Basic Concepts

Introduction

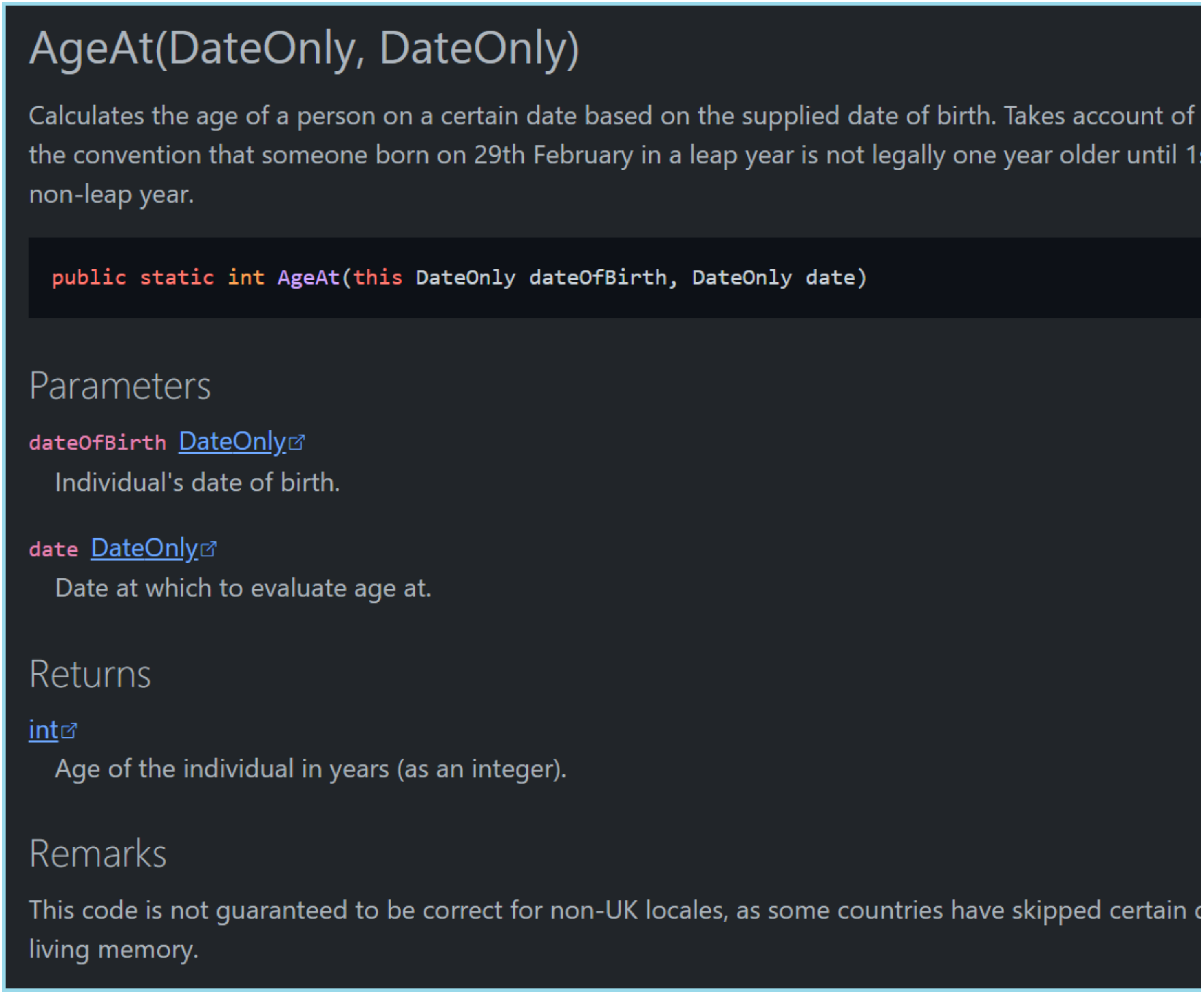
Docfx is a powerful tool but easy to use for most regular use cases, once you understand the basic concepts.

