



The path to upgrading: Getting your teams on the latest version

Mark Hall

Director, Innovation and
Transformation



Dan Isaacs

Solution Architect

Jacob Khan

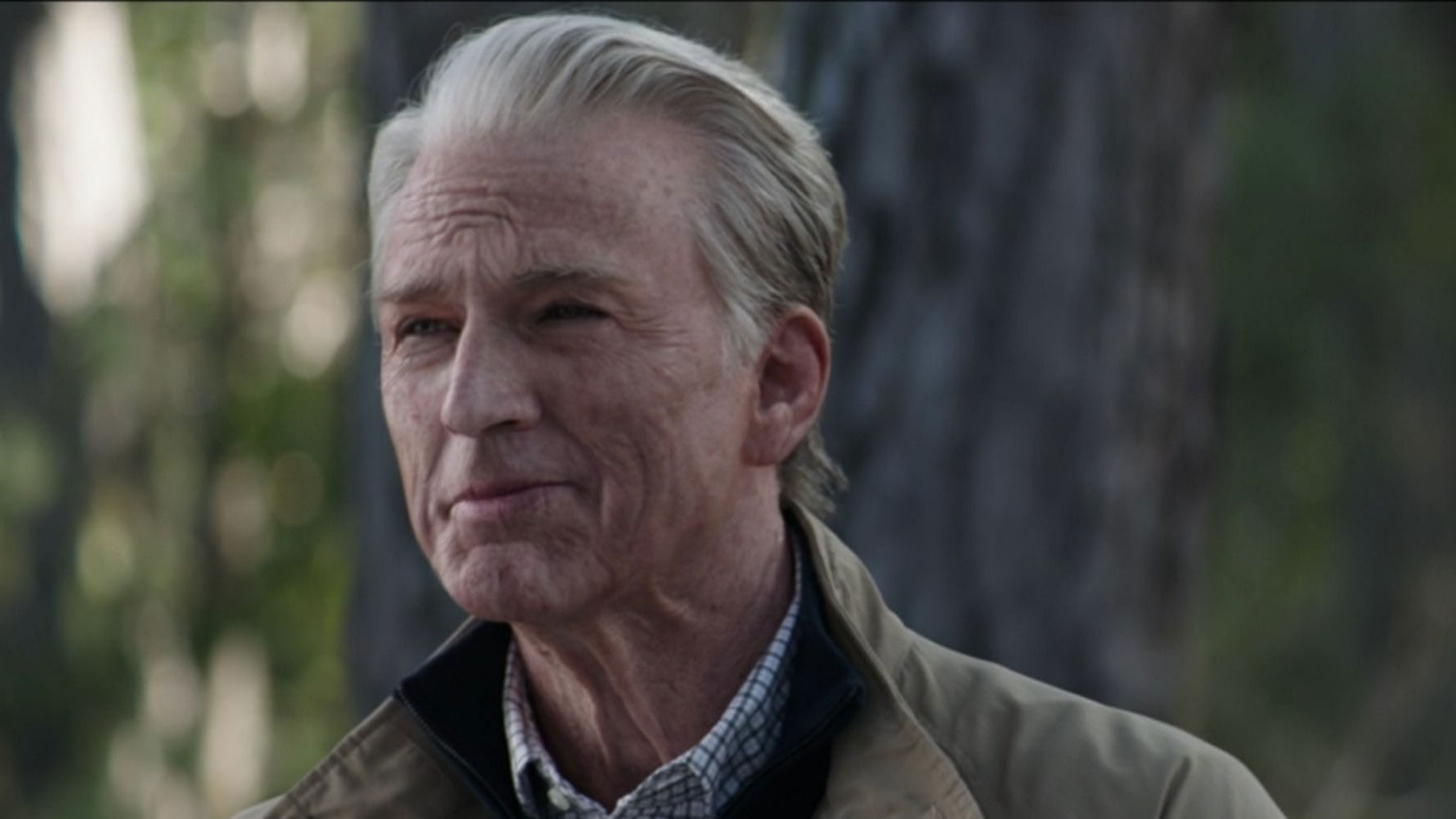
GVP, Solution Architect



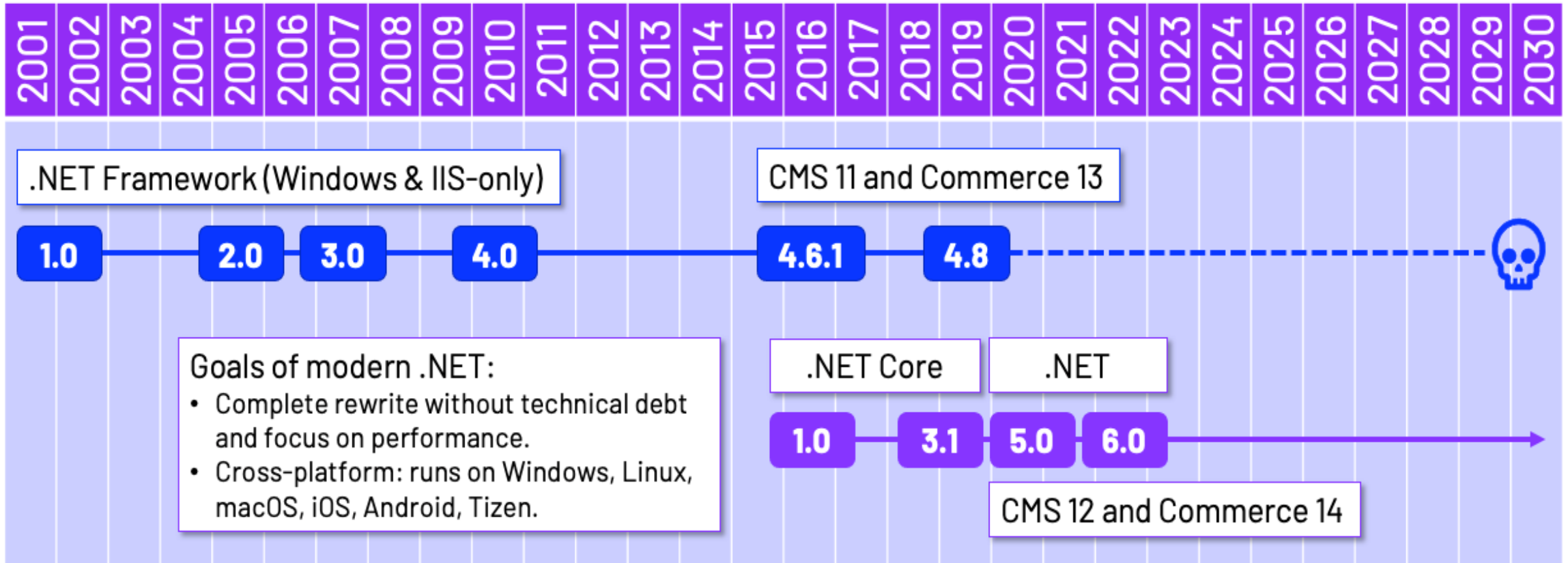
Agenda

Upgrading to the latest..

- What's the deal?
- Who has done this already?
- What is in the secret sauce?
- Developers will love this!
- How to upgrade...
- 4 reasons to upgrade now :D



.NET old is dying...



Why did we do this?

