Using EF Core 2 in Your Applications

Julie Lerman
MOST TRUSTED AUTHORITY ON ENTITY FRAMEWORK
@julielerman thedatafarm.com

Module Overview

Client apps where you can run EF Core

EF Core in a WPF application

Designing for connected data access

EF Core in an ASP.NET Core MVC app

User-friendly relationships in the MVC app

EF Core on the Desktop or Device

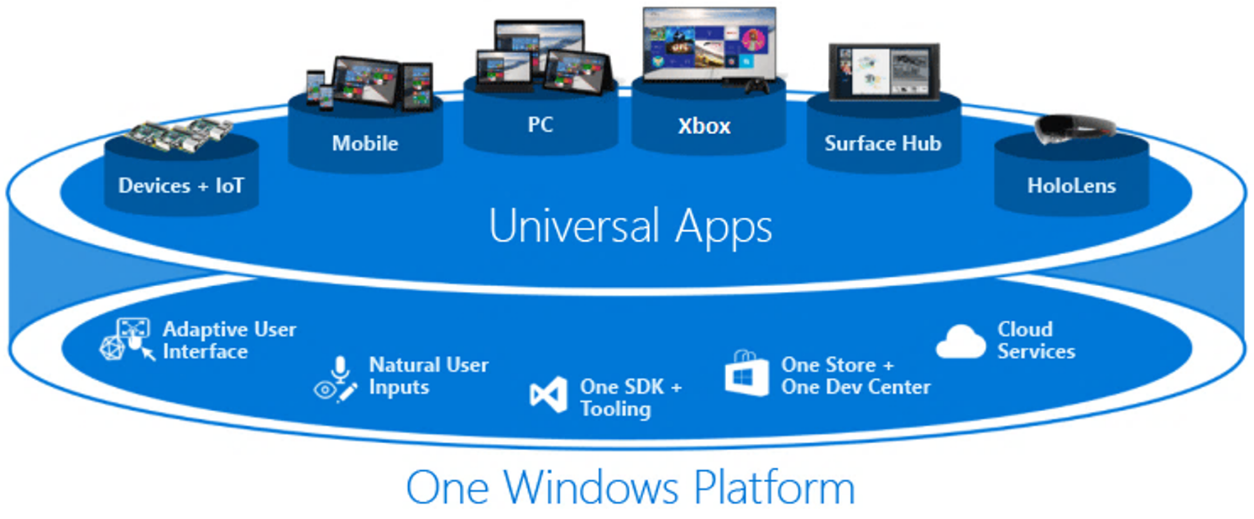
Desktop/Device and DbContext

EF Core sits on the client

Long-running DbContext ("connected") is most common

For complex/advanced architecture,
short-lived context ("disconnected") is a smart pattern to choose

EF Core 2 & Universal Windows Platform (UWP)