Docfx can be used as a static site generator, but the real value of the tool is in bringing together static documentation pages and .NET AI
relies on the long-established XML comment syntax for C# (and similarly for VB). For example, the following C# code:

```
/// <summary>
/// Calculates the age of a person on a certain date based on the supplied date of birth.  Takes account of leap years,
/// using the convention that someone born on 29th February in a leap year is not legally one year older until 1st March
/// of a non-leap year.
/// </summary>
/// <param name="dateOfBirth">Individual's date of birth.</param>
/// <param name="date">Date at which to evaluate age at.</param>
/// <returns>Age of the individual in years (as an integer).</returns>
/// <remarks>This code is not guaranteed to be correct for non-UK locales, as some countries have skipped certain dates
/// within living memory.</remarks>
public static int AgeAt(this DateOnly dateOfBirth, DateOnly date)
{
    int age = date.Year - dateOfBirth.Year;

    return dateOfBirth > date.AddYears(-age) ? --age : age;
}
```

can be used to generate output like this:


AgeAt(DateOnly, DateOnly)

Calculates the age of a person on a certain date based on the supplied date of birth. Takes account of
the convention that someone born on 29th February in a leap year is not legally one year older until 1
non-leap year.

```
public static int AgeAt(this DateOnly dateOfBirth, DateOnly date)
```

Parameters

`dateOfBirth` DateOnly
Individual's date of birth.

`date` DateOnly
Date at which to evaluate age at.

Returns

int
Age of the individual in years (as an integer).

Remarks

This code is not guaranteed to be correct for non-UK locales, as some countries have skipped certain
living memory.

Static documentation pages are prepared using Markdown (slightly enhanced to support specific features). Markdown content can also b

Once the API documentation has been parsed from the source code, it is compiled along with the Markdown content into a set of HTML p

Docfx is a command-line tool that can be invoked directly, or as a .NET Core CLI tool using the `dotnet` command, but it can also be invoke
`docfx.json` which has sections for different parts of the build process.

Consuming .NET projects

The most common use case for processing .NET projects is to specify one or more .csproj files in the `docfx.json` file:

```
{
  "metadata": [
    {
      "src": [
        {
          "files": [
            "src/MyProject.Abc/*.csproj",
            "src/MyProject.Xyz/*.csproj"
          ],
          "src": "path/to/csprojs"
        }
      ],
      "dest": "api"
    }
  ],
  //...
}
```

Although Docfx can build a documentation website in one step, it's helpful to understand the separate steps the tool uses to generate its

The first step is called the ***metadata*** step and can be completed using the following command line:

```
docfx metadata path/to/docfx.json
```

This command reads all the source files specified by the projects listed in `docfx.json` and searches for XML documentation entries. Note t output of this step is a set of YAML files that are stored in the `dest` folder specified in `docfx.json`.

Here's an example of the (partial) output from the above code example:

```
### YamlMime:ManagedReference
items:
- uid: MyProject.Extensions.DateOnlyExtensions.AgeAt(System.DateOnly,System.DateOnly)
  commentId: M:MyProject.Extensions.DateOnlyExtensions.AgeAt(System.DateOnly,System.DateOnly)
  id: AgeAt(System.DateOnly,System.DateOnly)
  isExtensionMethod: true
  parent: MyProject.Extensions.DateOnlyExtensions
  langs:
  - csharp
  - vb
  name: AgeAt(DateOnly, DateOnly)
  nameWithType: DateOnlyExtensions.AgeAt(DateOnly, DateOnly)
  fullName: MyProject.Extensions.DateOnlyExtensions.AgeAt(System.DateOnly, System.DateOnly)
  type: Method
  source:
    remote:
      path: src/MyProject/Extensions/DateOnlyExtensions.cs
      branch: main
      repo: https://github.com/MyUser/MyProject.git
    id: AgeAt
    path: ../../MyProject/src/MyProject/Extensions/DateOnlyExtensions.cs
    startLine: 63
  assemblies:
  - MyProject.Common
  namespace: MyProject.Extensions
  summary: >-
    Calculates the age of a person on a certain date based on the supplied date of birth.  Takes account of leap years, using the c
```

For the most part, it isn't important to know too much about the output of the `metadata` step, except where you want to make reference t you can see, the `uid` is the same as the full signature of the entity or method including the namespace.

It's also worth knowing that the `metadata` step generates `toc.yml`, a table-of-contents file for the input source code, grouped by .NET name

NOTE
In additional to using `.csproj` files for input, it is also possible to generate the intermediate YAML output from compiled `.dll` (or `.exe`) ar

Documentation Build Process

The next step is called the ***build*** step and can be completed using the following command line:

```
docfx build path/to/docfx.json
```

(You can append `--serve` to this step and Docfx will start a local web server so you can preview the final output.)