- Technology shift – modern .NET is the future according to Microsoft
- Performance improvement
 - **5-10x faster sites**
- It is a major release
- Old parts of UI removed and/or upgraded
- Including tools to help upgrading
- Higher developer satisfaction



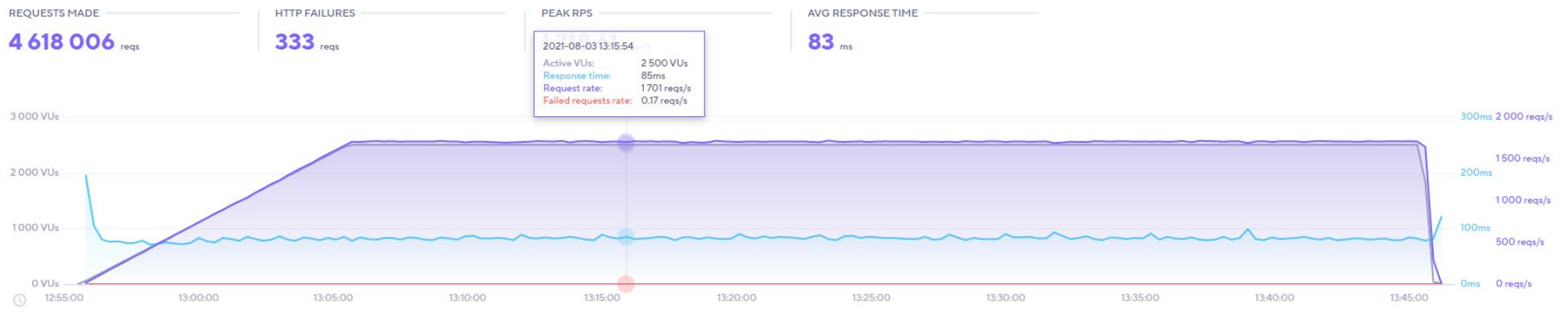
Average Response Time

Heavy load test based on 25/50K Concurrent Users on Optimizely Foundation

.NET 4.8 (CMS 11, Commerce 13)



.NET 5 (CMS 12, Commerce 14)



5.89x
faster

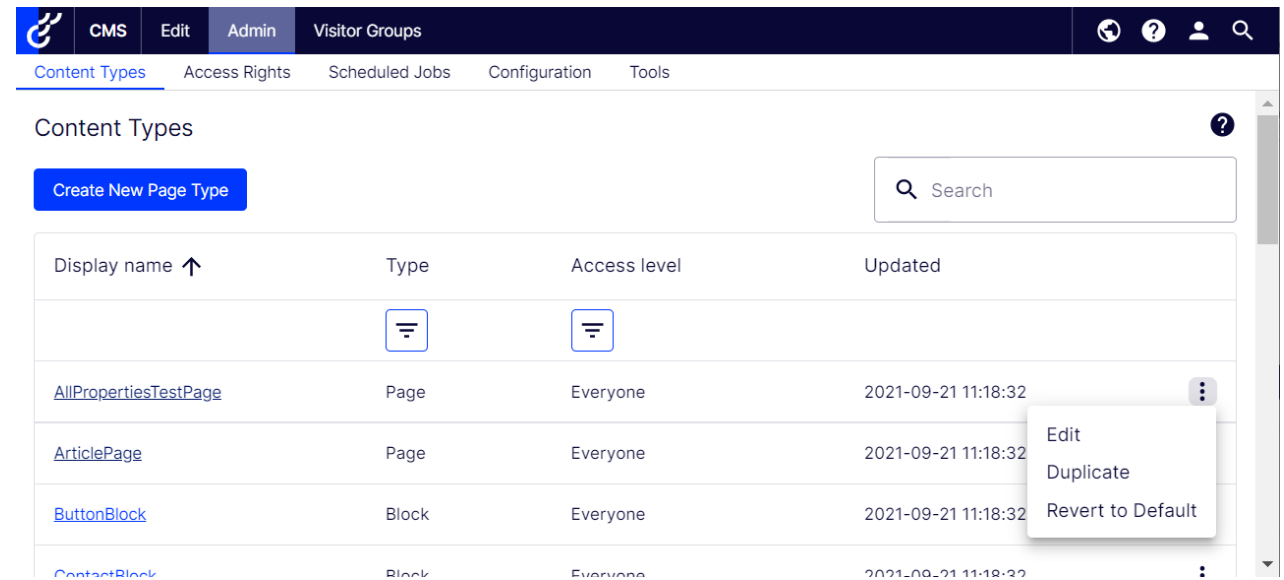
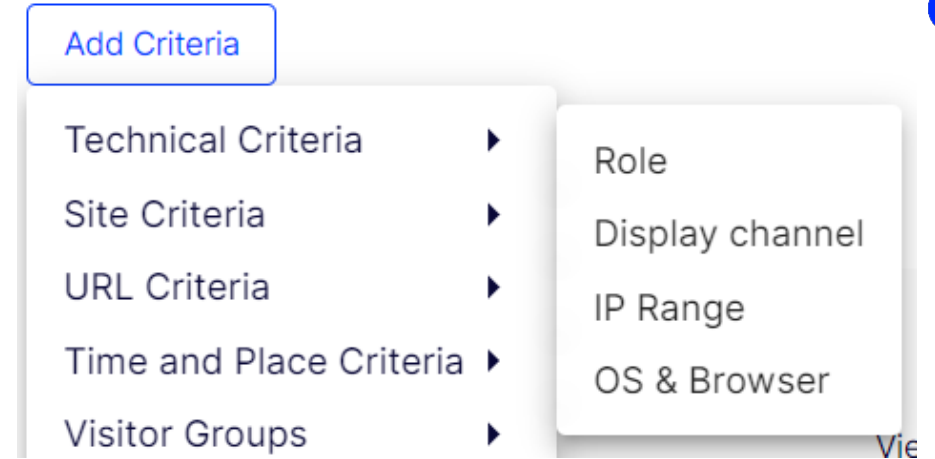
Based on avg response time

Note: These are preliminary results of internal testing based upon CMS 12 + Commerce 14 and the Foundation implementation. We intend to publish our methodology so partners and customers can do their own performance testing validation.



CMS 12 UI differences

- **Dashboard gone.**
- **Edit:** small changes.
- **Admin view** rebuilt using react
- **Visitor groups** – drag and drop replaced by menu





Commerce 14 UI differences

- **Edit:** small changes.
- **Commerce manager** gone
- **Admin view** rebuilt and config moved here
- Some config **API-only**
 - Ex: catalog import/export, meta-data fields coming back soon.

Manage Payments

English svenska

Create

Search

<input type="checkbox"/>	Name	Is Active	Is Default	Ordering	Created	Modified	
<input type="checkbox"/>	ExchangePayment	False	False	2	20/04/2010, 0...	20/04/2010, 0...	⋮
<input type="checkbox"/>	Cash on delivery	True	False	1	17/02/2021, 0...	17/02/2021, 0...	⋮

What's the deal in DXP?

DXP environment

CMS 11, Commerce 13

.NET Framework 4.8

Windows Server / IIS

DXP environment

CMS 12, Commerce 14

.NET 6 or later

Linux Server / Kestrel



ANYONE UPGRADED YET???





