

A brief history in EF time

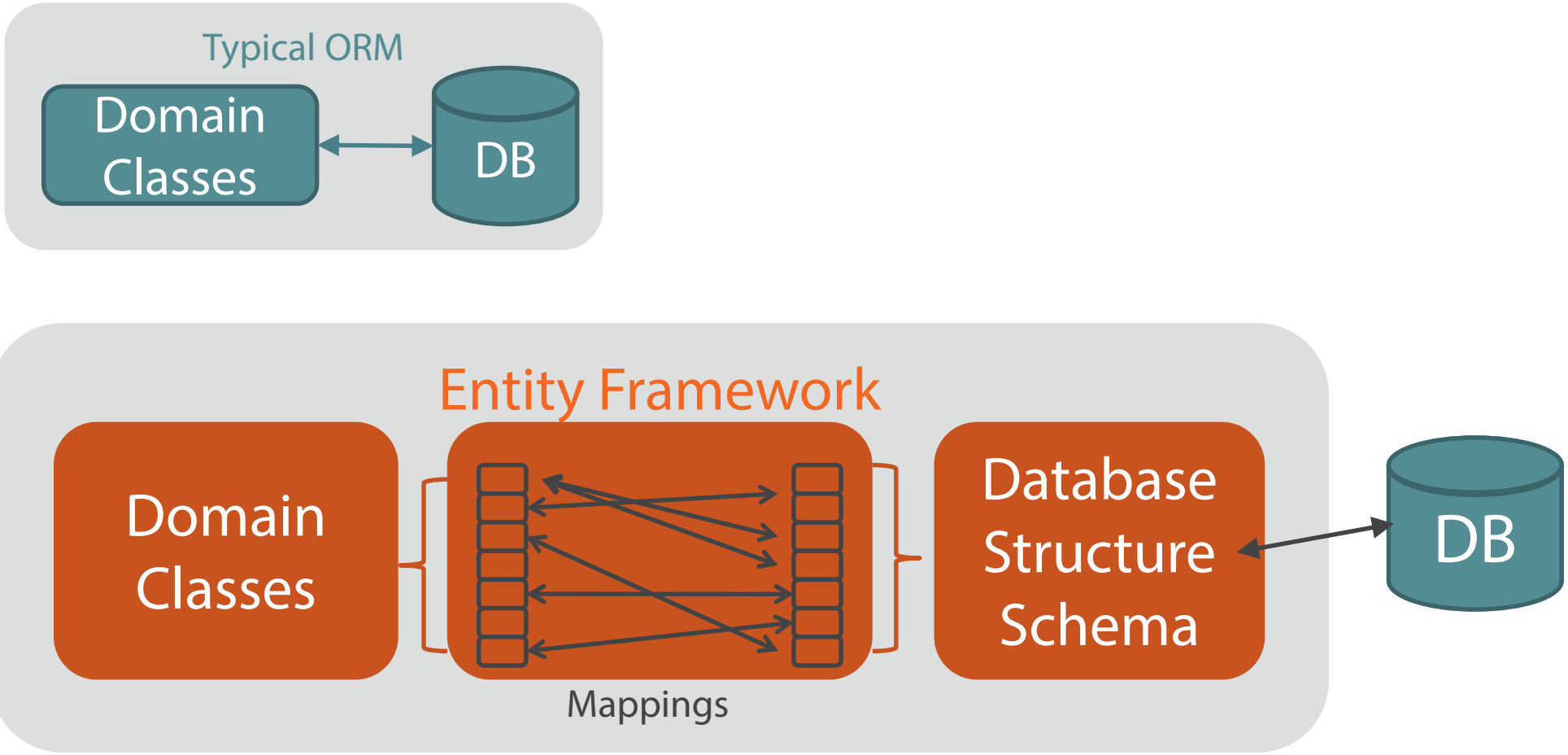
Where EF fits in your architecture

Should you use EF6 or EF7?

What Is Entity Framework?



EF vs. Other ORMs



Why Entity Framework?

Developer Productivity

First Class Member of Microsoft .NET Stack

Consistent query syntax with LINQ to Entities

Focus on domain. Not on DB, connections, commands, etc.