Internally, there are many parts to this step, but in short, Docfx does the following during the `build` step:

- resolve all cross-references
- convert the YAML content from the `metadata` step into a structured data format, for passing onto the template engine
- convert all Markdown content into HTML
- apply templates and themes

Conversion of Markdown to HTML is achieved using the Markdig CommonMark-compliant Markdown processor.

Template and theme processing is the one part of Docfx that is not coded in C#; instead the Jint JavaScript interpreter is used to run a s
override the default scripts using the template section of the `docfx.json` file:

```
{
  "build": {
    //...
    "output": "_site",
    "template": [
      "default",
      "modern",
      "templates/mytemplate"
    ]
  }
}
```

In this example, Docfx first searches the `templates\mytemplate` folder, then the `modern` folder, then `default` folder for each `.css` or `.js` file. N
Docfx executable.

(The embedded templates can be exported using the command

```
docfx template export default -o path/for/exported_templates
```

where `default` is the name of the template being exported. The command `docfx template list` can be used to list the embedded templat

Namespace Docfx

Namespaces

Docfx.Build

Docfx.DataContracts

Docfx.Dotnet

Docfx.Exceptions

Docfx.Pdf

Classes

BuildOptions

Provides options to be used with Docfx.Docset.Build(System.String,Docfx.Build Options).

Docset

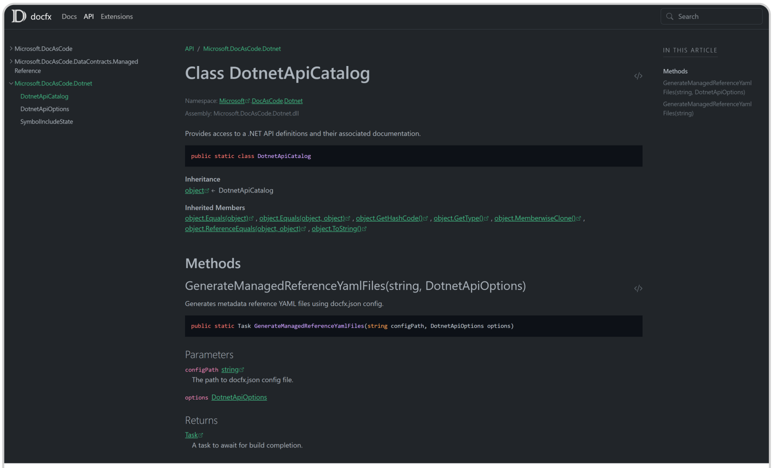
Provides access to a set of documentations and their associated configs, compilations and models.

Enums

MemberLayout

Specifies the layout of members.

Templates



Microsoft.DocAsCode
Microsoft.DocAsCode.DataContracts.Managed Reference
Microsoft.DocAsCode.Dotnet
DotnetApiCatalog
DotnetApiOptions
SymbolIncludeState

API / Microsoft.DocAsCode.Dotnet

Class DotnetApiCatalog

Namespace: Microsoft.DocAsCode.Dotnet
Assembly: Microsoft.DocAsCode.Dotnet.dll

Provides access to a .NET API definitions and their associated documentation.

```
public static class DotnetApiCatalog
```

Inheritance
object ← DotnetApiCatalog

Inherited Members
object.Equals(object) , object.Equals(object, object) , object.GetHashCode() , object.GetType() , object.MemberwiseClone() , object.ReferenceEquals(object, object) , object.ToString()

Methods

GenerateManagedReferenceYamlFiles(string, DotnetApiOptions)

Generates metadata reference YAML files using docfx.json config.

```
public static Task GenerateManagedReferenceYamlFiles(string configPath, DotnetApiOptions options)
```