MARK RUFFALO



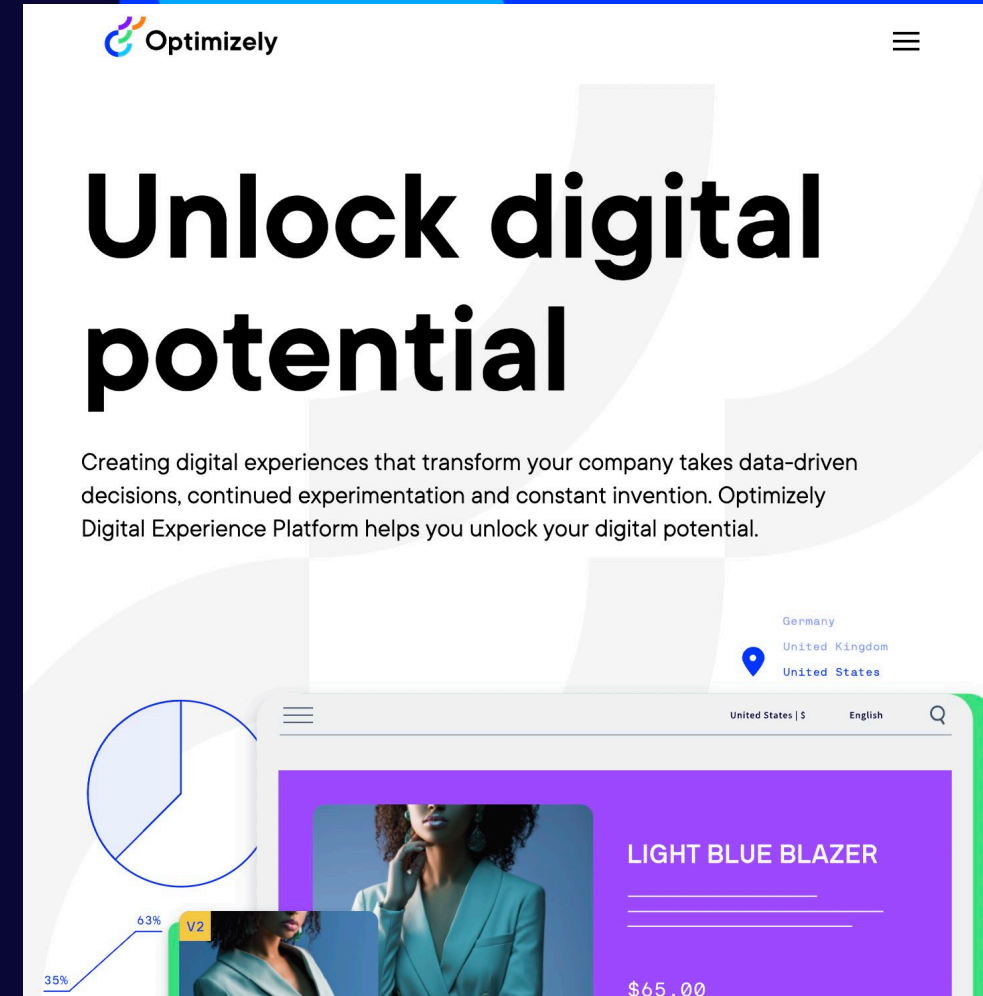
ED NORTON

**Who has done
this already?**

Optimizely.com

Before

- CMS 11
- .NET 4.8
- Windows in Azure
- 40 Editors
- 4 Developers
- 70+ Content Types
- 3.1 seconds time to interact



The screenshot shows the Optimizely website homepage. At the top left is the Optimizely logo, and at the top right is a hamburger menu icon. The main headline reads "Unlock digital potential" in large, bold, black font. Below the headline is a paragraph: "Creating digital experiences that transform your company takes data-driven decisions, continued experimentation and constant invention. Optimizely Digital Experience Platform helps you unlock your digital potential." In the top right corner, there are location and language selection options: "Germany", "United Kingdom", and "United States" (selected), along with "United States | \$" and "English". The main content area features a product card for a "LIGHT BLUE BLAZER" with a price of "\$65.00". To the left of the product card is a pie chart and a line graph. The pie chart shows a segment of 63% labeled "V2" and another segment of 35%. The line graph shows a data point of 35%.

About the upgrade

During - 2 months in total

- 1 week to upgrade code
- 3-4 weeks of QA
- Redesign also happened (took the most time)
- 1 month of optimization of code, feature enhancements
 - React JS to server side react
 - Vite instead of Webpack
 - Got rid of a few add-ons not needed anymore
- Pain-points
 - Translation Integration was not ready
 - Windows vs Linux



Results

After

- 1.3 seconds time to interactive
- Faster UI
- CMS Latest
- Linux
- 2 developers
- Better experience overall

 Optimizely



Unlimit Yourself



**What is the
secret sauce?**

(AKA: Why Upgrade?)

This is the Future

Take advantage

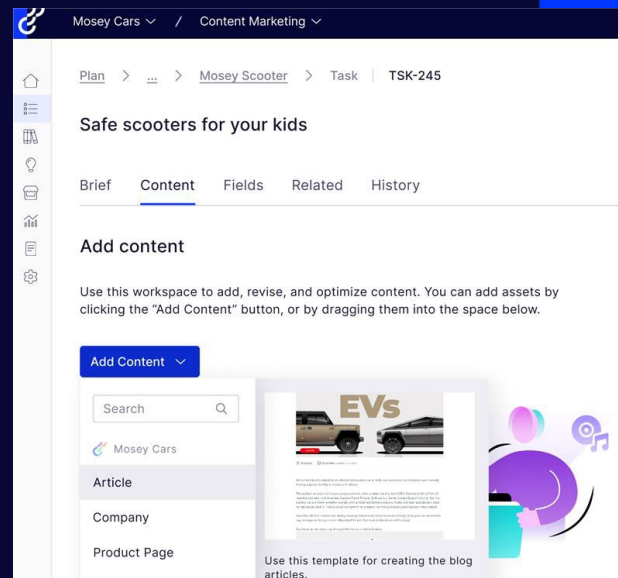
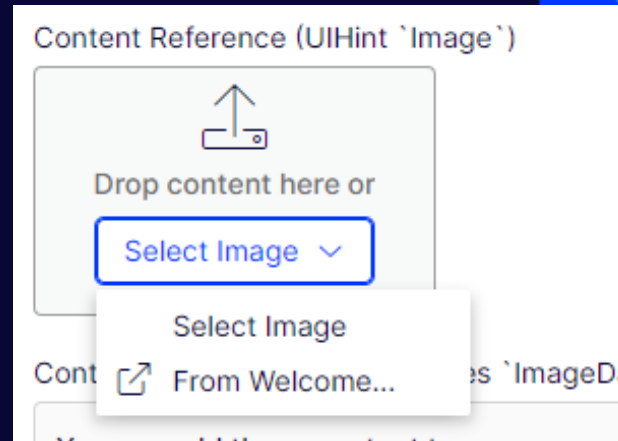
- Performance
- Developer experience
- CMS 12+ gets all the new features

Content Marketing Platform + DAM

Welcome (to the future)

- Use DAM assets in the CMS
 - ...via asset picker

- Structured Content



Headless?

