

Table of Contents

Overview	2
DebugInfo API	
C.Debugging	6
AssetReferences	7
Config	12
DebugInfo	18
UpdateMode	28
C.Debugging.Formatting	29
Str	30

DebugInfo Overview

Setup and Configuration

After installation, no setup is required to start using `DebugInfo` - it will initialise itself on game start.

However various global defaults can be accessed via the static `DebugInfo.Config` property.

NOTE

It's important any changes to the config happen before using `DebugInfo` to ensure that the changes are correctly applied.

Logging

The main logging function is `DebugInfo.Log`.

There are two basic versions, [one](#) that logs a single text message, and [another](#) that logs a key, value pair in two columns.

```
DebugInfo.Log("A single line of text.");
DebugInfo.Log("Key", "Value");
```

A single line of text.	
Key	Value

Color

It's possible to specify the color of the background; key text, value text, or both.

```
DebugInfo.Log("RedKey", "RedValue", Color.lightCoral);
DebugInfo.Log("GreenKey", Color.lawnGreen, "BlueValue", Color.lightSkyBlue);
```

RedKey	RedValue
GreenKey	BlueValue
RedBackground	

Formatting

Formatting and coloring individual parts of text can be done with the static methods provided by the [Str](#) class.

There are some global options for changing the formatting behaviour, such as the floating point [precision](#) field.

```
DebugInfo.Log("Time", $"{Str.F(Time.fixedTime)} ({Str.Cyan(Str.F(Time.frameCount))})");  
DebugInfo.Log("Velocity", Str.F(sphereRigidbody.linearVelocity), Str.TransformRgb);
```

Time	2.340 (144)
Velocity	<23.400, -22.955, 0.000>

Headings

The [Header](#) and [Spacer](#) methods can be used to create headings and gaps between rows.

```
DebugInfo.Heading("Heading");  
DebugInfo.Log("Key1", "Value1");  
  
// It's possible to set a height per spacer. If omitted the default as set in the  
// configuration is used.  
DebugInfo.Spacer(8);  
  
// Changing the background and border color is also possible.  
DebugInfo.Heading("Heading2", color: Color.lightCoral,  
                 bgColor: new Color(0.31f, 0.11f, 0.15f, 0.5f), borderColor: Color.lightCoral);  
DebugInfo.Log("Key2", "Value2");
```

Heading	
Key1	Value1
Heading2	
Key2	Value2

Groups

Rows can be grouped together into collapsible sections using the [Group](#) method.

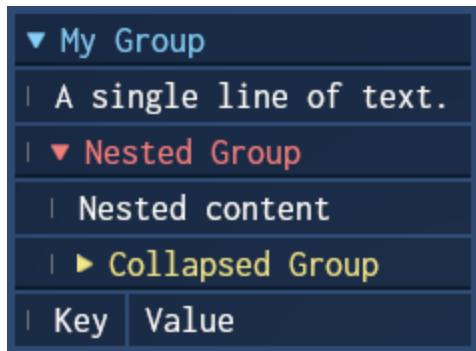
The [Group](#) method returns an [IDisposable](#) that must be used with a [using](#) block to correctly open and close the group.

Groups can be clicked to open and close them, and there are various options such as color, initial state, and a callback for when the group is opened or closed.

```

using (DebugInfo.Group("My Group", Color.lightSkyBlue)) {
    DebugInfo.Log("A single line of text.");
    using (DebugInfo.Group("Nested Group", Color.lightCoral)) {
        DebugInfo.Log("Nested content");
        using (DebugInfo.Group("Collapsed Group", Color.khaki, collapsed: true)) {
            DebugInfo.Log("Collapsed content");
        }
    }
    DebugInfo.Log("Key", "Value");
}

```



Notifications

DebugInfo also provides methods for showing once-off [notification messages](#).