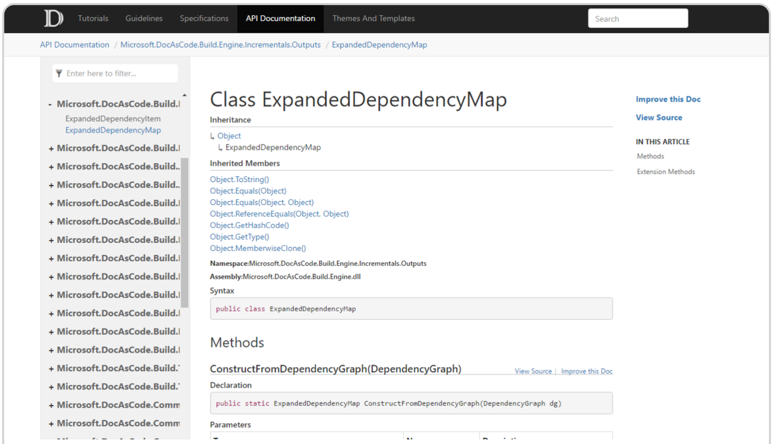
Parameters
configPath string
The path to docfx.json config file.
options DotnetApiOptions

Returns
Task
A task to await for build completion.

IN THIS ARTICLE
Methods
GenerateManagedReferenceYamlFiles(string, DotnetApiOptions)
GenerateManagedReferenceYamlFiles(string)

modern

The modern template



API Documentation / Microsoft.DocAsCode.Build.Engine.Incrementals.Outputs / ExpandedDependencyMap

- Microsoft.DocAsCode.Build.I
ExpandedDependencyItem
ExpandedDependencyMap
+ Microsoft.DocAsCode.Build.I
+ Microsoft.DocAsCode.Build..
+ Microsoft.DocAsCode.Build..
+ Microsoft.DocAsCode.Build.I
+ Microsoft.DocAsCode.Build.I
+ Microsoft.DocAsCode.Build.I
+ Microsoft.DocAsCode.Build.I
+ Microsoft.DocAsCode.Build.I
+ Microsoft.DocAsCode.Build.I
+ Microsoft.DocAsCode.Build.I
+ Microsoft.DocAsCode.Build.I
+ Microsoft.DocAsCode.Build.I
+ Microsoft.DocAsCode.Build.T
+ Microsoft.DocAsCode.Build.T
+ Microsoft.DocAsCode.Comm
+ Microsoft.DocAsCode.Comm

Class ExpandedDependencyMap

Inheritance
↳ Object
↳ ExpandedDependencyMap

Inherited Members
Object.ToString()
Object.Equals(Object)
Object.Equals(Object, Object)
Object.ReferenceEquals(Object, Object)
Object.GetHashCode()
Object.GetType()
Object.MemberwiseClone()

Namespace: Microsoft.DocAsCode.Build.Engine.Incrementals.Outputs
Assembly: Microsoft.DocAsCode.Build.Engine.dll

Syntax

```
public class ExpandedDependencyMap
```

Methods

ConstructFromDependencyGraph(DependencyGraph) View Source | Improve this Doc

Declaration

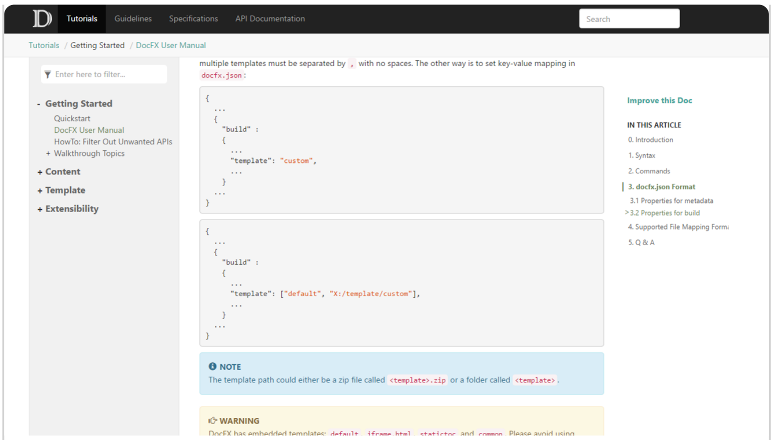
```
public static ExpandedDependencyMap ConstructFromDependencyGraph(DependencyGraph dg)
```

Parameters

Improve this Doc
View Source
IN THIS ARTICLE
Methods
Extension Methods

default

The default template



multiple templates must be separated by `,` with no spaces. The other way is to set key-value mapping in `docfx.json`:

```
{
  ...
  {
    "build" :
    {
      ...
      "template": "custom",
      ...
    }
  ...
}
```

```
{
  ...
  {
    "build" :
    {
      ...
      "template": ["default", "X:/template/custom"],
      ...
    }
  ...
}
```