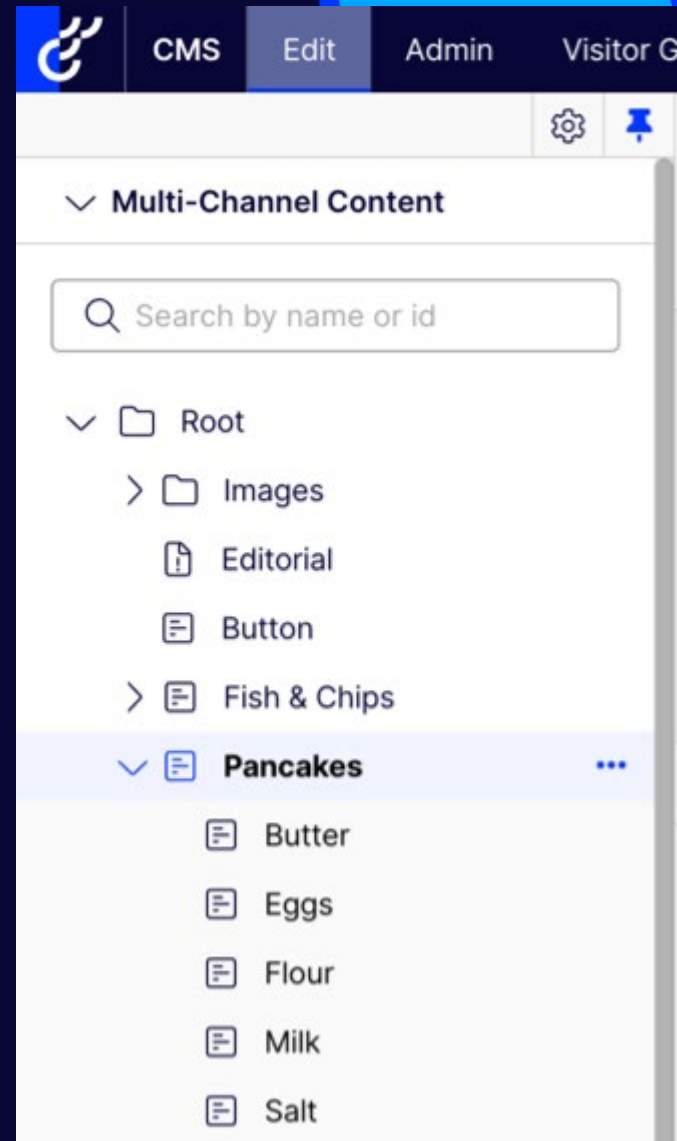
New REST APIs

- Content Definitions API
 - Manage content (and property) definitions without deployment
- Content Management API
 - Push content (or content changes) into the CMS

Headless Mode

AKA: Multi-Channel Content gadget

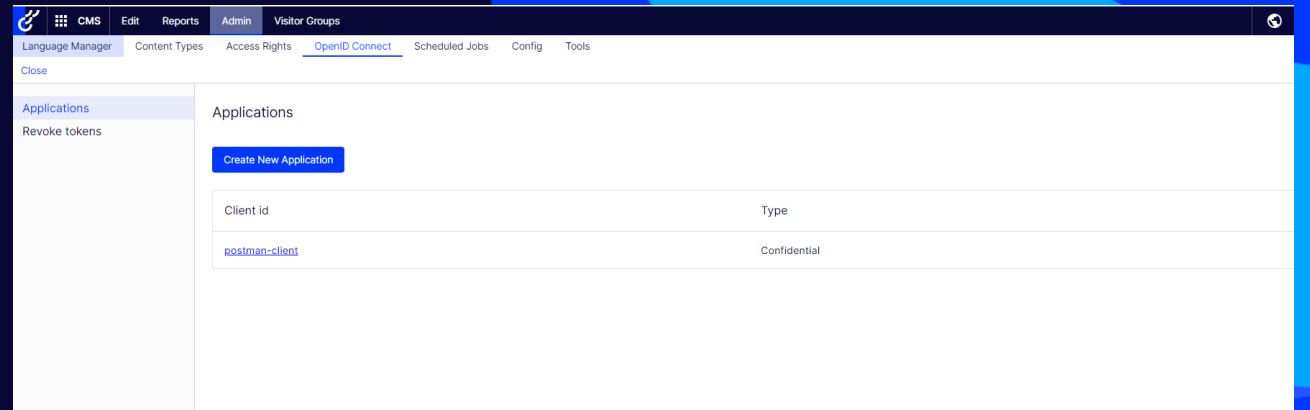
- Make it easier for editors working with multi-channel content



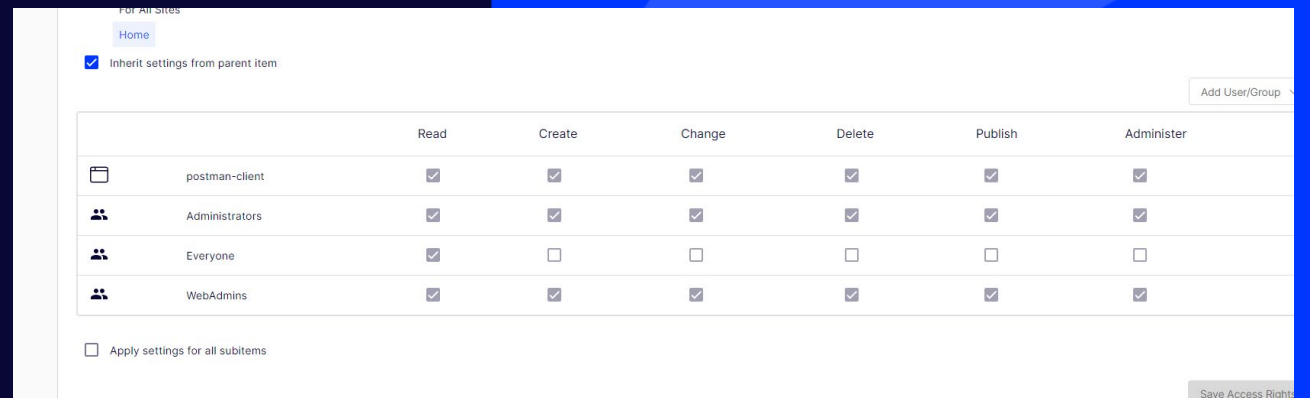
Application Management + OAuth

Application access to Content Cloud

- Managing applications
 - via UI or code



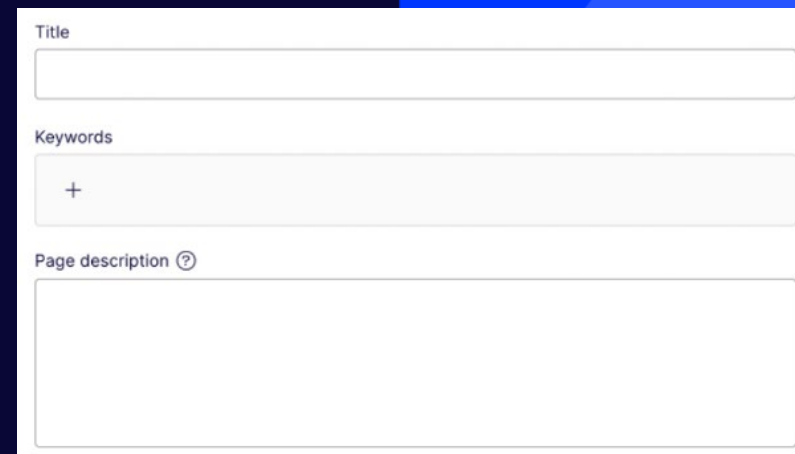
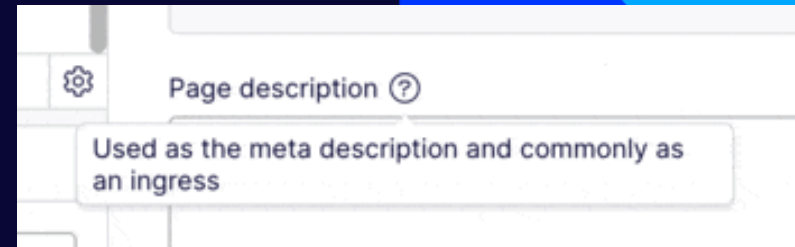
- Managing application access



Marketers, marketers, marketers

It's not just about the devs (sorry!)

- Help Text for content properties
- Display property labels above the input fields
- Increased and Aligned Property Width

A screenshot of a form with three input fields. The first field is labeled "Title" and is empty. The second field is labeled "Keywords" and contains a plus sign. The third field is labeled "Page description" with a question mark icon to its right and is empty.