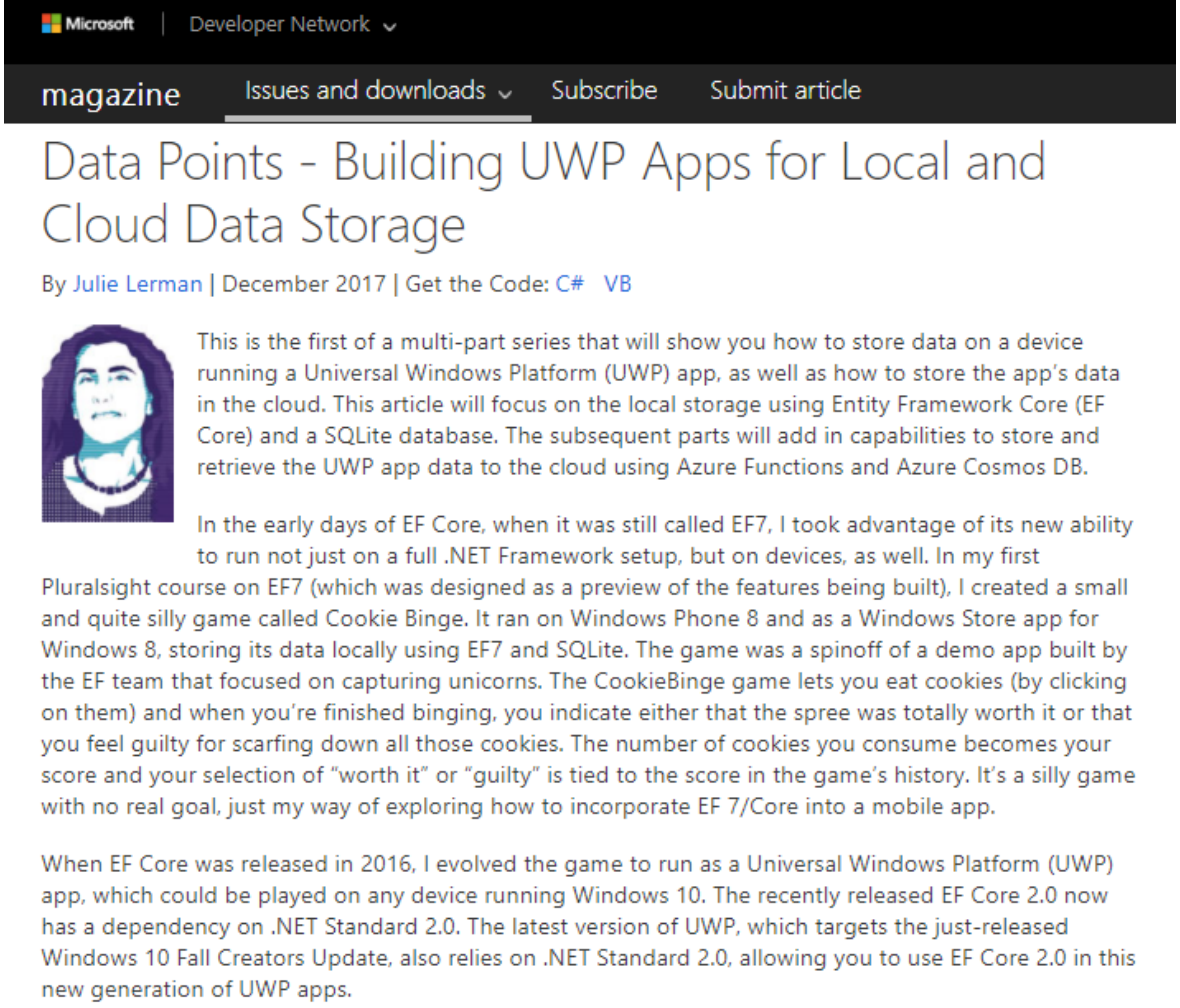


.NET Standard 2.0

Windows 10 Fall Creators Update

Migrations require a little extra work

Source: msdn.microsoft.com/en-us/windows/uwp/get-started/universal-application-platform-guide


Microsoft | Developer Network

magazine   Issues and downloads   Subscribe   Submit article

Data Points - Building UWP Apps for Local and Cloud Data Storage

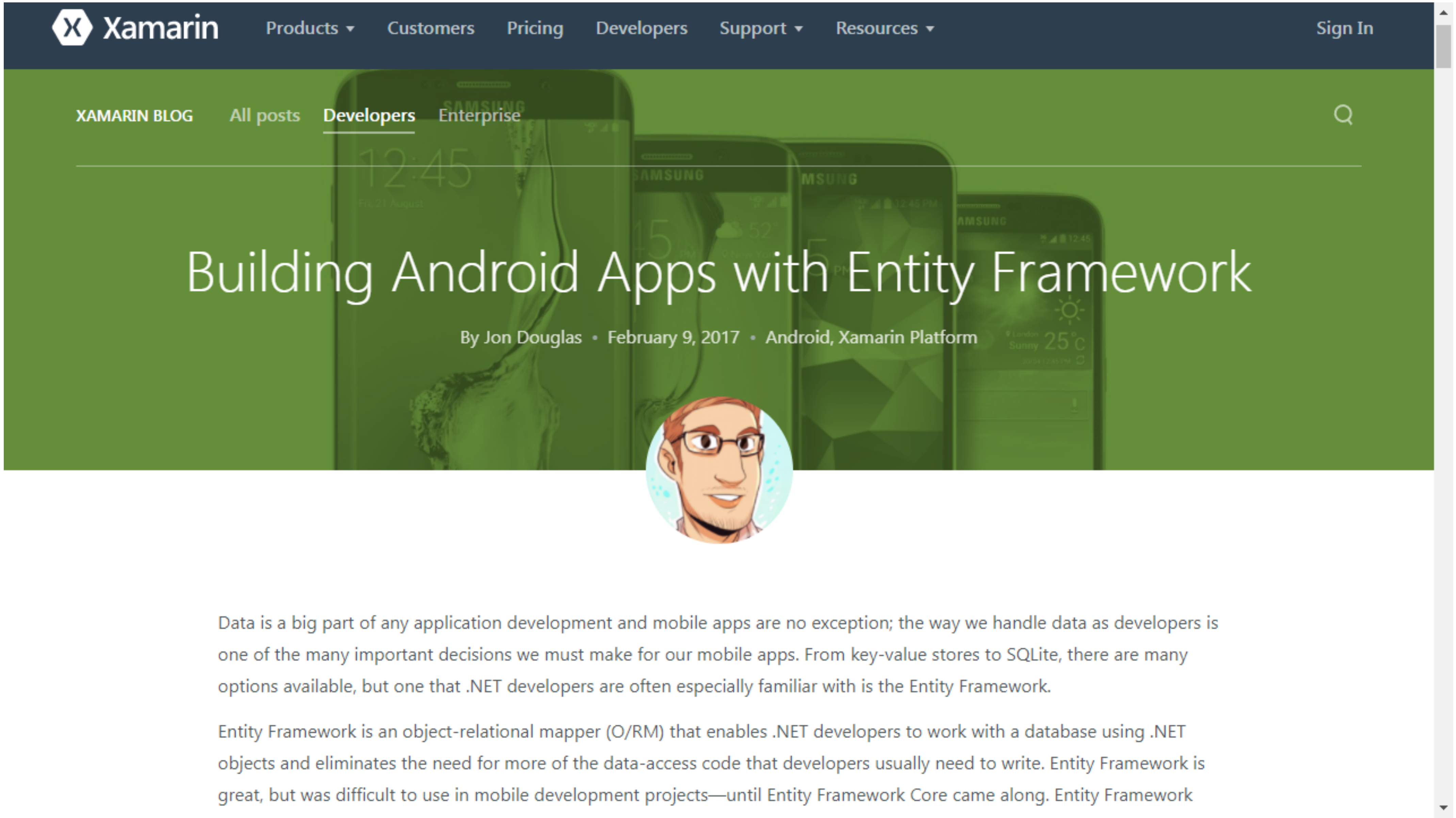
By Julie Lerman | December 2017 | Get the Code: C#   VB

This is the first of a multi-part series that will show you how to store data on a device running a Universal Windows Platform (UWP) app, as well as how to store the app's data in the cloud. This article will focus on the local storage using Entity Framework Core (EF Core) and a SQLite database. The subsequent parts will add in capabilities to store and retrieve the UWP app data to the cloud using Azure Functions and Azure Cosmos DB.