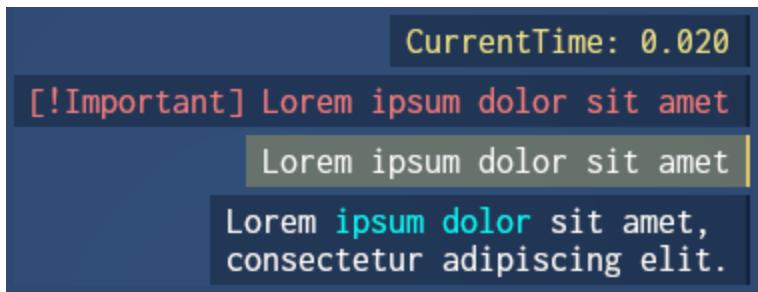
```

DebugInfo.Notify($"Lorem {Str.Cyan("ipsum dolor")} sit amet," +
    "\nconsectetur adipiscing elit.");
DebugInfo.Notify("Lorem ipsum dolor sit amet",
    bgColor: new Color(0.7f, 0.7f, 0.24f, 0.25f),
    borderColor: new Color(0.86f, 0.78f, 0.43f));

// Set the duration to 10 seconds before the notification fades out.
DebugInfo.Notify($"[!Important] Lorem ipsum dolor sit amet",
    color: Color.lightCoral, duration: 10);

// Giving the notification a unique id will cause subsequent calls to
// update the existing notification instead of creating new ones.
DebugInfo.Notify($"CurrentTime: {Str.F(Time.deltaTime)}", "UniqueId",
    color: Color.khaki);

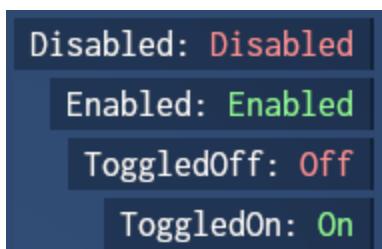
```



Toggle Messages

There are also convenience methods for creating on/off messages.

```
DebugInfo.NotifyOn("ToggledOn", true);
DebugInfo.NotifyOn("ToggledOff", false);
DebugInfo.NotifyEnabled("Enabled", true);
DebugInfo.NotifyEnabled("Disabled", false);
```



Namespace C.Debugging

Classes

[AssetReferences](#)

[Config](#)

Global configuration for DebugInfo.

Note that these settings should be changed before logging anything to properly take effect.

[DebugInfo](#)

The core DebugInfo class with various static methods for logging information.

Enums

[UpdateMode](#)

Class AssetReferences

```
[CreateAssetMenu(fileName = "AssetReferences", menuName = "DebugInfo.AssetReferences")]
public class AssetReferences : ScriptableObject
```

Inheritance

[object](#) ← [Object](#) ← [ScriptableObject](#) ← AssetReferences

Fields

cellPrefab

```
public Cell cellPrefab
```

Field Value

Cell

eventSystemPrefab

```
public EventSystem eventSystemPrefab
```

Field Value

EventSystem

groupHeadingPrefab

```
public GroupHeadingCell groupHeadingPrefab
```

Field Value

GroupHeadingCell

headingPrefab

```
public HeadingCell headingPrefab
```

Field Value

HeadingCell

indentMarginPrefab

```
public IndentMargin indentMarginPrefab
```

Field Value

IndentMargin

notificationPrefab

```
public Notification notificationPrefab
```

Field Value

Notification

rootPrefab

```
public DebugInfo rootPrefab
```

Field Value

[DebugInfo](#)

tablePrefab

```
public DebugInfoTable tablePrefab
```

Field Value

DebugInfoTable

Methods

Create<T>(GameObject, string, Transform)

```
public static T Create<T>(GameObject prefab, string name = null, Transform parent  
= null)
```

Parameters

prefab [GameObject](#)

name [string](#)

parent [Transform](#)

Returns

T

Type Parameters

T

Create<T>(GameObject, out T, string, Transform)

```
public static void Create<T>(GameObject prefab, out T component, string name = null,  
Transform parent = null)
```

Parameters

prefab [GameObject](#)

component T

name [string](#)

parent [Transform](#)

Type Parameters

T

Create<T>(MonoBehaviour, string, Transform)

```
public static T Create<T>(MonoBehaviour prefab, string name = null, Transform parent = null)
```

Parameters

prefab [MonoBehaviour](#)

name [string](#)

parent [Transform](#)

Returns

T

Type Parameters

T

Create<T>(MonoBehaviour, out T, string, Transform)

```
public static void Create<T>(MonoBehaviour prefab, out T component, string name = null, Transform parent = null)
```

Parameters

prefab [MonoBehaviour](#)

component T

name [string](#) ↗

parent [Transform](#) ↗

Type Parameters

T

Class Config

Global configuration for DebugInfo.