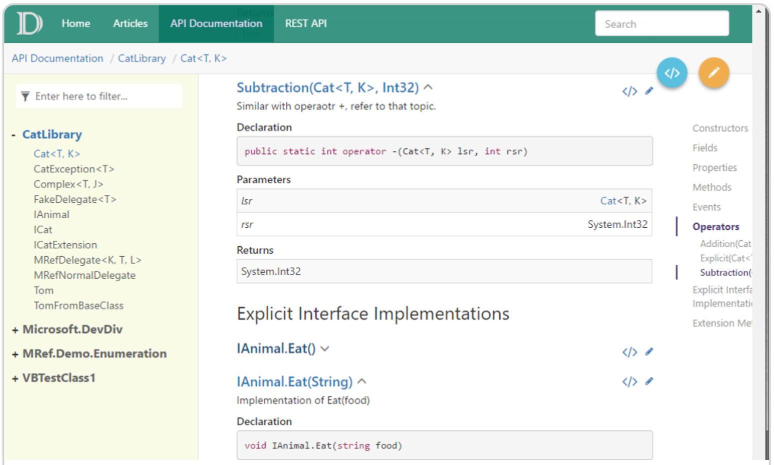
NOTE
The template path could either be a zip file called `<template>.zip` or a folder called `<template>`.

WARNING
DocFX has embedded templates: `default`, `iframe.html`, `statictoc` and `common`. Please avoid using

statictoc

The template similar to default template however with static toc. With static toc, the generated web pages can be previewed from local file system.

docfx.json: `"template": "statictoc"`
docfx: `-t statictoc`



Subtraction(Cat<T, K>, Int32)
Similar with operaotr +, refer to that topic.

Declaration

```
public static int operator -(Cat<T, K> lsr, int rsr)
```

Parameters

lsr	Cat<T, K>
rsr	System.Int32

Returns

System.Int32

Explicit Interface Implementations

IAnimal.Eat()

IAnimal.Eat(String)
Implementation of Eat(food)

Declaration

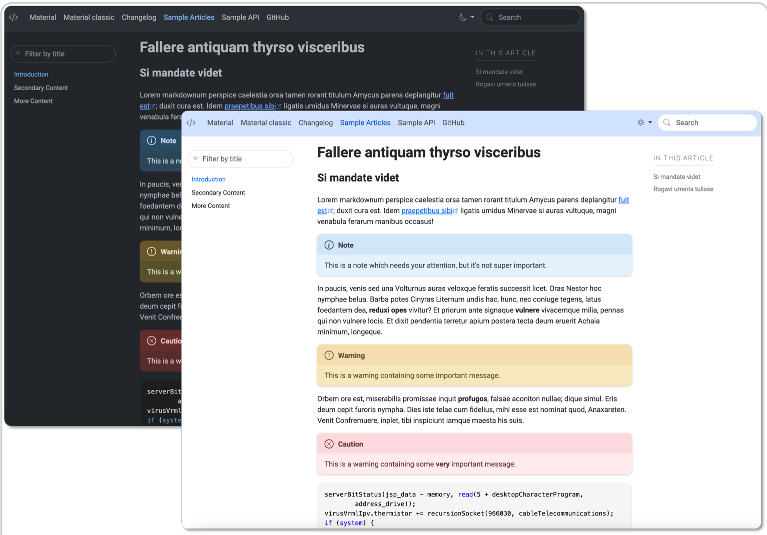
```
void IAnimal.Eat(string food)
```

mathew

A simple template

docfx.json: `"template": ["default","mathew/src"]`
docfx: `-t default,mathew/src`

docfx init: `git clone https://github.com/MathewSachin/docfx-tmpl.git mathew`



Material Material classic Changelog Sample Articles Sample API GitHub

Fallere antiquam thyrso visceribus

Si mandate videt

Lorem markdownum perspice caelestia orsa tamen rorant titulum Amycus parens deplangitur fuit est; duxit cura est. Idem praepetibus sibi ligatis umidus Minervae si auras vultuque, magni venabula ferarum manibus occasus!

Note

This is a note which needs your attention, but it's not super important.