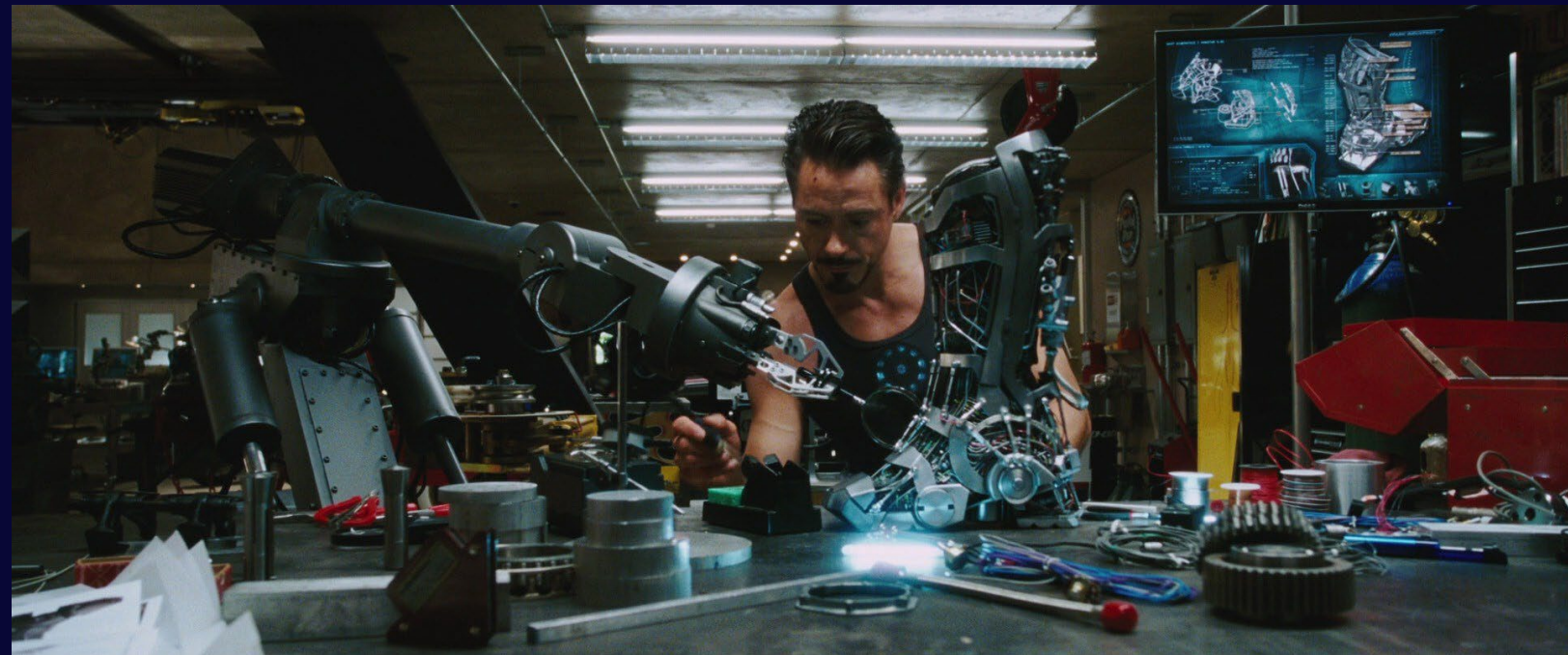
Want to see it in action?

Multiple examples available now

- Alloy
 - <https://github.com/epi-server/content-templates/tree/main/templates/Alloy.Mvc>
- Foundation Reference Site
 - <https://github.com/epi-server/foundation>
- Foundation React (headless)
 - <https://github.com/epi-server/foundation-spa-react/tree/Cms12>
- Music Festival (headless)
 - <https://github.com/epi-server/content-delivery-js-sdk/tree/master/samples/music-festival-vue-decoupled>
- Opticon Sample Site (headless)
 - Special release this week!



Developers will love this!



New APIs

Hot Reload

File Stream

Top Features of .NET

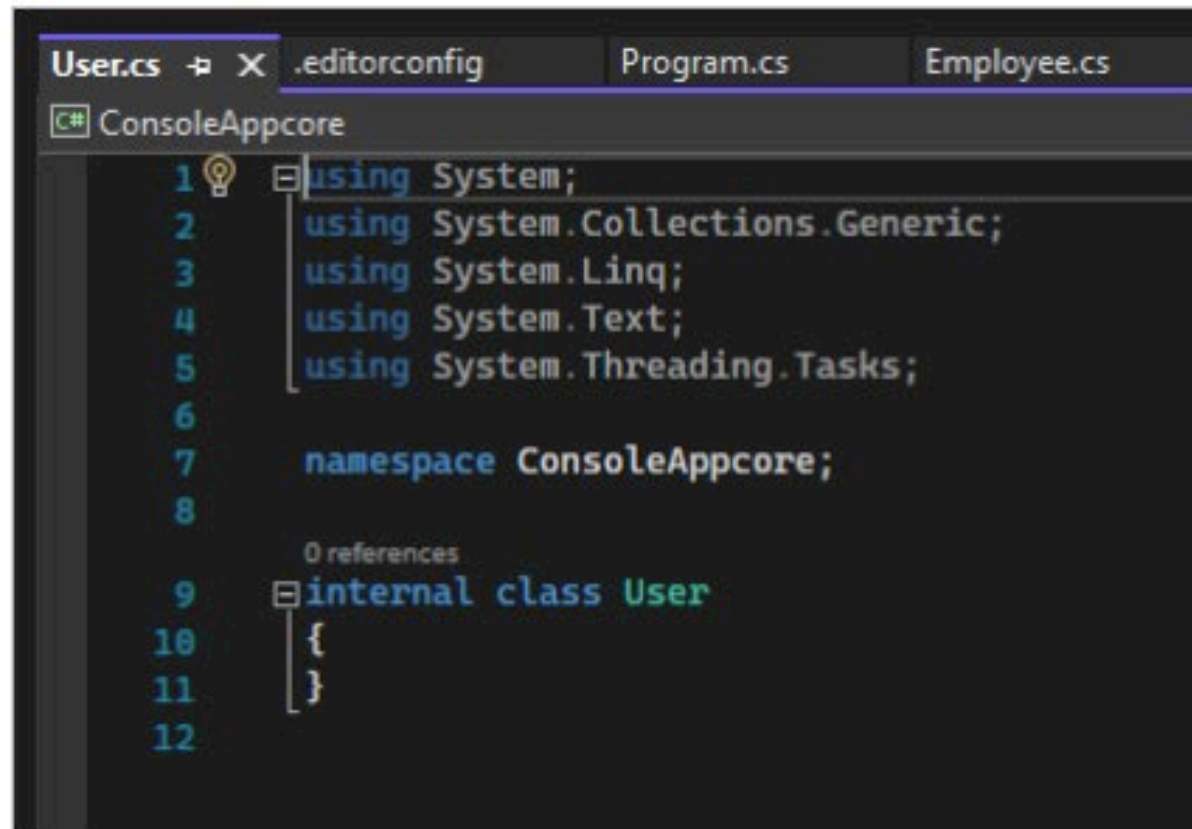
C# 10 Features

ARM64 Support

Crossgen2

Copy

```
namespace ConsoleAppcore;  
internal class User  
{  
}
```



The screenshot shows a Visual Studio code editor window with the following tabs: User.cs, .editorconfig, Program.cs, and Employee.cs. The active file is User.cs, which contains the following C# code:

```
1 using System;  
2 using System.Collections.Generic;  
3 using System.Linq;  
4 using System.Text;  
5 using System.Threading.Tasks;  
6  
7 namespace ConsoleAppcore;  
8  
9 internal class User  
10 {  
11 }  
12
```

The code is displayed with line numbers on the left. A lightbulb icon is visible next to line 1. The namespace and class name are highlighted in blue. The class name 'User' is highlighted in green. The code is indented with vertical bars. A '0 references' tooltip is visible above line 9. The editor has a dark theme.

```
global using System;
```