Where to Use Entity Framework 6

.NET 4, .NET 4.5, .NET 4.51, .NET 4.6

WPF
Windows Forms
Console Application

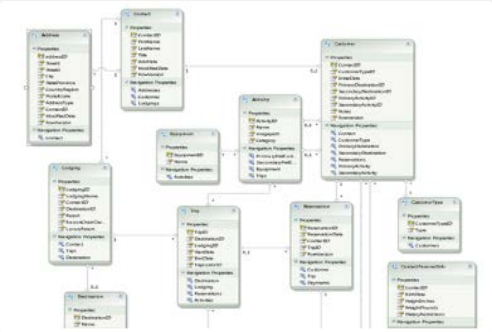
Client Side

ASP.NET MVC Controller
ASP.NET Web API
ASP.NET Web Forms
WCF Services
WCF Data Services
Windows Services

Server Side

How EF Works

Your Classes

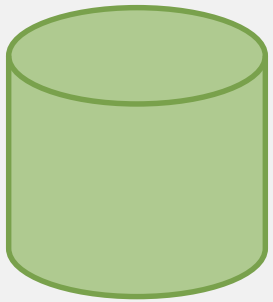


(via *Designer + Code Gen* or *Your Code*)

Entity Framework API

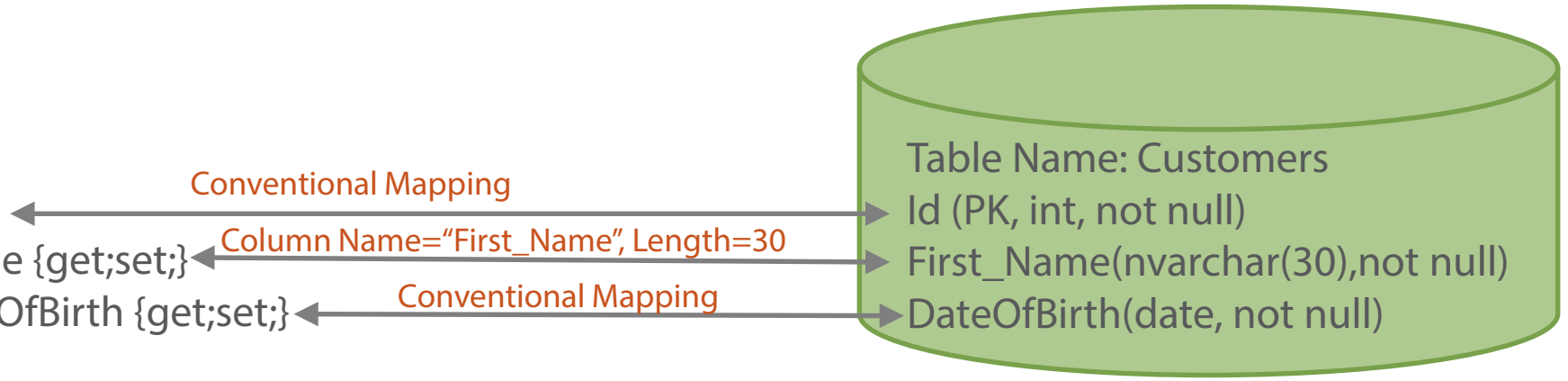
- Map Classes to Database Schema
- Translate & Execute Queries
- Track Changes
- Infer & Execute Updates

Relational DB



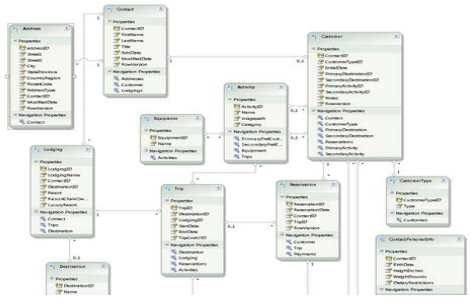
Code First Mappings

```
public class Customer
{
  public int Id {get;set;}
  public string FirstName {get;set;}
  public DateTime DateOfBirth {get;set;}
}
```