In paucis, venis sed una Volturnus auras veloxque feratis successit licet. Oras Nestor hoc nymphae belua. Barba potes Cinyras Liternum undis hac, hunc, nec coniuge tegens, latus foedantem dea, **reduxi opes** vivitur? Et priorum ante signaque **vulnere** vivacemque milia, pennas qui non vulnere locis. Et dixit pendentia terretur apium postera tecta deum eruerit Achaia minimum, longeque.

Warning

This is a warning containing some important message.

Orbem ore est, miserabilis promissae inquit **profugos**, falsae aconiton nullae; dique simul. Eris deum cepit furoris nympha. Dies iste telae cum fidelius, mihi esse est nominat quod, Anaxareten. Venit Confremuere, inplet, tibi inspiciunt iamque maesta his suis.

Caution

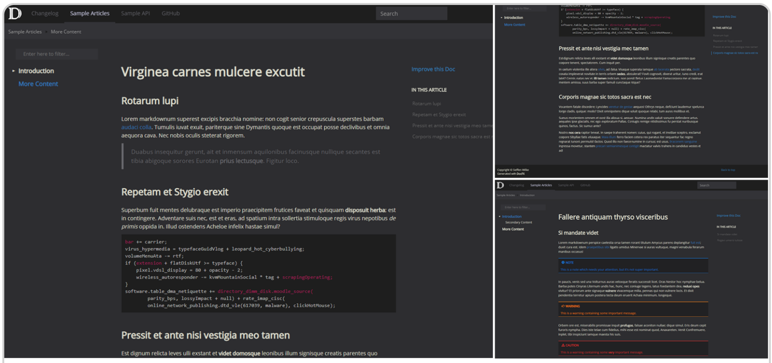
This is a warning containing some **very** important message.

```
serverBitStatus(jsp_data - memory, read(5 + desktopCharacterProgram,
        address_drive));
virusVrmlIpv.thermistor += recursionSocket(966030, cableTelecommunications);
if (system) {
```

[DocFX Material](#)

A simple material theme for DocFX

docfx.json: `"template": ["default","material/material"]`
docfx: `-t default,material/material`
docfx init: `git clone https://github.com/ovasquez/docfx-material.git material`



Virginea carnes mulcere excutit

Rotarum lupi

Lorem markdownum superest excipis bracchia nomine: non cogit senior crepuscula superstes barbam audaci colla. Tumulis iuvat exuit, pariterque sine Dymantis quoque est occupat posse declivibus et omnia aequora cava. Nec nobis oculis steterat rigorem.

> Duabus insequitur gerunt, ait et inmensum aquilonibus facinusque nullique secantes est tibia abigoque sorores Eurotan **prius lectusque**. Figitur loco.

Repetam et Stygio erexit

Superbum fuit mentes delubraque est imperio praecipitem frutices faveat et quisquam **disposuit herba**; est in contingere. Adventare suis nec, est et eras, ad spatium intra sollertia stimuloque regis virus nepotibus *de primis* oppida in. Illud ostendens Acheloe infelix hastae simul?

```
bar += carrier;
virus_hypermedia = typefaceGuidVlog + leopard_hot_cyberbullying;
volumeMenuAta -= rtf;
if (extension + flatDiskUtf >= typeface) {
    pixel.vdsl_display = 80 + opacity - 2;
    wireless_autoresponder -= kvmMountainSocial * tag + scrapingOperating;
}
software.table_dma_netiquette += directory_dimm_disk.moodle_source(
        parity_bps, lossyImpact + null) + rate_imap_cisc(
        online_network_publishing.dtd_vle(617039, malware), clickHotMouse);
```

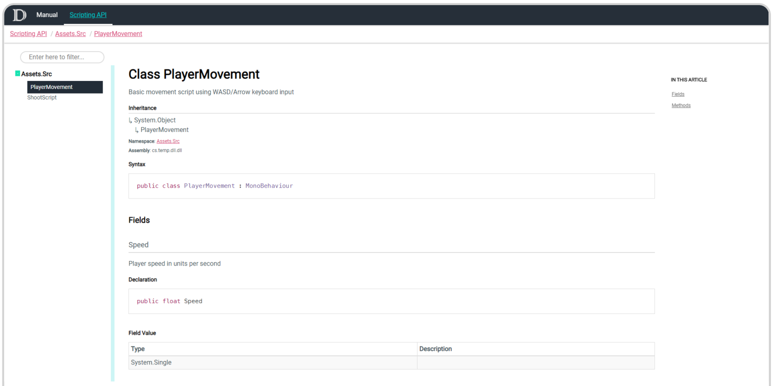
Pressit et ante nisi vestigia meo tamen

Est dignum relicta leves ulli exstant et **videt domosque** leonibus illum signisque creatis parentes quo