The screenshot displays three overlapping windows in Visual Studio illustrating the use of global using statements in C# 10. The top window shows the source file `User.Employee` with the following code:

```
1 using System;
2 using System.Collections.Generic;
3 using System.Linq;
4 using System.Text;
5 using System.Threading.Tasks;
6
7 namespace User
8 {
9     public class Employee
10    {
11    }
12 }
13
```

The middle window, titled `Program.cs`, shows the application's entry point:

```
1 global using User;
2
3 Employee employe = new Employee();
```

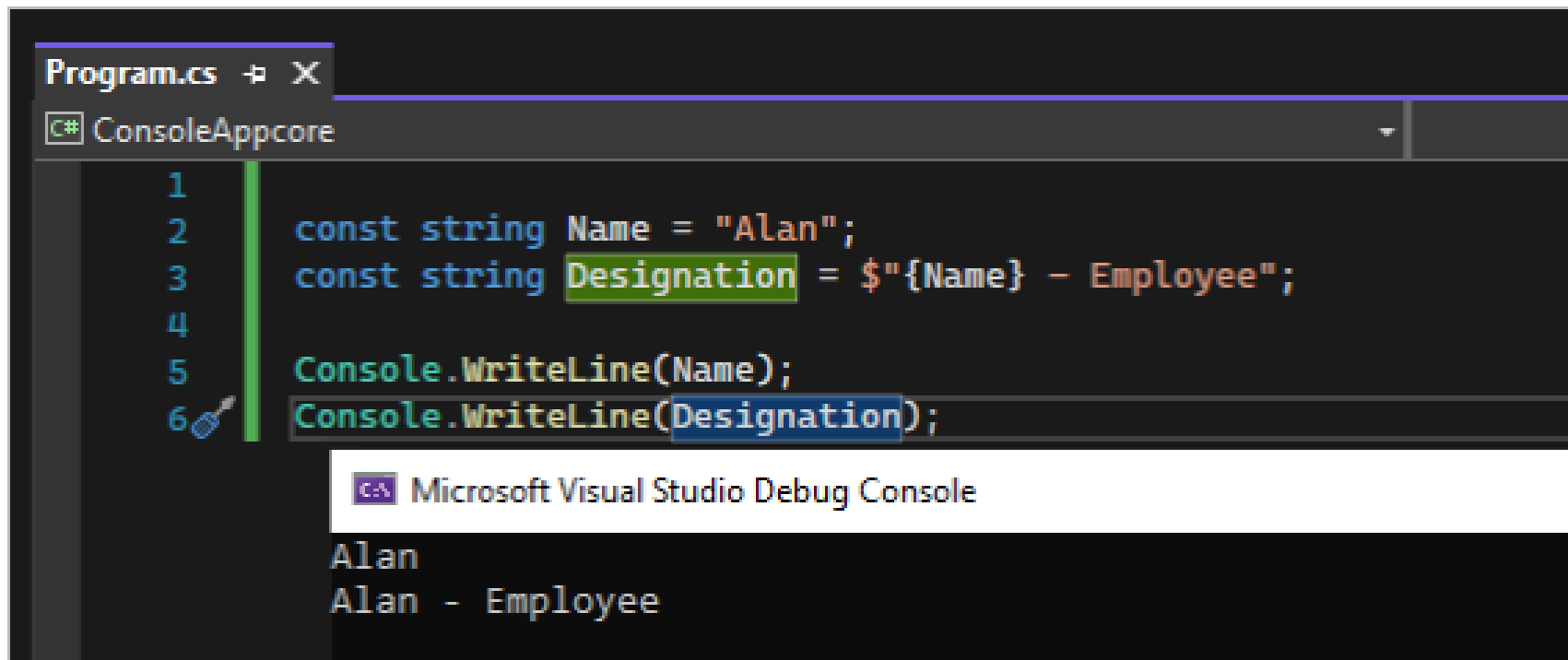
The bottom window, titled `Company.cs`, shows a class that uses the global using statement:

```
1
2 namespace ConsoleAppCore
3 {
4     public class Company
5     {
6         Employee employe = new Employee();
7     }
8 }
```

Global Using in C# 10

Copy

```
const string Name = "Alan";  
const string Designation = $"{Name} - Employee";
```



The screenshot shows a Visual Studio window with a C# file named Program.cs. The code defines two constant strings: Name (value: "Alan") and Designation (value: \$"{Name} - Employee"). The Designation variable is highlighted in green. Below the code, the Microsoft Visual Studio Debug Console shows the output of the program: "Alan" followed by "Alan - Employee".

```
Program.cs [X]  
C# ConsoleAppcore  
1  
2 const string Name = "Alan";  
3 const string Designation = $"{Name} - Employee";  
4  
5 Console.WriteLine(Name);  
6 Console.WriteLine(Designation);  
C# Microsoft Visual Studio Debug Console  
Alan  
Alan - Employee
```

C# 8

Copy

```
{ ParentProperty: { ChildProperty: Value } }
```

C# 10

Copy

```
{ ParentProperty.ChildProperty: Value }
```

```
var employee = new Employee();
employee.Name = "Employee";
employee.Location = new Location();
employee.Location.Country = "India";

if (employee is { Name: "Employee", Location.Country: "India" })
{
    Console.WriteLine(employee.Location.Country);
}

1 reference
public class Employee India
{
    2 references
    public string Name {h code 0.
    4 references
    public Location Location when debugging stops.
}

2 references
public class Location
{
    3 references
    public string Countr
```

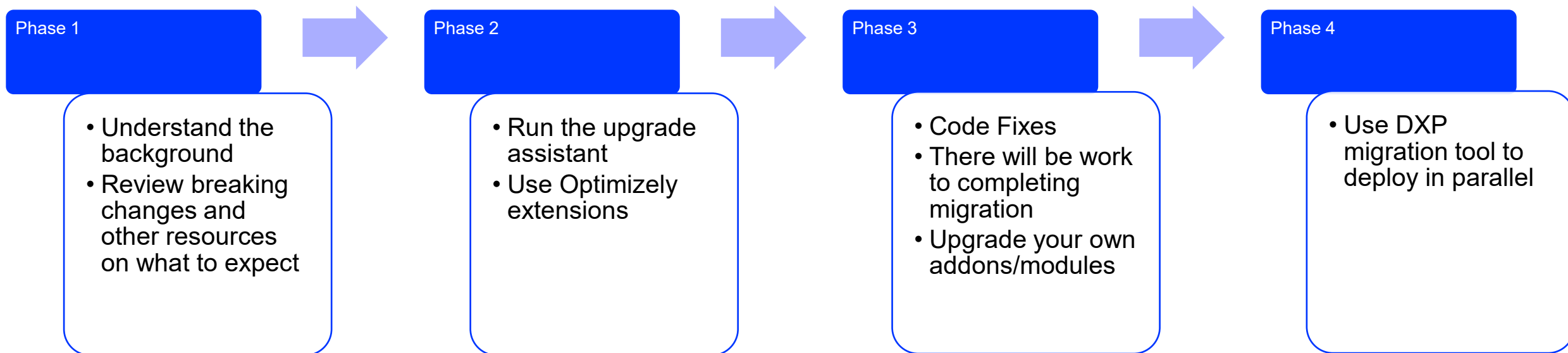
Microsoft Visual Studio Debug Console

C:\Users\Selva\source\repos\ConsoleAppcore\ConsoleAppcore\bi
To automatically close the console when debugging stops, ena
Press any key to close this window . . .