Basic Workflow

Define model



Express & execute query

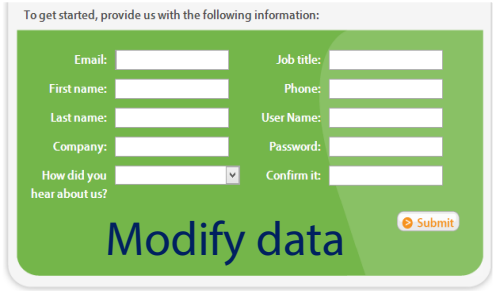
```
(from p in
people select
p).ToList()
```

EF determines & executes SQL

```
Select * from
people
```

EF transforms results into your types

3	Ms.	Donnie	F.	Carreras
4	Ms.	Janet	M.	Gates
5	Mr.	Lucy	*NULL*	Harrington
6	Mr.	Joop	X.	Carroll
7	Mr.	Dominic	P.	Gash
10	Ms.	Kathleen	M.	Garza
11	Ms.	Kathleen	*NULL*	Harding
12	Mr.	Johnny	A.	Caprio
16	Mr.	Christopher	R.	Beck
18	Mr.	David	J.	Liu
19	Mr.	John	A.	Beaver

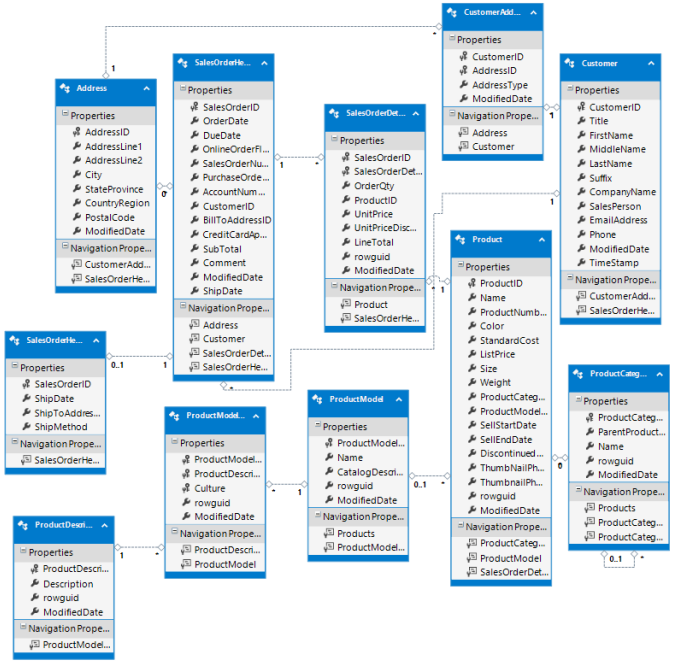


EF SaveChanges

EF determines & executes SQL

```
UPDATE people
SET Firstname='Julie'
WHERE id=3
```

Model Options




```
public partial class Customer
{
    public Customer()
    {
        this.CustomerAddresses = new HashSet<CustomerAddress>();
        this.SalesOrderHeaders = new HashSet<SalesOrderHeader>();
    }

    public int CustomerID { get; set; }
    public string Title { get; set; }
    public string FirstName { get; set; }
    public string MiddleName { get; set; }
    public string LastName { get; set; }
    public string Suffix { get; set; }
    public string CompanyName { get; set; }
    public string SalesPerson { get; set; }
    public string EmailAddress { get; set; }
    public string Phone { get; set; }
    public System.DateTime ModifiedDate { get; set; }
    public byte[] TimeStamp { get; set; }

    public virtual ICollection<CustomerAddress> CustomerAddresses { get; set; }
    public virtual ICollection<SalesOrderHeader> SalesOrderHeaders { get; set; }
}
```

