Sage 300 Web Screens SDK

JavaScript Minification

October 2017

The MIT License (MIT)

Copyright © 2017 The Sage Group plc or its licensors. All rights reserved.

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the “Software”), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED “AS IS”, WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

Contents

[1. Overview 4](#_Toc497118708)

[2. Environment Setup 5](#_Toc497118709)

[3. Using WebGrease 6](#_Toc497118710)

1. Overview

JavaScript Minification is a process to compress the size of JavaScript files by stripping the contents of all formatting, whitespace, comments, and other unnecessary data. In this optimization process, variable and function names are changed and shortened. Since web browsers do not care about readability, using minified JavaScript files can lead to performance improvements and increased code obfuscation.

Starting with Sage 300 2018, Sage 300 deploys minified JavaScript for the Sage 300 Web Screens.

With this change, the ***RegisterBundles*** routine of the ***BundleRegistration*** class in the Sage Web projects were modified to use ***Bundle*** instead of ***ScriptBundle***. This is because *ScriptBundle* represents a bundle and will minify the bundle before it is packaged. Performing Minification on already minified files may lead to unexpected behavior.

Therefore, partners should use *Bundle* and not *ScriptBundle* in their registration routine if Minification is performed.

This does not apply to Customization JavaScript files as those files are not minified.

[WebGrease](https://www.nuget.org/packages/WebGrease/) is the tool selected to perform this Minification. In the following tutorial, we will set up an environment and minify several JavaScript files. By following this procedure, partners can minify their code before deploying to a customer.

1. Environment Setup

* Create a folder to work within
* Copy into it the following files:
  + WG.exe
    - Available in your solution’s “**…packages\WebGrease.1.6.0\tools**” folder or via Package Manager download at **PM> Install-Package WebGrease -Version 1.6.0**
  + WebGrease.dll
    - Available in your solution’s “**…packages\WebGrease.1.6.0\lib**” folder
  + Antlr3.Runtime.dll
    - Available in your solution’s “**…packages\Antlr.3.5.0.2\lib**” folder
  + Newtonsoft.Json.dll
    - Available in your solution’s “**…packages\Newtonsoft.Json.6.0.8\lib\net45**” folder

1. Using WebGrease

Open the command line and navigate to the directory created in the previous step as the working directory.

You can run WG.exe without any parameters to see an exhaustive list of options available.

However, the general format for minifying files will be:

* WG.exe -m -in:*<input file/folder>* -out:*<output file/folder>*
  + -m is for minification
  + -in: allows you to specify a file or a folder as input
  + -out: allows you to specify a file or a folder as output

For instance, **WG.exe -m -in:in -out:out** will minify all JavaScript files located in the \in folder and output them to the \out folder.