In the early days of EF Core, when it was still called EF7, I took advantage of its new ability to run not just on a full .NET Framework setup, but on devices, as well. In my first Pluralsight course on EF7 (which was designed as a preview of the features being built), I created a small and quite silly game called Cookie Binge. It ran on Windows Phone 8 and as a Windows Store app for Windows 8, storing its data locally using EF7 and SQLite. The game was a spinoff of a demo app built by the EF team that focused on capturing unicorns. The CookieBinge game lets you eat cookies (by clicking on them) and when you're finished binging, you indicate either that the spree was totally worth it or that you feel guilty for scarfing down all those cookies. The number of cookies you consume becomes your score and your selection of "worth it" or "guilty" is tied to the score in the game's history. It's a silly game with no real goal, just my way of exploring how to incorporate EF 7/Core into a mobile app.

When EF Core was released in 2016, I evolved the game to run as a Universal Windows Platform (UWP) app, which could be played on any device running Windows 10. The recently released EF Core 2.0 now has a dependency on .NET Standard 2.0. The latest version of UWP, which targets the just-released Windows 10 Fall Creators Update, also relies on .NET Standard 2.0, allowing you to use EF Core 2.0 in this new generation of UWP apps.

Target Android, iOS & Mac with Xamarin Forms



Building Android Apps with Entity Framework

By Jon Douglas • February 9, 2017 • Android, Xamarin Platform

Data is a big part of any application development and mobile apps are no exception; the way we handle data as developers is one of the many important decisions we must make for our mobile apps. From key-value stores to SQLite, there are many options available, but one that .NET developers are often especially familiar with is the Entity Framework.

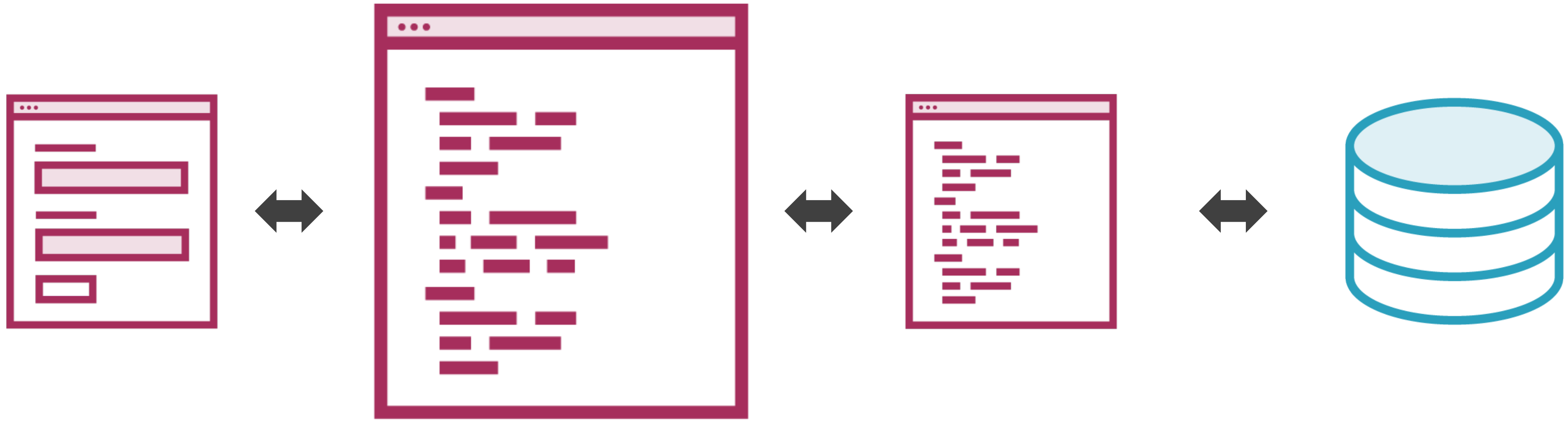
Entity Framework is an object-relational mapper (O/RM) that enables .NET developers to work with a database using .NET objects and eliminates the need for more of the data-access code that developers usually need to write. Entity Framework is great, but was difficult to use in mobile development projects—until Entity Framework Core came along. Entity Framework

The Desktop Application: Windows Presentation Foundation (WPF)