Migration Phases





Breaking changes & Add-ons

<https://docs.developers.optimizely.com/content-cloud/v11.0.0-content-cloud/docs/breaking-changes-in-content-cloud-cms-12>

<https://world.optimizely.com/net5/add-ons/>

<https://docs.developers.optimizely.com/integrations/v1.1.0-apps-and-integrations/docs/add-ons-platform-compatibility>

Content Cloud Products Guides Breaking changes in Content Cloud (CMS 12) Search

GETTING STARTED WITH CONTENT CLOUD
 Getting started with Content Cloud
 Setting up a development environment
 Installing Optimizely (CMS 11 and Commerce 13)
 Installing Optimizely updates (CMS 11 and Commerce 13)
 Installing a sample site
 Learning basic editing
 Creating a starter project
 Content Cloud documentation videos
 System requirements for Optimizely

CONTENT CLOUD LEARNING PATH
 Learning path
 Developer prerequisites
 What is a CMS?
 Technology stack
 Alloy demonstration templates
 Initial configuration
 Upgrading Optimizely
 Optimizely user interface
 Content model and views
 Creating and editing content
 Media support
 Page tree and routing
 Listings and navigation

Breaking changes in Content Cloud (CMS 12) Suggest Edits

This topic describes breaking changes for Optimizely Content Cloud in relation to version CMS 11, and the steps needed to update affected code.

To view the complete list of changes, see [the release notes feed](#).

Binary breaking changes do not necessarily require code changes but rather just a recompilation of the project. CMS 12 breaking changes to method signatures or to the behavior of methods, compared to the documented API in CMS 11, are described in this topic.

Version 12 targets .NET 5.0 so breaking changes between ASP.NET and .NET Framework 4.x to .NET 5.0 apply to Optimizely CMS projects also. For example, this includes ending support for writing templates that use WebForms or having WebForms views in MVC.

A breaking change may cause a component to fail. When a breaking change is made to a signature of a method, class, or interface, the former signature is often kept intact but set as obsolete and may cause a warning message in Visual Studio. To delay fixing the warning messages, you can disable Treat Warnings as Errors (right-click on your project > Properties > Build > set Treat Warnings as Errors to Specific warnings or None).

Classes that expose constructors that take dependencies are normally deleted without an obsolete warning in major releases, because the compiler provides information about what you need to change. Keeping those classes makes dependency injection complex because, over time, there would be multiple constructors to choose from that might overlap.

In each major version, obsolete methods are removed permanently to make sure that the API is kept clean and usable. If you could postpone fixing warning messages, you should make sure warning messages are fixed before upgrading to a major version. For CMS 12.0, many methods that were made obsolete in prior versions are now deleted.

NuGet packages

You should use the [Upgrade Assistant](#) to migrate from a CMS 11 project to CMS 12 because the tool manages most of the package updating.

TABLE OF CONTENTS

- NuGet packages
- Migration
- Dependency Injection
- Custom DI frameworks
- Service Registration
- ServiceLocator.Current
- Logging
- Routing
- Access checks
- Request language
- Routing extensions
- Extension methods
- Configuration
- Template Selection
- Plugin
- GuiPlugin
- PagePlugin
- Dashboard
- Reports
- VirtualPathProviders
- Security

Migrated Packages

Here is a list of packages that are owned and managed by Optimizely. Package status is either migrated (Released), currently being worked on (In Progress), or will not be migrated (Deprecated) to support .NET 5.

Sort By: Package ↑ Status ↑ Search:

ABTestingCriterion Status: Deprecated	EPIServer.AddOns.Helpers Status: Released	EPIServer.Azure Status: Released	EPIServer.Azure.Storage.Explorer Status: Released
EPIServer.Campaign.Commerce Status: Released	EPIServer.CloudPlatform.Cms Status: Released	EPIServer.CloudPlatform.Cms.Warmup Status: Released	EPIServer.CloudPlatform.Commerce Status: Released
EPIServer.CMS Status: Released	EPIServer.CMS.AspNet Status: Deprecated	EPIServer.CMS.AspNet.Core Status: Released	EPIServer.CMS.AspNet.Core.HtmlHelpers Status: Released
EPIServer.CMS.AspNet.Core.Migration Status: Released	EPIServer.CMS.AspNet.Core.Mvc Status: Released	EPIServer.CMS.AspNet.Core.Routing Status: Released	EPIServer.CMS.AspNet.Core.Templating Status: Released
EPIServer.CMS.Core Status: Released	EPIServer.CMS.TinyMce Status: Released	EPIServer.CMS.UI Status: Released	EPIServer.CMS.UI.Admin Status: Released
EPIServer.CMS.UI.AspNet.Identity Status: Released	EPIServer.CMS.UI.Core Status: Released	EPIServer.CMS.UI.Settings Status: Released	EPIServer.CMS.UI.Sources Status: Released

Tools for upgrading

- Upgrade assistant
 - Available on [GitHub](#)
- ASPNET projects to new .csproj style
- Change target framework
- Base class mapping
- Update packages
- Convert standards
- Replace IFindUIConfiguration with FindOptions

☰ README.md

Optimizely Upgrade Assistant Extensions

🔄 Continuous integration passing

This repository contains extensions to the Upgrade Assistant tool specific to Optimizely scenarios.

These extensions expand the upgrade-assistant tools's functionality to make Optimizely-specific changes during upgrade.

Installation

Install the latest version of the upgrade-assistant `dotnet tool install -g upgrade-assistant` or upgrade `dotnet tool update -g upgrade-assistant`

Grab the latest release from [here](#) and unzip the file to a location of your computer (ex C:\temp\epi.source.updater). Technically you should be able to point the zip file instead of extracting but there seems to be a bug in upgrade-assistant at the moment for that.

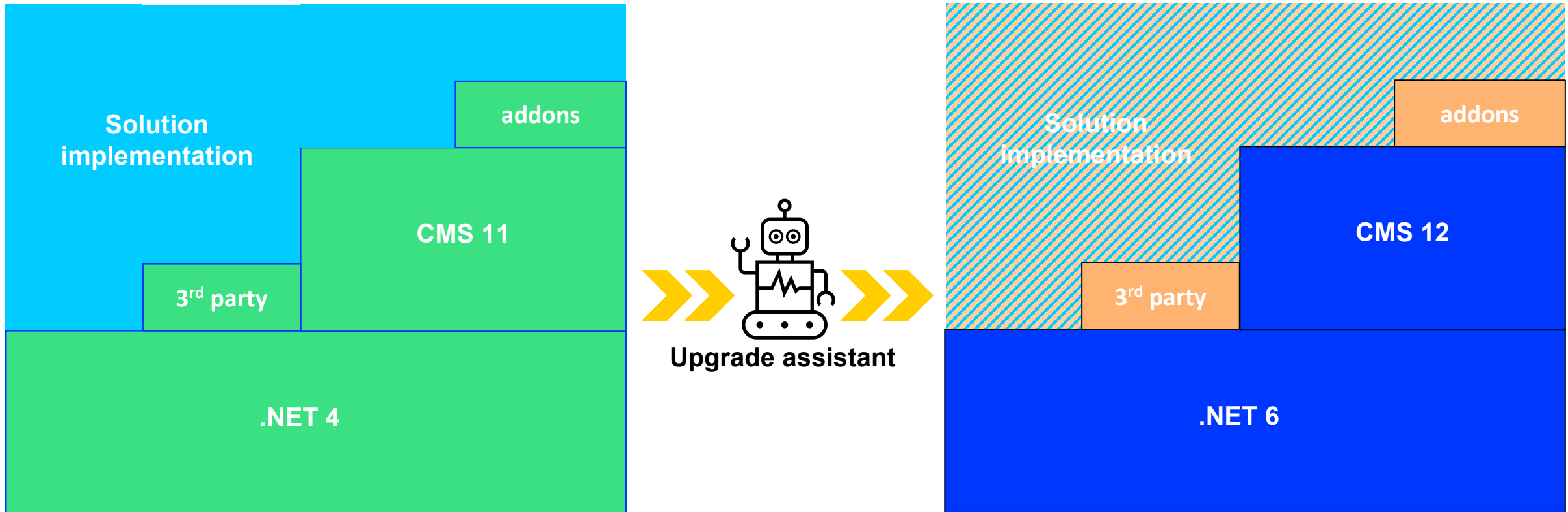
Using the extension

Some of the source code upgrade code analysers requires that packages referenced are available in selected target framework. So for now when CMS targets net5.0 the recommendation is to target net5.0 when converting as well (target framework can be set to net6.0 after conversion is completed). To achieve this first set environment variable like:

```
dotnet tool install -g upgrade-assistant  
  
upgrade-assistant upgrade {projectName}.csproj
```




