

Managing Scripts and Styles in your Rich Text Areas

agility

Guide

HTML areas in the CMS support <script> and <style> tags. But, to what extent should you be using these? A successful CMS implementation should empower your editors and keep them focused on creating content rather than finicking with the presentation layer. They shouldn't be worrying about adding the right Styles and Scripts to make their content look right.

When is it Okay to Inject Scripts/Styles in your Rich Text Area?

There are few cases where this is recommended, but we also realize that there are times when you need to make quick changes without having access to the website source code or ability to do releases. These should be few and far between and should be converted to a Module at the earliest opportunity by a developer.

Side-Effects

Having JS and CSS in Rich Text Areas can have the following effects on your CMS implementation for your website:

- Increase in bugs and general website deficiencies
- Increase it time it takes to modify the content
- Increased site maintenance costs
- Only web-savvy editors can modify the content
- Potential for vulnerabilities with un-approved/un-tested JavaScript being added to the site
- Performance issues

Where Should CSS and JS be Managed?

If certain CSS and JS are essential for content to appear correctly, it should be added to your website code.
Agility has the concept of Modules, and they represent the functional components of a page. Developers encapsulate HTML, CSS, and JavaScript so that when a Module is added to a page, they can guarantee the right CSS and JavaScript are available to the page automatically. This is the responsibility of the developer.

Make it a Module

If you are finding yourself adding JS and CSS within a Rich Text Area in the CMS, then this code belongs in a Module. Modules have code that is rendered when they are requested by a page. That code is comprised of HTML that is rendered by a template engine (.chstml, .jsx, etc.)
When you move your static HTML from your Rich Text Area to a Module, you should break down the content into individual fields. Rather than one big HTML area, use a combination of fields to store the structured data.
Example fields include:

- Text
- Number
- Long Text
- Image
- File
- YouTube

Then, in code on your website, use a template engine (.cshtml, .jsx, etc.) to hand-craft your presentation layer which simply outputs the raw data from the CMS and the appropriate HTML, CSS, and JS required for the desired output.

Benefits of a Module:

- Can be re-used on any page in the CMS
- Can be included in dev ops pipeline for approvals and quality assurance (QA, UAT, and Production releases)
- Non-web-savvy editors can create pages with rich functionality simply by entering fields on a form
- Lower maintenance costs

Who can Create a Module?

If you can write HTML/CSS/JS, then you likely have the skills to be able to create a Module as well.

Tips:

- Avoid development bottlenecks by involving more people in your dev ops
- Setup your website code in a way that empowers frontend developers to be able to contribute
- Ensure you have a gated check-in setup in your source control so that code is reviewed before it is accepted
- Implement QA workflows so the code is tested before putting into production
- Deploy early and often

Preventing Scripts and Styles in Rich Text Areas

To ensure that your editors and developers are following best practices, consider implementing these ways to validate your content:

1. Use Content Workflow so only approved content (without <style> and <scripts>) can be published.
2. Use the onBeforeSave JavaScript hook within your Content/Module definitions input form script to validate your content before it is allowed to be saved.
3. Consider creating your own custom Rich Text Area field using Custom Field Types that does not allow editors to use <script> or <style>



agility

Agility CMS is a Content Management System (CMS) with built-in Hybrid components for better Editor Experience.

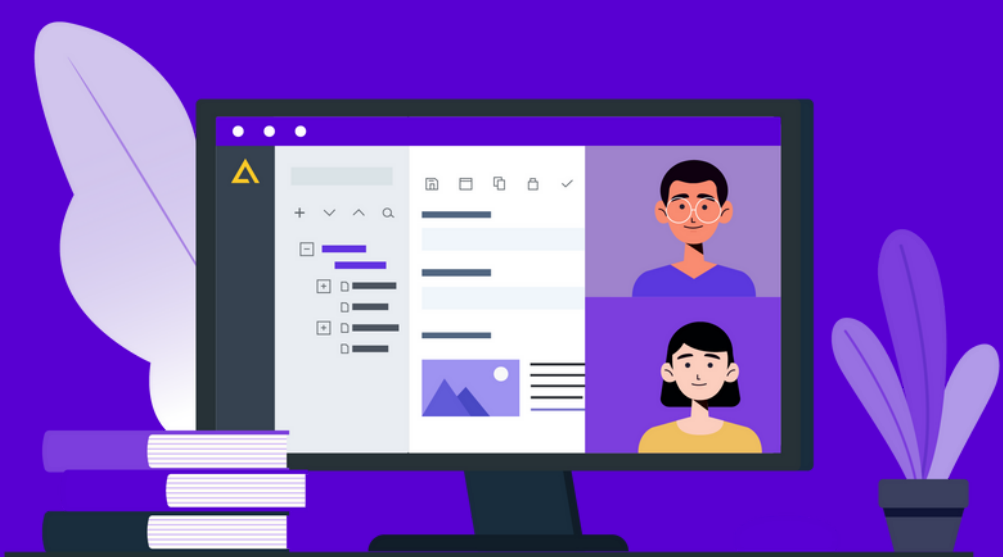
The platform adopts all the aspect of Headless CMS that makes an omnichannel approach easy, with the understanding that Page Management is of utmost importance in any website project.

Agility is proprietary CMS with full Support and Security that gives creative freedom to both Developers and Marketers and eliminates friction between the teams.

Developers have full freedom to use any coding language and any framework and code and define custom Page Templates and Module Definitions.

These pre-defined custom Modules and Page Templates are available to the Marketing Teams to effortlessly create a variety of pages to meet their fast-changing needs and market requirements. Agility CMS believes in Content First Approach and heavily invests in helping clients to build future-proof Content Architecture.

Book a Demo >



Agility CMS Differentiators

- *It is a modern Enterprise-grade Composable DXP with flexible Headless approach in its heart.*
- *Unlike many other Headless CMS, Agility CMS also offers tools for Enterprise teams to effortlessly manage content delivery to websites and apps like Page Management, Sitemaps, Content Modelling.*
- *With Page Management built-in, developers don't have to build an entirely new way to manage pages and routing. At the same time, marketers can easily edit content across a variety of channels without any coding experience.*
- *Agility CMS provides enterprise-level security and support, it is SOC2 compliant.*
- *Agility CMS is Jamstack friendly and allows to start quickly with your favorite frameworks and deploy with a few clicks.*

Details



License Model: *Agility CMS is a multi-tenant SaaS platform, and hosted on MicrosoftAzure Cloud*



Price: *Starts with FREE Developer Edition. Paid plans have extensive TRIAL.* *Learn more.*



Key Components: *Page Management, Content Relationships, Content Modelling, POS, Search, User Generated Content, Personalization, Digital Asset Management. Unlimited content models, Locales, API calls,*



Upgrades: *Automatic, monthly platform updates.*

agility