Creating the WPF Application

WPF data-binding is no different than when using earlier versions of EF

Data Access Wrapper Contains EF Calls



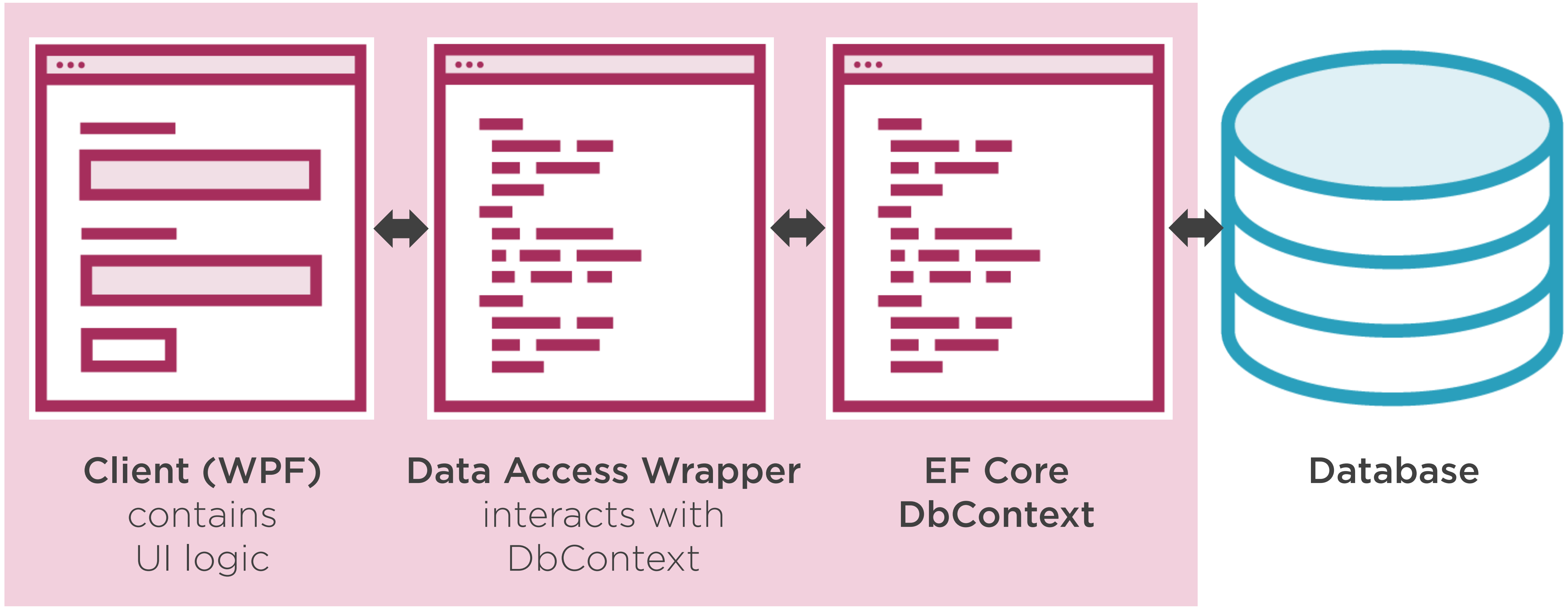
User Interface
contains
UI logic

Data Access Wrapper
interacts with
DbContext

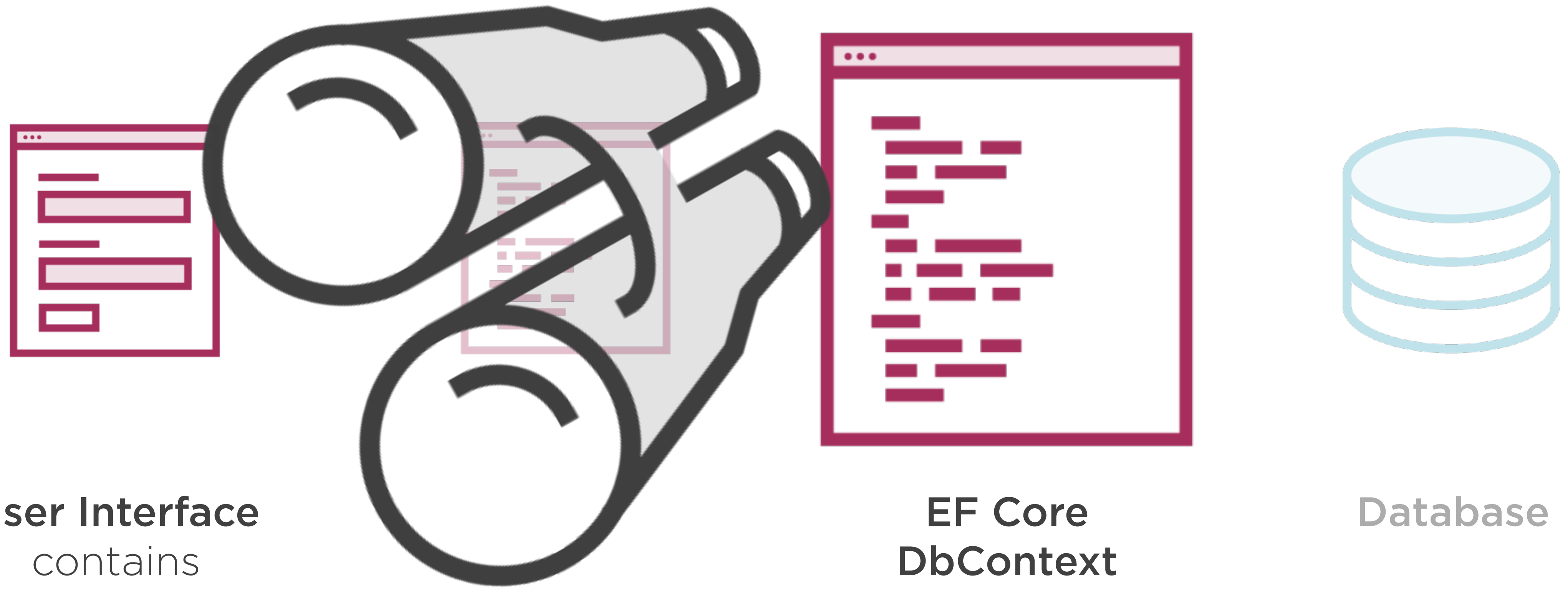
EF Core
DbContext

Database

Connected: EF Core on Device with UI



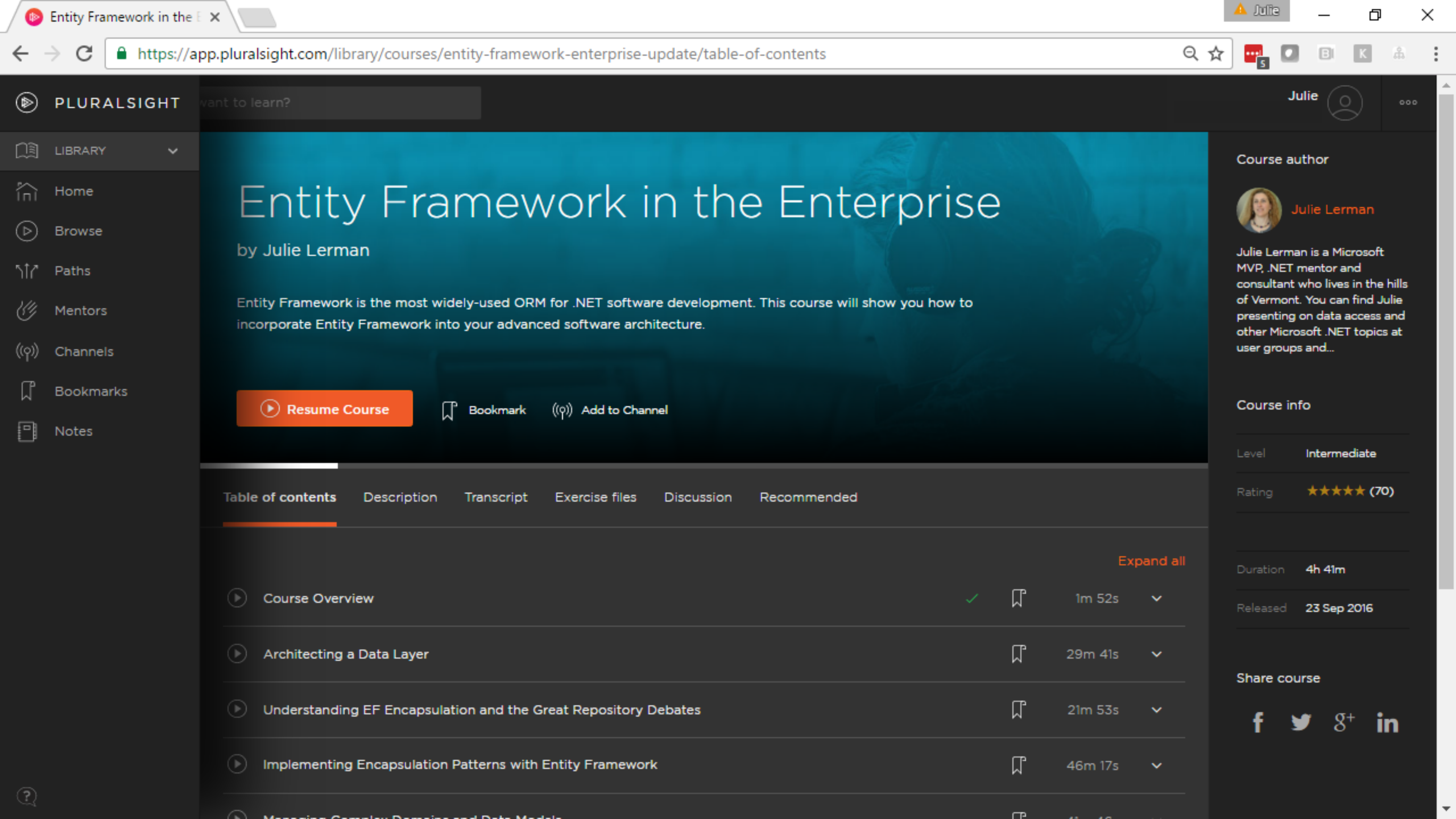
Connected: DbContext Can Track Entities



User Interface
contains
UI logic

EF Core
DbContext

Database


Entity Framework in the E

https://app.pluralsight.com/library/courses/entity-framework-enterprise-update/table-of-contents

PLURALSIGHT

want to learn?

Julie

LIBRARY

Home

Browse

Paths

Mentors

Channels

Bookmarks

Notes

Entity Framework in the Enterprise

by Julie Lerman

Entity Framework is the most widely-used ORM for .NET software development. This course will show you how to incorporate Entity Framework into your advanced software architecture.

