Table of Contents

ReadMe

This readme contains basic information about **GrapeCity Documents for Word**, also referred to as **GcWord**.

Release Notes

Refer to the following release notes for the major releases of the product.

For details about latest hotfixes, see the nuget page.

Version 2.2.0.310

Breaking Changes

This version of the product has the following breaking changes.

- The type of Document property has been changed from GcWordDocument to DocumentBase on the following classes: ContentObject, ContentRange, RangeBase, Style, StyleCollection, ListTemplateCollection.
- The ImageData(GcWordDocument document, bool isPictureBullet) constructor has been changed to ImageData(DocumentBase document, bool isPictureBullet). This means that it now accepts DocumentBase class in document argument instead of the GcWordDocument class.
- Earlier, theme colors could be modified directly via GcWordDocument.Theme. But now, theme colors cannot be modified directly via GcWordDocument.Theme. Instead, users need to use Settings to modify theme colors now.
- The Theme.ColorScheme[ThemeColorId] indexer has been removed. Instead, users can use the new ThemeColor Settings.GetThemeColor(ThemeColorId) method.
- The ColorScheme.RemapColor(ThemeColorId, ThemeColorSchemeId) method has been removed. Instead, users can use the new method - Settings.RemapColor(ThemeColorId, ThemeColorSchemeId).
- Sections in GcWord OM are not objects but section type breaks (this provides consistency). Now, Section class does not inherit the ContentObject class.
- The SectionCollection class is inherited from ContentCollection<Section> instead of ContentObjectCollection class.
- Removed Section.Guid and Section.ParentContent properties.
- Removed SectionCollection.Add(), SectionCollection.Insert() and Section.Split() methods. Instead of these methods, users can use the new Paragraph.AddSectionBreak() method.

Changes From the Previous Release

This version of the product has the following change.

- Range() and CompareLocationWith(Range range) methods now accept RangeBase class in range argument instead of the Range class.

New Features and Improvements

The following features have been added with this version of the product.

New classes:

- DocumentBase class - Represents the base class for GcWordDocument and GlossaryDocument classes.
- GlossaryDocument - Represents a supplementary document storage which stores the definition and content to be carried with the document for future insertion and/or use, but which should not be visible within contents of the main document story.
- FormattedMark class - Specifies the set of properties applied to the special content mark
- BuildingBlockCollection class - Represents a collection of building blocks.
- BuildingBlock class - Represents a building block in a document.
- Category class - Specifies the categorization for a building block. This categorization should not imply any behaviors around the building block, and is only used to organize set of building blocks within an application or user interface i.e. to disambiguate between two building blocks with the same entry name.
- CustomXmlPart class - Represents a custom XML data storage in the document.
- CustomXmlPartCollection class - Represents a collection of CustomXmlPart objects in the document.
- LastRenderedPageBreak class - Represents the position delimited the end of a page when this document was last saved by an application which paginates its content.
- ContentControl class - Represents an individual content control. Content controls are bound and potentially labelled regions in a document that serve as containers for specific types of content. Individual content controls may contain contents such as dates, lists, or paragraphs of formatted text.
- ContentControlCollection class - Represents a collection of ContentControl items.
- ControlContent class - Provides access to the content of a ContentControl.
- ContentControlEndMark class - Specifies set of properties applied to the mark present to delimit the end of control contents.
- CheckBoxSymbol class - Specifies a symbol to be used for a checkbox state.
- DropDownItem class - Specifies a single list item within the content control. Each list item should be displayed in the list for the content control (if a user interface is present).
- DropDownItemCollection class - Represents a list of DropDownItem objects.
- XmlMapping class - Specifies the information to be used to establish a mapping between a content control and an XML node stored within a Custom XML Data part in the document.

New enumerations:

- BuildingBlockGallery enum - Specifies a building block gallery.
- BuildingBlockType enum - Specifies the type of a building block.
- BuildingBlockInsertOptions enum - Specifies how a building block is inserted into a document.
- ContentControlType enum - Specifies type of the content control.
- ContentControlLevel enum - Specifies layout level of the content control.
- ContentControlAppearance enum - Specifies appearance of the content control.
- CalendarType enum - Specifies the calendar type.
- DateStorageFormat enum - Specifies the date translation to be applied to the date content control.
- WebExtensionRelationship enum - Specifies relationship between a content control and an Office Web Extension.

New members:

- Cell CellCollection.Insert(string text, InsertLocation location) method - Inserts a Cell into this collection at the specified location.
- Row RowCollection.Insert(string[] texts, InsertLocation location) method - Inserts a Row into this collection at the specified location.