Note that these settings should be changed before logging anything to properly take effect.

```
public class Config
```

Inheritance

[object](#) ← Config

Fields

backgroundColor

The default cell background colour for all labels.

```
public Color backgroundColor
```

Field Value

[Color](#)

cellSpacing

The spacing between columns and rows.

```
public Vector2 cellSpacing
```

Field Value

[Vector2](#)

closeNotificationOnClick

If true, clicking on a notification will immediately close it.

```
public bool closeNotificationOnClick
```

Field Value

[bool ↗](#)

defaultNotificationBorderColor

If set shows a coloured border on the right hand side of all notifications.

```
public Color defaultNotificationBorderColor
```

Field Value

[Color ↗](#)

defaultNotificationColor

The default notification background colour.

```
public Color defaultNotificationColor
```

Field Value

[Color ↗](#)

defaultSpacerSize

The default height for spacers when a size isn't explicitly set.

```
public float defaultSpacerSize
```

Field Value

[float ↗](#)

groupIndent

The size of the indent inside of groups.

```
public float groupIndent
```

Field Value

[float](#)

headingBorderColor

The default border color for headings.

```
public Color headingBorderColor
```

Field Value

[Color](#)

headingTextPadding

The padding inside a heading cell around the text.

```
public Vector2 headingTextPadding
```

Field Value

[Vector2](#)

labelAlign

The default text alignment for all labels.

```
public TextAnchor labelAlign
```

Field Value

[TextAnchor](#)

margin

The spacing between the table and the screen.

```
public Vector2 margin
```

Field Value

[Vector2](#)

notificationFadeTime

How many seconds it takes a notification to fade in or out.

```
public float notificationFadeTime
```

Field Value

[float](#)

notificationSlideDistance

How far to the right notifications will slide when popping in and out.

```
public float notificationSlideDistance
```

Field Value

[float](#)

notificationTime

How many seconds a notification will stay on screen for.

```
public float notificationTime
```

Field Value

[float ↗](#)

showGroupIndentMargin

If true shows a border/margin on the left side of rows inside of groups.

```
public bool showGroupIndentMargin
```

Field Value

[bool ↗](#)

textColor

The default text colour for all labels.

```
public Color textColor
```

Field Value

[Color ↗](#)

textPadding

The padding inside a cell around the text.

```
public Vector2 textPadding
```

Field Value

updateMode

Controls when and how the debug info updates.

```
public UpdateMode updateMode
```

Field Value

[UpdateMode](#)

Class DebugInfo

The core DebugInfo class with various static methods for logging information.

```
public class DebugInfo : MonoBehaviour
```

Inheritance

[object](#) ← [Object](#) ← [Component](#) ← [Behaviour](#) ← [MonoBehaviour](#) ← DebugInfo

Properties

Config

Global configuration for DebugInfo. Changes to this should be done before using DebugInfo to ensure that the values are correctly applied to logs/notifications.

```
public static Config Config { get; }
```

Property Value

[Config](#)

Methods

Group(string, Color?, Color?, Color?, bool?,
Action<GroupHeadingRow, bool>)

Wraps subsequent logs in a collapsable group.

Returns an [IDisposable](#) that must be used with the `using` statement.

Usage:

```
using (DebugInfo.Group("My Group")) { /* ... */ }
```

```
public static GroupScope Group(string label, Color? color = null, Color? bgColor = null,  
Color? borderColor = null, bool? collapsed = null, Action<GroupHeadingRow, bool>  
onCollapsed = null)
```

Parameters

label [string](#)?

The group heading text.

color [Color](#)?

The text color.

bgColor [Color](#)?

The background color.

borderColor [Color](#)?

The bottom border color.

collapsed [bool](#)?

If non-null, sets the default state when the group is first created.

onCollapsed [Action](#)<GroupHeadingRow, [bool](#)>

A callback triggered when the group is toggled by the user.

Returns

GroupScope

Heading(string, Color?, Color?, Color?, TextAnchor?)

A header only has a single column, is slightly larger than normal rows, and has a subtle bottom border.

```
public static DebugInfoTable Heading(string label, Color? color = null, Color? bgColor = null, Color? borderColor = null, TextAnchor? alignment = null)
```

Parameters

label [string](#)?

The heading text.

color [Color](#)?

The text color.

bgColor [Color](#)?

The background color.

borderColor [Color](#)?

The bottom border color.

alignment [TextAnchor](#)?

How to align the text.

Returns

DebugInfoTable

Log(string, Color?, Color?)

Displays a row with a single column of text.

```
public static DebugInfoTable Log(string label, Color? color = null, Color? bgColor = null)
```

Parameters

label [string](#)

The text to display1111.

color [Color](#)?

The text color.

bgColor [Color](#)?

The background color.

Returns

Log(string, Color?, string, Color?, Color?)

Displays a row with a key and value column each with a different color.

```
public static DebugInfoTable Log(string label, Color? labelColor, string value, Color?  
valueColor, Color? bgColor = null)
```

Parameters

label [string](#)?