Resume Course

Bookmark

Add to Channel

Table of contents | Description | Transcript | Exercise files | Discussion | Recommended

Expand all

Course Overview	✓	1m 52s
Architecting a Data Layer		29m 41s
Understanding EF Encapsulation and the Great Repository Debates		21m 53s
Implementing Encapsulation Patterns with Entity Framework		46m 17s
Managing Complex Domains and Data Models		

Course author

Julie Lerman

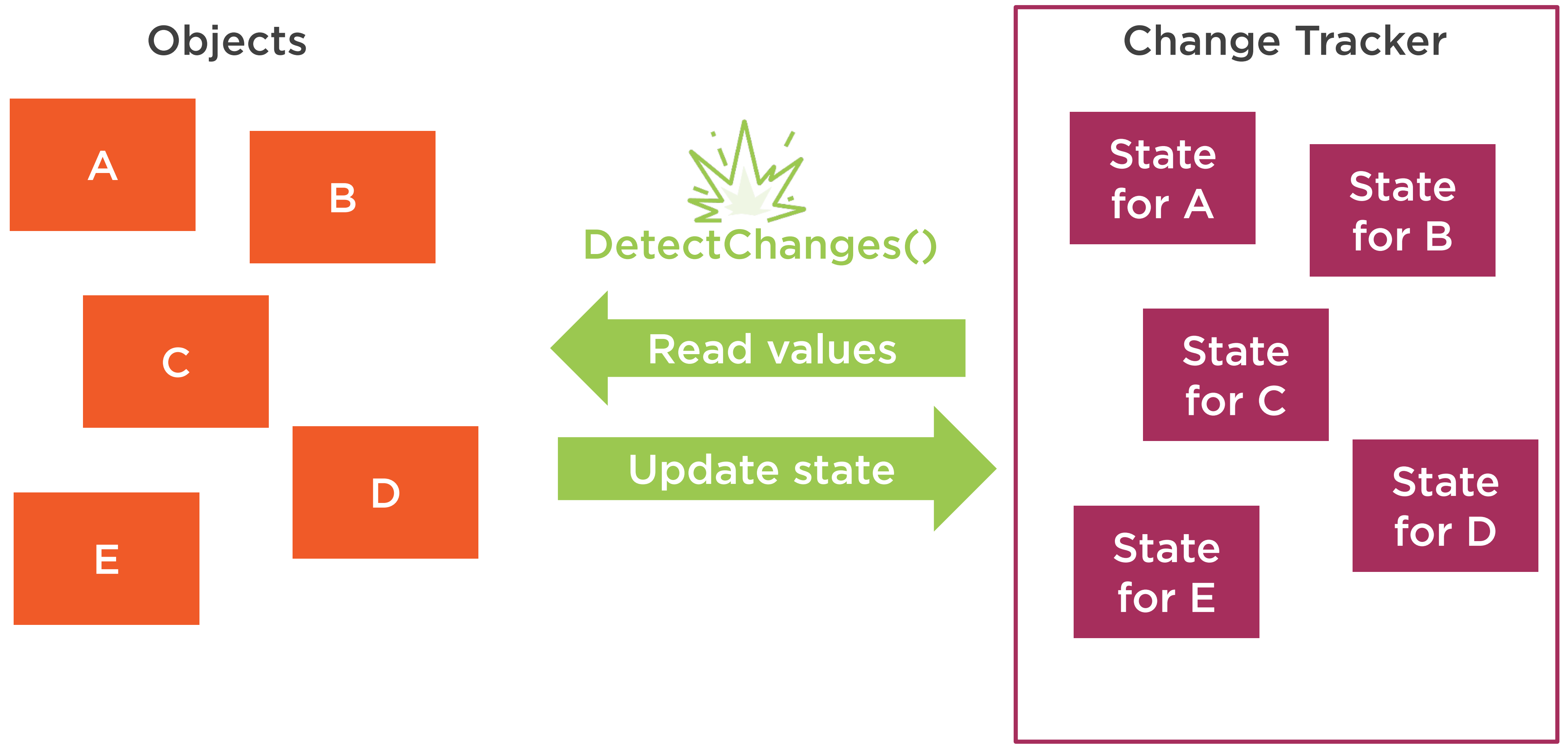
Julie Lerman is a Microsoft MVP, .NET mentor and consultant who lives in the hills of Vermont. You can find Julie presenting on data access and other Microsoft .NET topics at user groups and...