- BodyType.BuildingBlock property - Represents the body of a building block.
- List<string> Settings.AttachedXmlSchemas - Gets the list of custom XML schema whose target namespace is associated with this document when it is loaded, if such a schema is available to the hosting application. Applications can also load and utilize any additional schemas as well as those explicitly mentioned here. These custom XML schemas can then be used to validate the structure of the CustomXmlPart.XmlDocument in the document etc.
- ContentControlCollection RangeBase.ContentControls property - Gets the collection of content controls included in this range.
- ContentControl Range.ParentContentControl property - Gets the parent content control where the range content is stored.
- Added ImageData.ContentType property - Gets the image content type.
- Added Marker.StartSection property - Gets the Section which starts in this marker.
- Added Marker.EndSection property - Gets the Section which ends in this marker.
- Added Paragraph.AddSectionBreak(SectionStart) method - Breaks the parent section right after this paragraph.
- Added the GcWordDocument.GlossaryDocument property - Gets the glossary document.
- Added the GcWordDocument.CustomXmlParts property - Gets the collection of CustomXmlPart objects.

Improvements:

- Improved performance when calculating values of derived properties.
- Improved Word to PDF export.

Bug Fixes

The following issues have been resolved since the last release.

- Fixed the incorrect saving issue of background color shading. Now, MS Word doesn't show the message "not enough memory to update the display" while loading the saved document.
- Now, no wrong exception messages are shown when users apply a wrong style type to a content object.

Version 2.1.0.260

Breaking Changes

- Renamed ThemeColor enum to ThemeColorId.
- Changed default value for WordColor.ThemeShade from 0 to 255.
- Changed default value for WordColor.ThemeTint from 0 to 255.

New Features and Improvements

- Added EastAsianLayout class which specifies any East Asian typography settings which shall be applied to the contents of a run.
- Added EastAsianTypography class which provides options for East Asian typography.
- Added HyphenationOptions class which allows to configure document hyphenation options.
- Added ColorScheme class which represents the color scheme of a Microsoft Office theme.
- Added ThemeColor class which represents a color in the color scheme of a theme.
- Added ThemeColorSchemeId enum which specifies the theme colors for document themes.
- Added ViewType enum which provides possible values for the view mode in application.
- Added ZoomType enum which provides possible values for how large or small the document appears on the screen in the application.
- Added ViewOptions class which provides various options that control how a document is shown in application.
- Filled CompatibilityOptions class which contains compatibility options (previously it contained only CompatibiltyMode property).
- Added FontBase.EastAsianLayout property that gets East Asian typography settings which shall be applied to

the contents of the run.
- Added FontBase.FitTextWidth property that gets or sets that the contents of the run which shall not be automatically displayed based on the width of its contents, rather its contents shall be resized to fit the width specified by the this property.
- Added FontBase.FitTextId property that gets or sets a unique ID which shall be used to link multiple contiguous runs with specified "FitTextWidth" property to each other to ensure that their contents are correctly merged into the specified width in the document.
- Added Settings.HyphenationOptions property which provides access to document hyphenation options.
- Added Settings.EastAsianTypography property which provides access to East Asian typography options.
- Added Settings.ViewOptions which provides various options that control how a document is shown in application.
- Added Theme.ColorScheme which gets a set of colors which are referred to as a color scheme.
- Improved FontBase.Name property behavior - If HintType is FontHintType.ComplexScript returns NameBi", if HintType is FontHintType.EastAsia returns NameFarEast.
- Improved performance of creating a new GcWordDocument instance more than 4 times.
- Allowed to open a DOCX while it is open in MS Word (use ReadWrite share mode instead of Read in GcWordDocument.Load()).

Bug Fixes

- Fixed the issue where page color disappears after export.
- Fixed the issue with saving picture bullets for built-in list templates.

Installation

Setting up an application

GcWord references are available through NuGet, a Visual Studio extension that adds the required libraries and references to your project automatically. To work with GcWord, you need to have following references in your application:

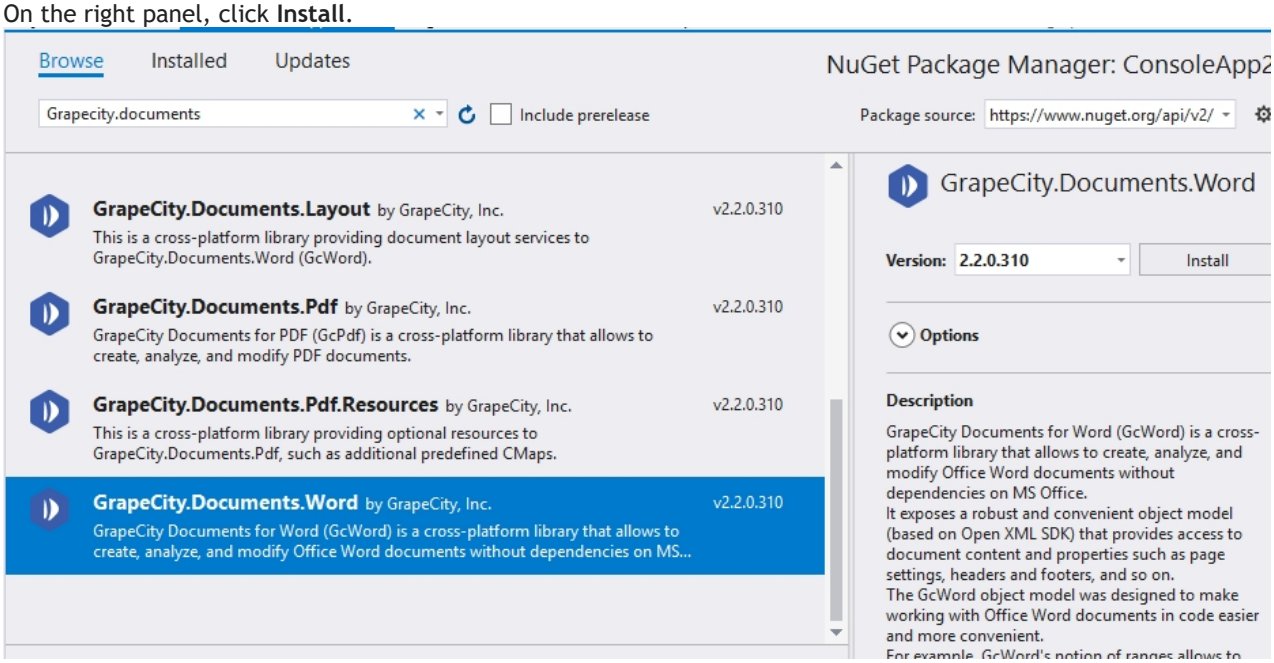
Reference	Purpose
GrapeCity.Documents.Word	To use GcWord in an application, you need to reference (install) just the GrapeCity.Documents.Word package. It will pull in the required infrastructure packages.
GrapeCity.Documents.Layout	To enable saving Word documents to PDF, install the GrapeCity.Documents.Layout package (GcLayout for short). It provides extension methods allowing to save GcWordDocument as PDF.
GrapeCity.Documents.Imaging	For image handling, you need to reference (install) the GrapeCity.Documents.Imaging package.
GrapeCity.Documents.Common	GrapeCity.Documents.Common is an infrastructure package used by other packages. You do not need to reference it directly.
GrapeCity.Documents.Common.Windows	On a Windows system, you can optionally install GrapeCity.Documents.Common.Windows. It provides support for font linking specified in the Windows registry, and access to native Windows imaging APIs, improving performance and adding some features (e.g. TIFF support).
GrapeCity.Documents.DX.Windows	GrapeCity.Documents.DX.Windows is an infrastructure package used by GrapeCity.Documents.Common.Windows. You do not need to reference it directly.

Add reference to GcWord in your application from nuget.org

In order to use GcWord in a .NET Core, ASP.NET Core, .NET Framework application (any target that supports .NET Standard 2.0), install the NuGet packages in your application using the following steps:

Visual Studio for Windows

1. Open Visual Studio for Windows.
2. Create any application (any target that supports .NET Standard 2.0).
3. Right-click the project in Solution Explorer and choose **Manage NuGet Packages**.
4. In the **Package source** on top right, select **nuget.org**.
5. Click **Browse** tab on top left and search for "Grapecity.Documents".
6. On the left panel, select **GrapeCity.Documents.Word**
7. On the right panel, click **Install**.



8. In the **Preview Changes** dialog, click **OK** and choose **I Accept** in the next screen.

This will add required references of the package to your application.

Visual Studio for Mac

1. Open Visual Studio for MAC.
2. Create any application (any target that supports .NET Standard 2.0).
3. In tree view on the left, right-click **Dependencies** and choose **Add Packages**.
4. In the Search panel, type "GrapeCity.Documents".
5. From the list of packages displayed in the left panel, select **GrapeCity.Documents.Word** and click **Add Packages**.
6. Click **Accept**.

This will automatically add references of the package and its dependencies to your application.