The key/label column text.

labelColor [Color](#)?

The key/label column text color.

value [string](#)?

The value column text.

valueColor [Color](#)?

The value column text color.

bgColor [Color](#)?

The background color.

Returns

DebugInfoTable

Log(string, string, Color?, Color?)

Displays a row with a key and value column.

```
public static DebugInfoTable Log(string label, string value, Color? color = null, Color?  
bgColor = null)
```

Parameters

label [string](#) ↗

The key/label column text.

value [string](#) ↗

The value column text.

color [Color](#) ↗?

The text color for the whole row.

bgColor [Color](#) ↗?

The background color.

Returns

[DebugInfoTable](#)

Notify(string, string, Color?, Color?, Color?, float)

Shows a temporary notification message.

```
public static void Notify(string message, string id = null, Color? borderColor = null,  
Color? bgColor = null, Color? color = null, float duration = 0)
```

Parameters

message [string](#) ↗

The text to display.

id [string](#) ↗

An optional unique identifier for this notification. If non-null subsequent calls with the same id will update the existing notification instead of creating new ones.

borderColor [Color](#) ↗?

The right-hand side border color. If null, the default color set in [Config](#) will be used. If the alpha is zero, no border will be shown.

bgColor [Color](#)?

The background color.

color [Color](#)?

The text color.

duration [float](#)

If > 0, sets how long the notification will remain visible for, otherwise uses the default set in [Config](#).

NotifyEnabled(string, bool, Color?, Color?, Color?, float)

Displays an enabled/disabled message in the format **LABEL: ENABLED/DISABLED** based on a condition. If the condition is true the **ON** text will be displayed, otherwise the **OFF** text is shown.

The enabled/disabled text can be globally controlled with the [enabledText](#) and [disabledText](#) fields. This notification is automatically made unique based on the **text** parameter.

```
public static void NotifyEnabled(string text, bool on, Color? borderColor = null, Color? bgColor = null, Color? color = null, float duration = 0)
```

Parameters

text [string](#)

The label text to display.

on [bool](#)

borderColor [Color](#)?

The right-hand side border color. If null, the default color set in [Config](#) will be used. If the alpha is zero, no border will be shown.

bgColor [Color](#)?

The background color.

color [Color](#)?

The text color.

duration [float](#)

If > 0 , sets how long the notification will remain visible for, otherwise uses the default set in [Config](#).

NotifyOn(string, bool, Color?, Color?, Color?, float)

Displays an on/off message in the format **LABEL: ON/OFF** based on a condition.

If the condition is true the **ON** text will be displayed, otherwise the **OFF** text is shown.

The on/off text can be globally controlled with the [onText](#) and [offText](#) fields.

This notification is automatically made unique based on the **text** parameter.

```
public static void NotifyOn(string text, bool on, Color? borderColor = null, Color?  
bgColor = null, Color? color = null, float duration = 0)
```

Parameters

text [string](#)

The label text to display.

on [bool](#)

Whether to show the on or off text.

borderColor [Color](#)?

The right-hand side border color. If null, the default color set in [Config](#) will be used. If the alpha is zero, no border will be shown.

bgColor [Color](#)?

The background color.

color [Color](#)?

The text color.

duration [float](#)

If > 0 , sets how long the notification will remain visible for, otherwise uses the default set in [Config](#)

NotifyToggle(string, bool, string, string, Color?, Color?, Color?, float)

Displays a toggle message in the format `LABEL: ON/OFF` based on a condition. If the condition is true the `ON` text will be displayed, otherwise the `OFF` text is shown.

This notification is automatically made unique based on the `text` parameter.

```
public static void NotifyToggle(string text, bool on, string onText, string offText,  
Color? borderColor = null, Color? bgColor = null, Color? color = null, float duration  
= 0)
```

Parameters

`text` [string](#) ↗

The label text to display.

`on` [bool](#) ↗

Whether to show the on or off text.

`onText` [string](#) ↗

The value text to show the condition is true.

`offText` [string](#) ↗

The value text to show the condition is false.

`borderColor` [Color](#) ↗?

The right-hand side border color. If null, the default color set in [Config](#) will be used. If the alpha is zero, no border will be shown.

`bgColor` [Color](#) ↗?

The background color.

`color` [Color](#) ↗?

The text color.

duration [float](#)

If > 0, sets how long the notification will remain visible for, otherwise uses the default set in [Config](#).

Spacer(float?)

Adds empty space between the previous and next row.

```
public static DebugInfoTable Spacer(float? space = null)
```

Parameters

space [float](#)?