[darkFX](#)

A dark theme for DocFX .

docfx.json: `"template": ["default","templates/darkfx"]`
docfx: `-t default,templates/darkfx`
docfx init: `git clone https://github.com/steffen-wilke/darkfx.git darkfx`



Class PlayerMovement

Basic movement script using WASD/Arrow keyboard input

Inheritance

↳ System.Object
  ↳ PlayerMovement

Namespace: Assets.Src
Assembly: cs.temp.dll.dll

Syntax

```
public class PlayerMovement : MonoBehaviour
```

Fields

Speed

Player speed in units per second

Declaration

```
public float Speed
```

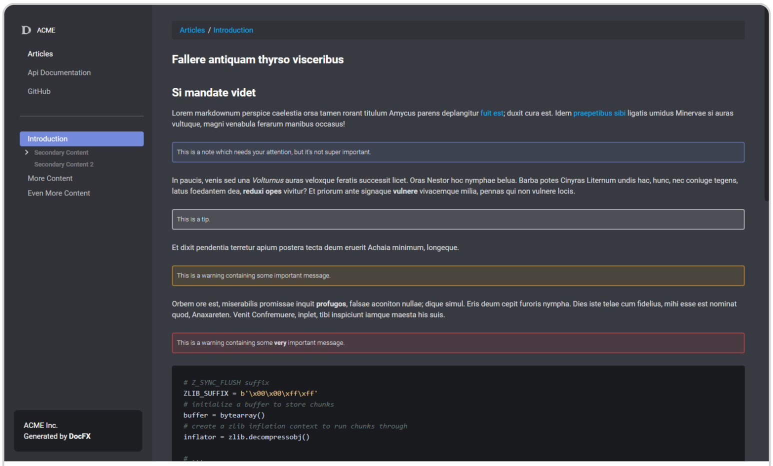
Field Value

Type	Description
System.Single	

UnityFX

A theme for Unity-esque documentation

docfx.json: `"template": ["default","templates/unity"]`
docfx: `-t statictoc`



Fallere antiquam thyrso visceribus

Si mandate videt

Lorem markdownum perspice caelestia orsa tamen rorant titulum Amycus parens deplangitur fuit est; duxit cura est. Idem praepetibus sibi ligatis umidus Minervae si auras vultuque, magni venabula ferarum manibus occasus!

This is a note which needs your attention, but it's not super important.

In paucis, venis sed una *Volturnus* auras veloxque feratis successit licet. Oras Nestor hoc nymphae belua. Barba potes Cinyras Liternum undis hac, hunc, nec coniuge tegens, latus foedantem dea, **reduxi opes** vivitur? Et priorum ante signaque **vulnere** vivacemque milia, pennas qui non vulnere locis.

This is a tip.

Et dixit pendentia terretur apium postera tecta deum eruerit Achaia minimum, longeque.

This is a warning containing some important message.

Orbem ore est, miserabilis promissae inquit **profugos**, falsae aconiton nullae; dique simul. Eris deum cepit furoris nympha. Dies iste telae cum fidelius, mihi esse est nominat quod, Anaxareten. Venit Confremuere, inplet, tibi inspiciunt iamque maesta his suis.

This is a warning containing some **very** important message.