Visual Studio Code for Linux

1. Open Visual Studio Code.
2. Install **Nuget Package Manager** from **Extensions**.
3. Create a folder "MyApp" in your **Home** folder.
4. In the Terminal in Visual Studio Code, type "`cd MyApp`"
5. Type command "`dotnet new console`"
   Observe: This creates a .NETCore application with MyApp.csproj file and Program.cs.
6. Press **Ctrl+P**. A command line opens at the top.
7. Type command: ">"
   Observe: "**Nuget Package Manager: Add Package**" option appears.
8. Click the above option.
9. Type "**Grapecity**" and press Enter.
   Observe: GrapeCity packages get displayed in the dropdown.
10. Choose **GrapeCity.Documents.Word**.
11. Type following command in the Terminal window: "`dotnet restore`"

This will add references of the package to your application.

Product Requirements

GcWord system requirements, depending upon the framework you are using to create an application, are:

- Our packages include two targets, NET Standard 2.0 and .NET Framework 4.6.1. In order to use them, your application needs to target either of the following:
  - .NET Core 2.0 or later
  - .NET Framework 4.6.1 or later
- Visual Studio 2015+/Visual Studio for MAC/Visual Studio Code for Linux

For OS versions supported in .NET Core 2.0+, see .NET Core 2.0+ - Supported OS versions.

License Information

Types of Licenses

GrapeCity Documents for Word supports the following types of license:

- **Unlicensed**
- **Evaluation License**
- **Licensed**

Unlicensed

After downloading, the product works in unlicensed mode. The following limitations are imposed when the product is used without license:

- Only 200 paragraphs in a Word file can be saved for analyzing.
- On saving the Word file, a text is displayed on the beginning of the first page of that file:
  '*Created with unlicensed copy of GrapeCity Word. The document is limited to 200 paragraphs. Contact us.sales@grapecity.com to get your 30-day evaluation key.*'

> **Note:** Please note that the above text is inserted as the first paragraph into the document when it is saved, which changes the indices of other paragraphs in the saved file (but not in the current object model).

In case of PDF export, following page header is displayed on all the pages of the PDF file.

'Created with unlicensed copy of GrapeCity Word. Contact us.sales@grapecity.com to get your 30-day evaluation key.'

Evaluation License

GcWord evaluation license is available to users for 30 days to evaluate the product. If you want to evaluate the product, you can ask for the evaluation license key by sending an email to us.sales@grapecity.com.

The evaluation version has an expiration date that is determined when an evaluation key is generated. After applying the evaluation license key, you can use the complete product until the license expiry date.

After the expiry date, the product works in unlicensed mode with the above mentioned limitations.

In such case, following watermark is displayed in the Word file:
'Created with expired evaluation copy of GrapeCity Word. The document is limited to 200 paragraphs. Contact us.sales@grapecity.com to purchase license.'

> **Note:** Please note that the above text is inserted as the first paragraph into the document when it is saved, which changes the indices of other paragraphs in the saved file (but not in the current object model).

On exporting a Word document to a PDF, following page header is displayed on all the pages of the PDF file:
'Created with expired evaluation copy of GrapeCity Word. Contact us.sales@grapecity.com to purchase license.'

Licensed

GcWord production license is issued at the time of purchase of the product. If you have a production license, you can access all the features of GcWord without any limitations.

Apply License

To apply evaluation/production license in GcWord, the long string key needs to be copied to code in one of the following two ways.

- Pass it as an argument to the GcWordDocument's ctor:
  `var doc = new GcWordDocument("key")`
  This licenses the instance being created.

- Call a static method on GcWordDocument:
  `GcWordDocument.SetLicenseKey("key");`
  This licenses all the instances while the program is running.

Redistribution

In order to distribute the application, make sure you meet the installation criteria specified in the Product Requirements page in this documentation. Further, you also need to have a valid Distribution License to successfully distribute the application.

> GcWord makes it easy to deploy your application to your local servers or cloud offerings such as Azure.

For more information about Distribution License, contact our Sales department using one of these methods:

World Wide Web site	https://www.grapecity.com/
E-mail	us.sales@grapecity.com
Phone	1.800.858.2739 or 412.681.4343

Fax	(412) 681-4384

Viewing Help Content

For detailed information on Documents for Word, please refer online documentation.

Technical Support

If you have a technical question about this product, consult the following source:

- Product forum: https://www.grapecity.com/forums
- Email: us.sales@grapecity.com

Contacting Sales

If you would like to find out more about our products, contact our Sales department using one of these methods:

World Wide Web site	https://www.grapecity.com/
E-mail	us.sales@grapecity.com
Phone	(800) 858-2739 or (412) 681-4343 outside the U.S.A.
Fax	(412) 681-4384

End-User License Agreement

The GrapeCity licensing information, including the GrapeCity end-user license agreement, frequently asked licensing questions, and the GrapeCity licensing model, is available online. For detailed information on licensing, refer GrapeCity Licensing. For GrapeCity end-user license agreement, refer End-User License Agreement For GrapeCity Software.