If not provided the default spacing set in [Config](#) will be used.

Returns

```
DebugInfoTable
```

TryGroup(bool)

A convenience method for conditionally wrapping logs in a group.

Usage:

```
using (DebugInfo.TryGroup(condition) ?? DebugInfo.Group("My Group")) { /* ... */ }
```

```
public static IDisposable TryGroup(bool condition)
```

Parameters

condition [bool](#)

If true, the group will work as per normal. If false, no group will be created and subsequent logs will be output normally.

Returns

[IDisposable](#)

UpdateAll()

Make sure to call this if [updateMode](#) is set to [Manual](#).

If the update mode is not set to [Manual](#), this will do nothing and issue a warning.

Failing to call this when the update mode is manual will prevent previous frame logs from being reset, causing memory leaks and performance issues.

```
public static void UpdateAll()
```

Enum UpdateMode

```
public enum UpdateMode
```

Fields

`FixedUpdate = 1`

Updates automatically happen every frame during Unity's `FixedUpdate`.

`Manual = 2`

Updates do not happen automatically and [UpdateAll\(\)](#) must be called every frame.

It's important that [UpdateAll\(\)](#) is called every frame to reset the previous frame's data otherwise all log calls will continue to accumulate causing a memory leak and performance issues.

[UpdateAll\(\)](#) should be called as late as possible after all logging calls for the current frame for info to correctly update.

`Update = 0`

Updates automatically happen every frame during Unity's `Update`.

Namespace C.Debugging.Formatting

Classes

[Str](#)

Contains helper methods for consistently formatting debug text.

Class Str

Contains helper methods for consistently formatting debug text.

```
public static class Str
```

Inheritance

[object](#) ↗ ← Str

Fields

CollisionHex

```
public static readonly string CollisionHex
```

Field Value

[string](#) ↗

CollisionRgb

```
public static readonly Color CollisionRgb
```

Field Value

[Color](#) ↗

OffHex

```
public static readonly string OffHex
```

Field Value

[string](#) ↗

OffRgb

```
public static readonly Color OffRgb
```

Field Value

[Color ↗](#)

OnHex

```
public static readonly string OnHex
```

Field Value

[string ↗](#)

OnRgb

```
public static readonly Color OnRgb
```

Field Value

[Color ↗](#)

StateHex

```
public static readonly string StateHex
```

Field Value

[string ↗](#)

StateRgb

```
public static readonly Color StateRgb
```

Field Value

[Color ↗](#)

TransformHex

```
public static readonly string TransformHex
```

Field Value

[string ↗](#)

TransformRgb

```
public static readonly Color TransformRgb
```

Field Value

[Color ↗](#)

colorPrefixText

The prefix when formatting a Unity Color with [E\(Color\)](#).

```
public static string colorPrefixText
```

Field Value

[string ↗](#)

defaultPrecision

The default precision when formatting floats.

```
public static int defaultPrecision
```

Field Value

[int ↗](#)

disabledText

The "disabled" text for [OnMsg\(string, bool\)](#).

```
public static string disabledText
```

Field Value

[string ↗](#)

enabledText

The "enabled" text for [OnMsg\(string, bool\)](#).

```
public static string enabledText
```

Field Value

[string ↗](#)

offText

The "off" text for [OnMsg\(string, bool\)](#).

```
public static string offText
```

Field Value

[string ↗](#)

onText

The "on" text for [OnMsg\(string, bool\)](#).

```
public static string onText
```

Field Value

[string ↗](#)

Methods

AliceBlue(string)

```
public static string AliceBlue(string v)
```

Parameters

v [string ↗](#)

Returns

[string ↗](#)

AntiqueWhite(string)

```
public static string AntiqueWhite(string v)
```

Parameters

v [string ↗](#)

Returns

[string ↗](#)

Aquamarine(string)

```
public static string Aquamarine(string v)
```

Parameters

v [string ↗](#)

Returns

[string ↗](#)

Azure(string)

```
public static string Azure(string v)
```

Parameters

v [string ↗](#)

Returns

[string ↗](#)

B<T>(T)

```
public static string B<T>(T v)
```

Parameters

v T

Returns

[string](#)

Type Parameters

T

Beige(string)

```
public static string Beige(string v)
```

Parameters

v [string](#)

Returns

[string](#)

Bisque(string)

```
public static string Bisque(string v)
```

Parameters

v [string](#)

Returns

[string](#)

Black(string)

```
public static string Black(string v)
```

Parameters

v [string](#)

Returns

[string](#)