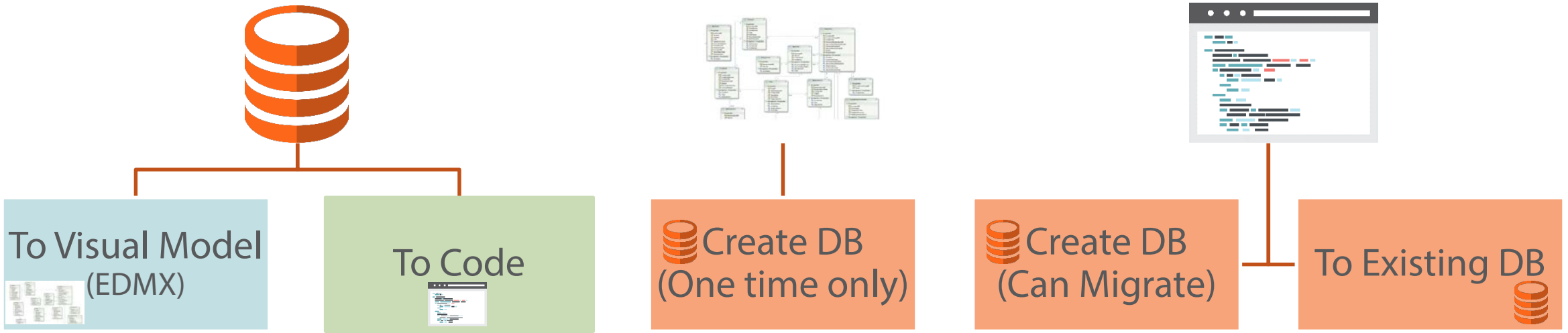
Model.EntityType="Customer"
Property.Name="Title"
Key.Name="First_Name"

Storage.EntityType="Customer"
Property.Name="Title"
Key.Name="First_Name"

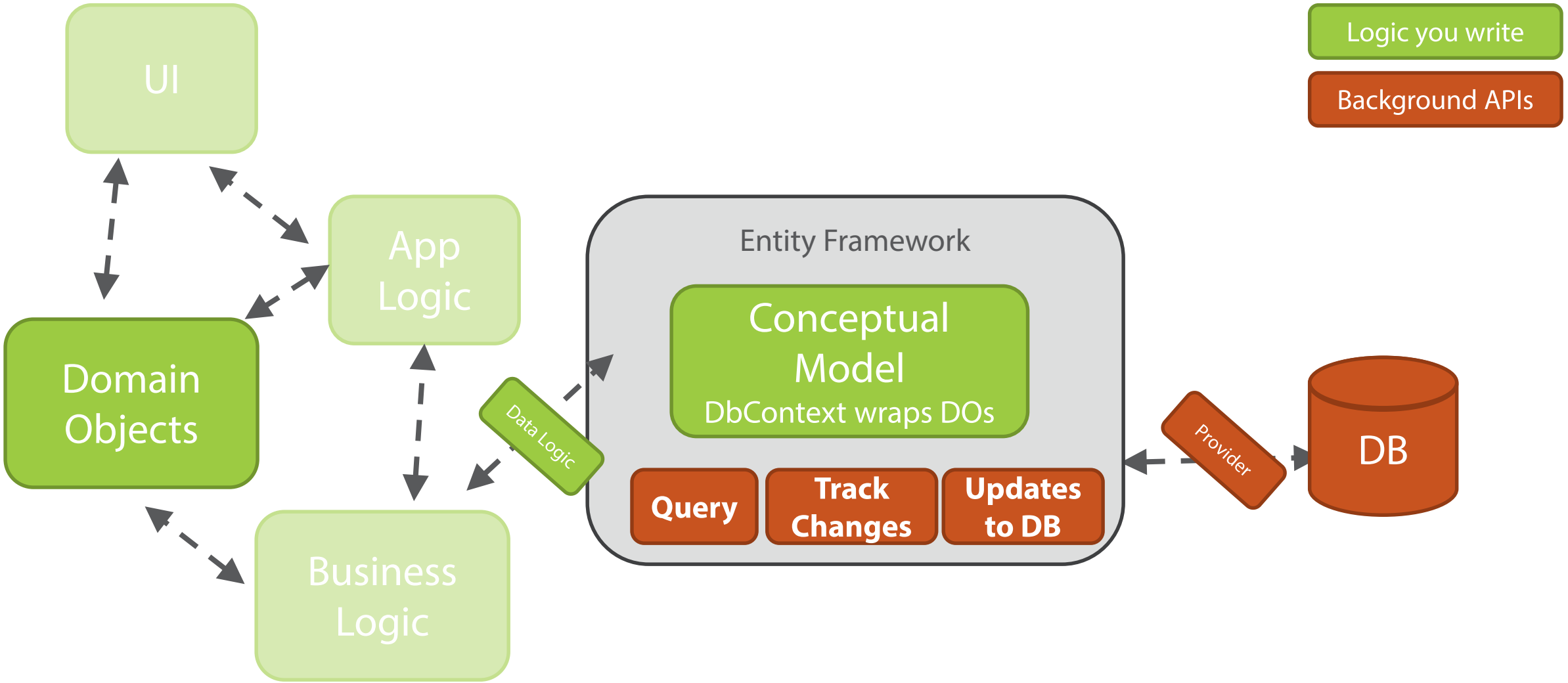
Mapping:
[Model.Entity("Customer"),
Storage.Entity("Customers")]
[Model.Property("FirstName"),
Storage.Property("First_Name")]

```
public class Customer : Contact
{
  public Customer(string firstName,string lastName, string email)
  {
    FullName = new FullName(firstName,lastName);
    EmailAddress = email;
    Status = CustomerStatus.Silver;
  }
  public Customer()    {    }
  public string SalesPersonId { get; private set; }
  public CustomerStatus Status { get; private set; }
  public Address ShippingAddress { get; private set; }
  public Address BillingAddress { get; private set; }
  public CustomerCreditCard CreditCard { get; private set; }
  public void SetShippingAddressBillingAddress()
  {
    ShippingAddress = new Address
                      {
                        Street1 = BillingAddress.Street1,
                        City = BillingAddress.City,
                        PostalCode = BillingAddress.PostalCode
                      };
  }
    public void CreateNewShippingAddress(string street, string city, string zip)
  {
   BillingAddress = new Address
                    {
                      Street1 = street,
                      City = city,
                      PostalCode = zip
                    };
  }
```