Upgrade code



<https://world.optimizely.com/blogs/drew-null/dates/2022/5/upgrading-optimizely-cms-11-to-12-and-commerce-13-to-14/>

Tools for Upgrading

- Paas portal migration tool
- Parallel Linux environment
- Sync db, blob and access
- Early access available now!
- GA Release in Q2

Project Name: ellutmp1

Organization : [AnWa](#)

Plan: Digital Experience Cloud - Professional

Platform: Windows

[Deployments](#)

[Environments](#)

[Hostnames](#)

[Troubleshoot](#)

[API](#)

[.NET 5 Migration](#)



Migrate the application to .NET 5. New resources will be provisioned and migration to .NET 5 can be done in parallel with this project.

See [Documentation](#) for more information

[Start Migration](#)


[Continue](#)


[Abort Migration](#)




Preparing for Optimizely CMS 12 and Commerce 14

This Masterclass was designed for developers, architects and project managers that need to understand the technical implications of major Optimizely platform iterations. It will target all modern .NET 5 and later versions, including CMS 12, Commerce 14 and Search & Navigation 14. You will also learn how to prepare your current version to CMS 12.

 Scheduled course

 3 hours

 English

Presented by Scott Reed | Solution Architect and OMVP, Niteco and Mark Price | Technical Trainer, Optimizely

A list of breaking changes will show you where to expect future changes and which legacy features they can be replaced with. We will also discuss modern .NET techniques used by CMS 12 and Commerce 14, like the ASP.NET Core start-up process and configuration solutions to make future migrations go as smooth as possible.

Course agenda:

- Introduction to Optimizely CMS 12 and Commerce 14
- New skills and techniques for modern .NET
- Preparing CMS 11 solutions today for CMS 12, tomorrow
- Q&A

Materials: Slide deck, exercise book, code solutions and a migration check list.

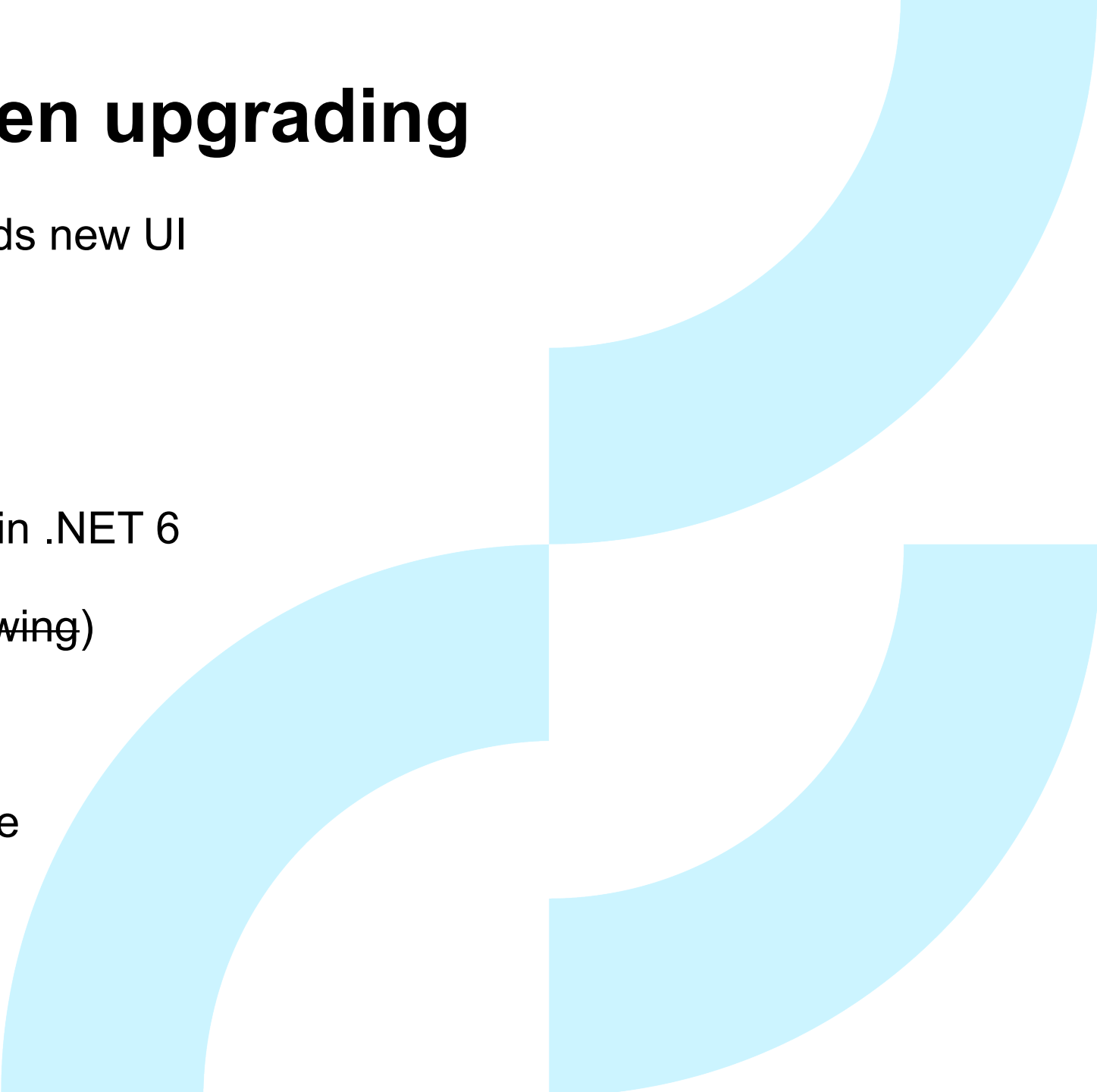
<https://www.optimizely.com/support/education/product/migrating-to-optimizely-cms-12-and-commerce-14/>

Things to consider when upgrading



<https://github.com/DrewNull/optimizely-cms12-upgrade>

Thing to consider when upgrading

- New admin UI → admin plugins needs new UI
 - Completely new UI in React
 - Commerce manager is gone
 - Content output cache does not exist in .NET 6
 - No more imageResizer (~~System.Drawing~~)
 - Check add-ons/modules used
 - DXP: ADE:s, Add application package
- 

Why haven't you upgraded yet?



Haven't had time...



Other things came up...



Do not want to be first...



Do not see enough value...



Why Upgrade?

Page Performance

Performance on page load speeds makes a huge difference in customer satisfaction and SEO.

Developers Developers Developers

Retaining and attracting best talent, finding new developers easier.

Marketer Satisfaction

Quick interface to work in that is modern and nice.

New Features

All the new stuff will come in CMS 12.





Competition