BlanchedAlmond(string)

```
public static string BlanchedAlmond(string v)
```

Parameters

v [string](#)

Returns

[string](#)

Blue(string)

```
public static string Blue(string v)
```

Parameters

v [string](#)

Returns

[string](#)

BlueViolet(string)

```
public static string BlueViolet(string v)
```

Parameters

v [string](#)

Returns

[string](#)

Brown(string)

```
public static string Brown(string v)
```

Parameters

v [string](#)

Returns

[string](#)

Burlywood(string)

```
public static string Burlywood(string v)
```

Parameters

v [string](#)

Returns

[string](#)

CadetBlue(string)

```
public static string CadetBlue(string v)
```

Parameters

v [string](#)

Returns

[string](#)

Chartreuse(string)

```
public static string Chartreuse(string v)
```

Parameters

v [string](#)

Returns

[string](#)

Chocolate(string)

```
public static string Chocolate(string v)
```

Parameters

v [string](#)

Returns

[string](#)

Clear(string)

```
public static string Clear(string v)
```

Parameters

v [string](#)

Returns

[string](#)

Clr(string, string)

```
public static string Clr(string v, string clr)
```

Parameters

v [string](#)

clr [string](#)

Returns

[string](#)

CollisionClr(string)

```
public static string CollisionClr(string v)
```

Parameters

v [string](#)

Returns

[string](#)

Coral(string)

```
public static string Coral(string v)
```

Parameters

v [string ↗](#)

Returns

[string ↗](#)

CornflowerBlue(string)

```
public static string CornflowerBlue(string v)
```

Parameters

v [string ↗](#)

Returns

[string ↗](#)

Cornsilk(string)

```
public static string Cornsilk(string v)
```

Parameters

v [string ↗](#)

Returns

[string ↗](#)

Crimson(string)

```
public static string Crimson(string v)
```

Parameters

v [string ↗](#)

Returns

[string ↗](#)

Cyan(string)

```
public static string Cyan(string v)
```

Parameters

v [string ↗](#)

Returns

[string ↗](#)

DeepPink(string)

```
public static string DeepPink(string v)
```

Parameters

v [string ↗](#)

Returns

[string ↗](#)

DeepSkyBlue(string)

```
public static string DeepSkyBlue(string v)
```

Parameters

v [string ↗](#)

Returns

[string ↗](#)

DimGray(string)

```
public static string DimGray(string v)
```

Parameters

v [string ↗](#)

Returns

[string ↗](#)

DodgerBlue(string)

```
public static string DodgerBlue(string v)
```

Parameters

v [string ↗](#)

Returns

[string ↗](#)

DrkBlue(string)

```
public static string DrkBlue(string v)
```

Parameters

v [string ↗](#)

Returns

[string ↗](#)

DrkCyan(string)

```
public static string DrkCyan(string v)
```

Parameters

v [string ↗](#)

Returns

[string ↗](#)

DrkGoldenRod(string)

```
public static string DrkGoldenRod(string v)
```

Parameters

v [string ↗](#)

Returns

[string ↗](#)

DrkGray(string)

```
public static string DrkGray(string v)
```

Parameters

v [string ↗](#)

Returns

[string ↗](#)

DrkGreen(string)

```
public static string DrkGreen(string v)
```

Parameters

v [string ↗](#)

Returns

[string ↗](#)

DrkKhaki(string)

```
public static string DrkKhaki(string v)
```

Parameters

v [string ↗](#)

Returns

[string ↗](#)

DrkMagenta(string)

```
public static string DrkMagenta(string v)
```

Parameters

v [string ↗](#)

Returns

[string ↗](#)

DrkOliveGreen(string)

```
public static string DrkOliveGreen(string v)
```

Parameters

v [string ↗](#)

Returns

[string ↗](#)

DrkOrange(string)

```
public static string DrkOrange(string v)
```

Parameters

v [string ↗](#)

Returns

[string ↗](#)

DrkOrchid(string)

```
public static string DrkOrchid(string v)
```

Parameters

v [string ↗](#)

Returns

[string ↗](#)

DrkRed(string)

```
public static string DrkRed(string v)
```

Parameters

v [string ↗](#)

Returns

[string ↗](#)

DrkSalmon(string)

```
public static string DrkSalmon(string v)
```

Parameters

v [string ↗](#)

Returns

[string ↗](#)

DrkSeaGreen(string)

```
public static string DrkSeaGreen(string v)
```

Parameters

v [string ↗](#)

Returns

[string ↗](#)

DrkSlateBlue(string)

```
public static string DrkSlateBlue(string v)
```

Parameters

v [string ↗](#)