Course info

Level	Intermediate
Rating	★★★★★ (70)
Duration	4h 41m
Released	23 Sep 2016

Share course

DbContext Fixes up State Periodically



Create & Migrate Database at Runtime

Existing migrations can be executed in code

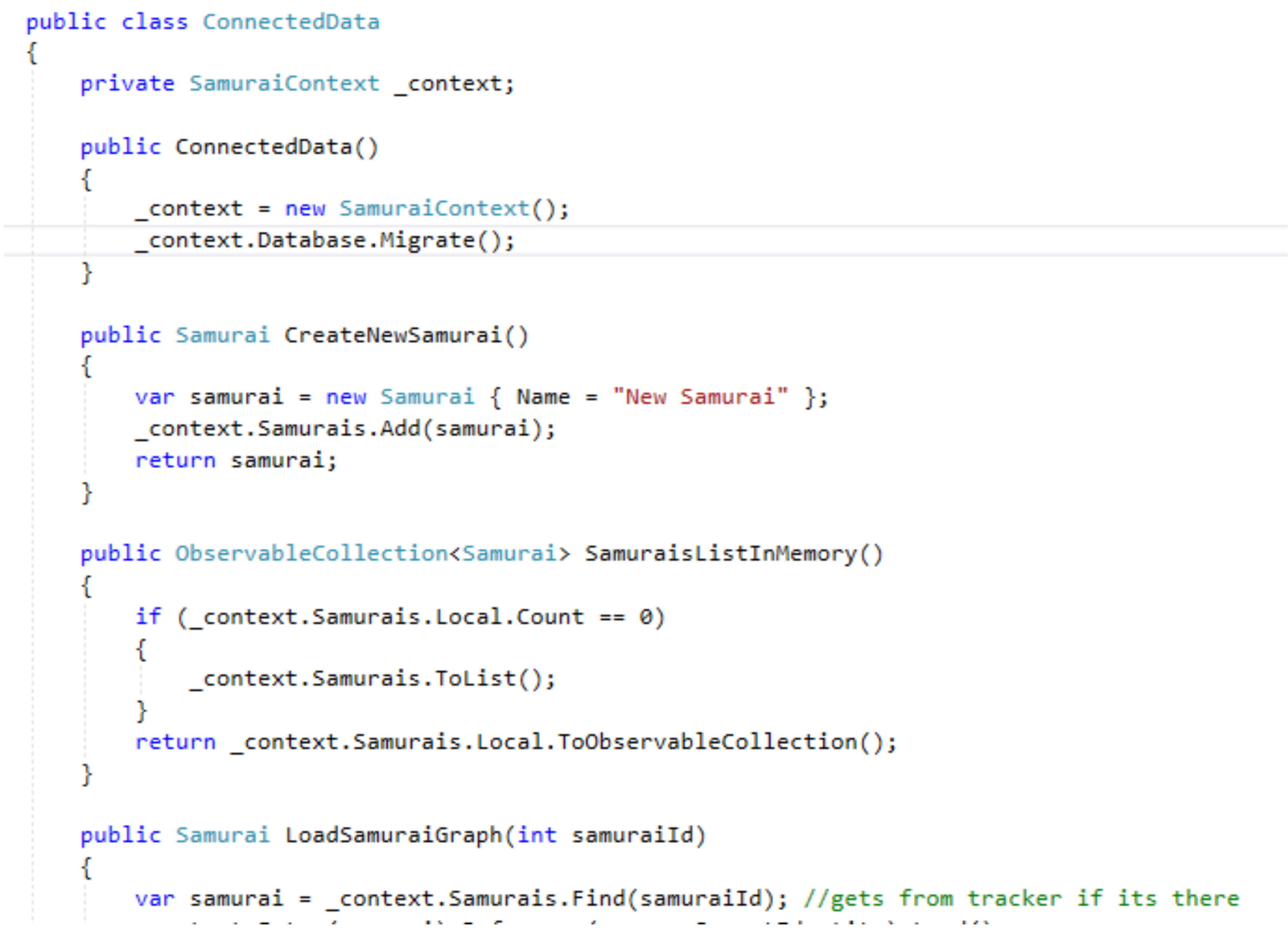
Will create database locally

Useful for desktop/device bound apps

Walking Through the WPF Data Access

ConnectedData.cs does not follow repository pattern

It encapsulates the specific calls needed by my app



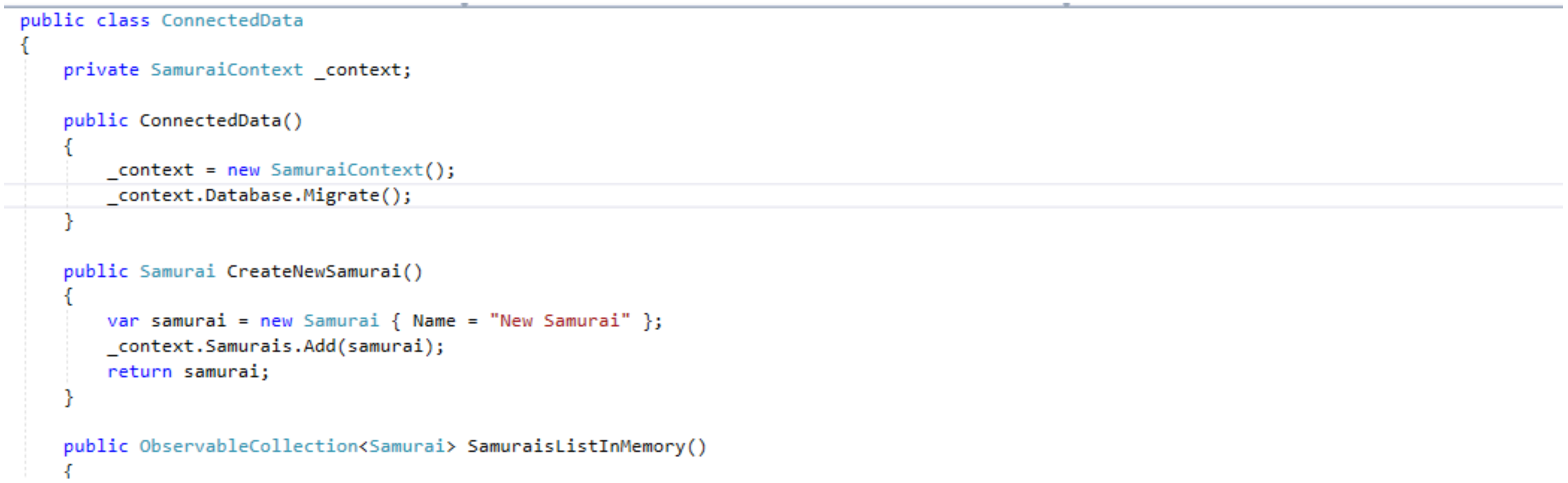
```
public class ConnectedData
{
    private SamuraiContext _context;

    public ConnectedData()
    {
        _context = new SamuraiContext();
        _context.Database.Migrate();
    }

    public Samurai CreateNewSamurai()
    {
        var samurai = new Samurai { Name = "New Samurai" };
        _context.Samurais.Add(samurai);
        return samurai;
    }

    public ObservableCollection<Samurai> SamuraisListInMemory()
    {
        if (_context.Samurais.Local.Count == 0)
        {
            _context.Samurais.ToList();
        }
        return _context.Samurais.Local.ToObservableCollection();
    }

    public Samurai LoadSamuraiGraph(int samuraiId)
    {
        var samurai = _context.Samurais.Find(samuraiId); //gets from tracker if its there
```



```
public class ConnectedData
{
    private SamuraiContext _context;

    public ConnectedData()
    {
        _context = new SamuraiContext();
        _context.Database.Migrate();
    }

    public Samurai CreateNewSamurai()
    {
        var samurai = new Samurai { Name = "New Samurai" };
        _context.Samurais.Add(samurai);
        return samurai;
    }

    public ObservableCollection<Samurai> SamuraisListInMemory()
    {
```