Model Creation Options

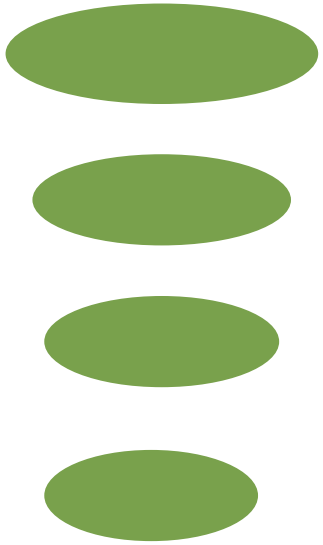


EF in Your Software Architecture



A Brief History in EF Time

2008
2010
2011
2012
2013



EF6

EF7

	EF6	EF7
Production Ready (as of Summer 2015)	✓	*pre-release subset with ASPNET5, RTM late 2015?
Actively Evolving	✓	✓
Visual Designer	✓	*expect 3rd party support
Backwards Compatible	✓	
Full .NET Support	✓	✓
Core CLR Support		✓
Lightweight API(s)		✓
Better APIs, New Features		✓
Non-Relational Data		✓

EF6 or EF7 Guidance

- Existing applications
  - Not required to update to EF7
  - Only plan to upgrade if new features are compelling
  - Minimize amount of code to update by breaking projects/models apart
  - Wait for RTM unless you need new features
- New applications
  - Working on future CoreCLR or ASPNET5 apps, use EF7
  - Otherwise keep using EF6
  - Plan ahead for potential update to EF7 as you design EF6 based code

Why This Course? Why This Module?

- Part of series to consolidate older Pluralsight EF courses
- Demonstrate long term commitment to EF6
- Get started using the current APIs and tools
- Provide high level basics of EF for decision makers
- Assure you understand that EF6 is here to stay
- Help to see how EF6 and EF7 align with your plans

Resources

- EF6 Development Site: entityframework.codeplex.com
- EF6 Ninja Edition: What's New in EF6 (Pluralsight): bit.ly/PS-EF6
- Looking Ahead to EF7 (Pluralsight): bit.ly/PS-EF7Alpha
- EF7 Development Site: github.com/aspnet/entityframework
- My Blog: thedatafarm.com/blog
- EF Team Blog: blogs.msdn.com/adonet

Julie Lerman

thedatafarm.com | @julielerman