Returns

[string ↗](#)

DrkSlateGray(string)

```
public static string DrkSlateGray(string v)
```

Parameters

v [string ↗](#)

Returns

[string ↗](#)

DrkTurquoise(string)

```
public static string DrkTurquoise(string v)
```

Parameters

v [string ↗](#)

Returns

[string ↗](#)

DrkViolet(string)

```
public static string DrkViolet(string v)
```

Parameters

v [string ↗](#)

Returns

[string ↗](#)

EnabledMsg(string, bool)

```
public static string EnabledMsg(string text, bool on)
```

Parameters

text [string ↗](#)

on [bool ↗](#)

Returns

[string ↗](#)

F(int)

```
public static string F(int v)
```

Parameters

v [int](#)

Returns

[string](#)

F(float)

```
public static string F(float v)
```

Parameters

v [float](#)

Returns

[string](#)

F(float, int)

```
public static string F(float v, int precision)
```

Parameters

v [float](#)

precision [int](#)

Returns

[string](#)

F(Color)

```
public static string F(Color v)
```

Parameters

v [Color](#)

Returns

[string](#)

F(Vector2)

```
public static string F(Vector2 v)
```

Parameters

v [Vector2](#)

Returns

[string](#)

F(Vector3)

```
public static string F(Vector3 v)
```

Parameters

v [Vector3](#)

Returns

[string](#)

F<T>(T)

```
public static string F<T>(T v)
```

Parameters

v T

Returns

[string ↗](#)

Type Parameters

T

Firebrick(string)

```
public static string Firebrick(string v)
```

Parameters

v [string ↗](#)

Returns

[string ↗](#)

FloralWhite(string)

```
public static string FloralWhite(string v)
```

Parameters

v [string ↗](#)

Returns

[string](#)

ForestGreen(string)

```
public static string ForestGreen(string v)
```

Parameters

v [string](#)

Returns

[string](#)

Gainsboro(string)

```
public static string Gainsboro(string v)
```

Parameters

v [string](#)

Returns

[string](#)

GhostWhite(string)

```
public static string GhostWhite(string v)
```

Parameters

v [string](#)

Returns

[string ↗](#)

Gold(string)

```
public static string Gold(string v)
```

Parameters

v [string ↗](#)

Returns

[string ↗](#)

GoldenRod(string)

```
public static string GoldenRod(string v)
```

Parameters

v [string ↗](#)

Returns

[string ↗](#)

Gray(string)

```
public static string Gray(string v)
```

Parameters

v [string ↗](#)

Returns

[string ↗](#)

Gray1(string)

```
public static string Gray1(string v)
```

Parameters

v [string ↗](#)

Returns

[string ↗](#)

Gray2(string)

```
public static string Gray2(string v)
```

Parameters

v [string ↗](#)

Returns

[string ↗](#)

Gray3(string)

```
public static string Gray3(string v)
```

Parameters

v [string ↗](#)

Returns

[string ↗](#)

Gray4(string)

```
public static string Gray4(string v)
```

Parameters

v [string ↗](#)

Returns

[string ↗](#)

Gray5(string)

```
public static string Gray5(string v)
```

Parameters

v [string ↗](#)

Returns

[string ↗](#)

Gray6(string)

```
public static string Gray6(string v)
```

Parameters

v [string ↗](#)

Returns

[string](#)

Gray7(string)

```
public static string Gray7(string v)
```

Parameters

v [string](#)

Returns

[string](#)

Gray8(string)

```
public static string Gray8(string v)
```

Parameters

v [string](#)

Returns

[string](#)

Gray9(string)

```
public static string Gray9(string v)
```

Parameters

v [string](#)

Returns

[string ↗](#)

Green(string)

```
public static string Green(string v)
```

Parameters

v [string ↗](#)

Returns

[string ↗](#)

GreenYellow(string)

```
public static string GreenYellow(string v)
```

Parameters

v [string ↗](#)

Returns

[string ↗](#)

Grey(string)

```
public static string Grey(string v)
```

Parameters

v [string ↗](#)

Returns

[string](#)

Honeydew(string)

```
public static string Honeydew(string v)
```

Parameters

v [string](#)

Returns

[string](#)

HotPink(string)

```
public static string HotPink(string v)
```

Parameters

v [string](#)

Returns

[string](#)

I<T>(T)

```
public static string I<T>(T v)
```

Parameters

v T

Returns

[string ↗](#)

Type Parameters

T

IndianRed(string)

```
public static string IndianRed(string v)
```

Parameters

v [string ↗](#)