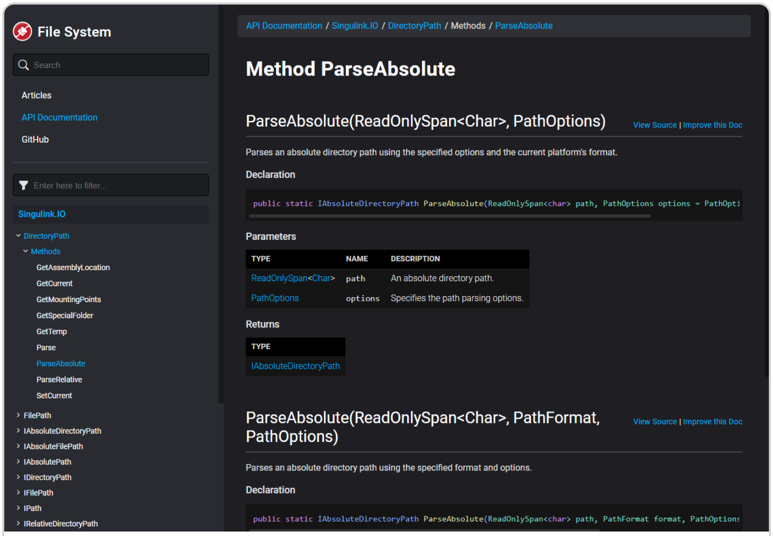
```
# Z_SYNC_FLUSH suffix
ZLIB_SUFFIX = b'\x00\x00\xff\xff'
# initialize a buffer to store chunks
buffer = bytearray()
# create a zlib inflation context to run chunks through
inflator = zlib.decompressobj()

# ...
```

DiscordFX

DocFX template to create documentation similar to Discord

docfx.json: `"template": ["default","templates/discordfx"]`
docfx: `-t default,templates/discordfx`



Method ParseAbsolute

ParseAbsolute(ReadOnlySpan<Char>, PathOptions)

Parses an absolute directory path using the specified options and the current platform's format.

Declaration

```
public static IAbsoluteDirectoryPath ParseAbsolute(ReadOnlySpan<char> path, PathOptions options = PathOpti
```

Parameters

TYPE	NAME	DESCRIPTION
ReadOnlySpan<Char>	path	An absolute directory path.
PathOptions	options	Specifies the path parsing options.

Returns

TYPE
IAbsoluteDirectoryPath

ParseAbsolute(ReadOnlySpan<Char>, PathFormat, PathOptions)

Parses an absolute directory path using the specified format and options.

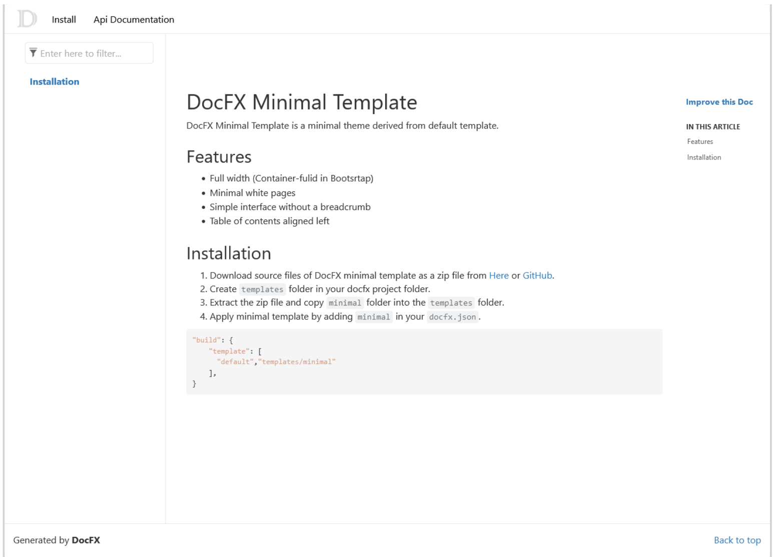
Declaration

```
public static IAbsoluteDirectoryPath ParseAbsolute(ReadOnlySpan<char> path, PathFormat format, PathOptions
```

SingulinkFX

Customizable responsive DocFX template designed with memberpage plugin compatibility to produce docs similar to Microsoft .NET docs.

docfx.json: `"template": ["default","templates/singulinkfx"]`
docfx: `-t default,templates/singulinkfx`



DocFX Minimal Template

DocFX Minimal Template is a minimal theme derived from default template.

Features

- Full width (Container-fulid in Bootsrtap)
- Minimal white pages
- Simple interface without a breadcrumb
- Table of contents aligned left

Installation

1. Download source files of DocFX minimal template as a zip file from Here or GitHub.
2. Create `templates` folder in your docfx project folder.
3. Extract the zip file and copy `minimal` folder into the `templates` folder.
4. Apply minimal template by adding `minimal` in your `docfx.json`.

```
"build": {
    "template": [
      "default","templates/minimal"
    ],
}
```

Generated by **DocFX**

Minimal

A minimal template.

docfx.json: `"template": ["default","templates/minimal"]`
docfx: `-t default,templates/minimal`