ConnectedData.cs is not a repository pattern

It is simply a class to encapsulate the data calls needed for my app

EF Core in ASP.NET Core MVC

Two ASP.NET Core Apps

Getting Started

MVC

Controllers and views

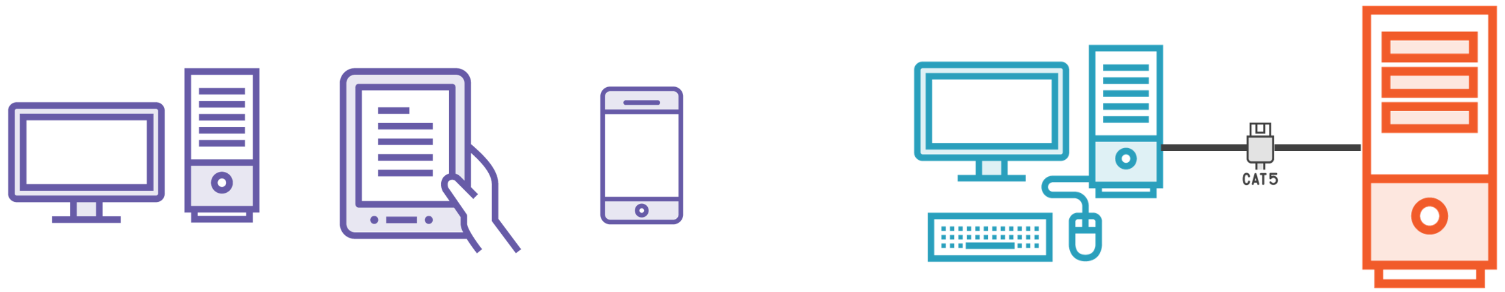
Intermediate

Web API

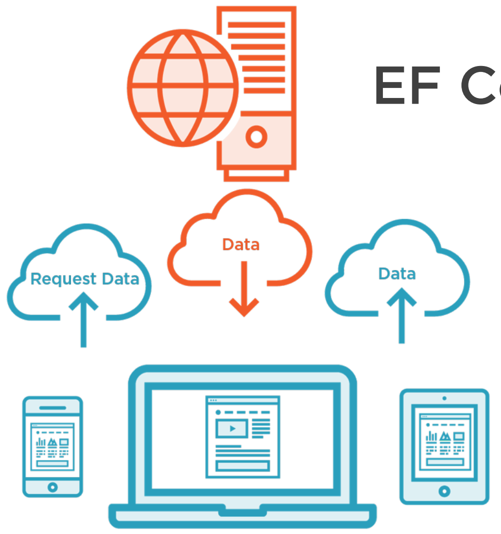
Disconnected access patterns

Desktop/Device vs Server

EF Core sits on the client



EF Core sits on the server



Adding Related Data into the MVC App

“MVC EF templates are clever, but not brilliant. You’ll need to apply your own brilliance here.”

- Me

Coding the MVC App’s Relationships

Controller and Markup Changes Summary

Replace queries for Samurai queries that eager load Quotes & SecretIdentity

Edit Samurai views to display SecretIdentity.RealName & Quotes

Round-trip SecretIdentity Id and SamuraiId with HTML hidden values

Removed ‘bind’ in Samurai Edit HTTPPost to allow related data to flow back

Explore the sample code

Review

EF Core can be used in a variety of apps

In connected app, design for the “always-tracking” context

Encapsulation/Separation of Concerns is useful even in small applications

Even when connected, context doesn’t update state in real time

MVC controller/view templates for EF don’t understand all relationships

Incorporate relationships with minimal changes to the code

Beware the controller template bug!

Resources

Entity Framework Core on GitHub github.com/aspnet/entityframework

EF Core Roadmap bit.ly/efcoreroadmap

EF Core Documentation docs.efproject.net

Entity Framework Core: Getting Started (for EF Core 1.1) bit.ly/PS_EFCoreStart

Building UWP Apps for Local and Cloud Data Storage msdn.com/magazine/mt814412

Building Android Apps with Entity Framework
blog.xamarin.com/building-android-apps-with-entity-framework/

Json Formatter Chrome Extension:
chrome.google.com/webstore/detail/json-formatter/

Fiddler – Web Debugging Tool: telerik.com/fiddler

Entity Framework Core 2: Getting Started

Julie Lerman
MOST TRUSTED AUTHORITY ON ENTITY FRAMEWORK

@julielerman thedatafarm.com