Returns

[string ↗](#)

Indigo(string)

```
public static string Indigo(string v)
```

Parameters

v [string ↗](#)

Returns

[string ↗](#)

Ivory(string)

```
public static string Ivory(string v)
```

Parameters

v [string ↗](#)

Returns

[string ↗](#)

Khaki(string)

```
public static string Khaki(string v)
```

Parameters

v [string ↗](#)

Returns

[string ↗](#)

Lavender(string)

```
public static string Lavender(string v)
```

Parameters

v [string ↗](#)

Returns

[string ↗](#)

LavenderBlush(string)

```
public static string LavenderBlush(string v)
```

Parameters

v [string ↗](#)

Returns

[string ↗](#)

LawnGreen(string)

```
public static string LawnGreen(string v)
```

Parameters

v [string ↗](#)

Returns

[string ↗](#)

LemonChiffon(string)

```
public static string LemonChiffon(string v)
```

Parameters

v [string ↗](#)

Returns

[string ↗](#)

LimeGreen(string)

```
public static string LimeGreen(string v)
```

Parameters

v [string ↗](#)

Returns

[string ↗](#)

Linen(string)

```
public static string Linen(string v)
```

Parameters

v [string ↗](#)

Returns

[string ↗](#)

LiteBlue(string)

```
public static string LiteBlue(string v)
```

Parameters

v [string ↗](#)

Returns

[string ↗](#)

LiteCoral(string)

```
public static string LiteCoral(string v)
```

Parameters

v [string ↗](#)

Returns

[string ↗](#)

LiteCyan(string)

```
public static string LiteCyan(string v)
```

Parameters

v [string ↗](#)

Returns

[string ↗](#)

LiteGoldenRod(string)

```
public static string LiteGoldenRod(string v)
```

Parameters

v [string ↗](#)

Returns

[string ↗](#)

LiteGoldenRodYellow(string)

```
public static string LiteGoldenRodYellow(string v)
```

Parameters

v [string ↗](#)

Returns

[string ↗](#)

LiteGray(string)

```
public static string LiteGray(string v)
```

Parameters

v [string ↗](#)

Returns

[string ↗](#)

LiteGreen(string)

```
public static string LiteGreen(string v)
```

Parameters

v [string ↗](#)

Returns

[string ↗](#)

LitePink(string)

```
public static string LitePink(string v)
```

Parameters

v [string ↗](#)

Returns

[string ↗](#)

LiteSalmon(string)

```
public static string LiteSalmon(string v)
```

Parameters

v [string ↗](#)

Returns

[string ↗](#)

LiteSeaGreen(string)

```
public static string LiteSeaGreen(string v)
```

Parameters

v [string ↗](#)

Returns

[string ↗](#)

LiteSkyBlue(string)

```
public static string LiteSkyBlue(string v)
```

Parameters

v [string ↗](#)

Returns

[string ↗](#)

LiteSlateBlue(string)

```
public static string LiteSlateBlue(string v)
```

Parameters

v [string ↗](#)

Returns

[string ↗](#)

LiteSlateGray(string)

```
public static string LiteSlateGray(string v)
```

Parameters

v [string ↗](#)

Returns

[string ↗](#)

LiteSteelBlue(string)

```
public static string LiteSteelBlue(string v)
```

Parameters

v [string ↗](#)

Returns

[string ↗](#)

LiteYellow(string)

```
public static string LiteYellow(string v)
```

Parameters

v [string ↗](#)

Returns

[string ↗](#)

Magenta(string)

```
public static string Magenta(string v)
```

Parameters

v [string ↗](#)

Returns

[string ↗](#)

Maroon(string)

```
public static string Maroon(string v)
```

Parameters

v [string ↗](#)

Returns

[string ↗](#)

MediumAquamarine(string)

```
public static string MediumAquamarine(string v)
```

Parameters

v [string ↗](#)

Returns

[string ↗](#)

MediumBlue(string)

```
public static string MediumBlue(string v)
```

Parameters

v [string ↗](#)

Returns

[string ↗](#)

MediumOrchid(string)

```
public static string MediumOrchid(string v)
```

Parameters

v [string ↗](#)

Returns

[string ↗](#)

MediumPurple(string)

```
public static string MediumPurple(string v)
```

Parameters

v [string ↗](#)

Returns

[string ↗](#)

MediumSeaGreen(string)

```
public static string MediumSeaGreen(string v)
```

Parameters

v [string ↗](#)

Returns

[string ↗](#)

MediumSlateBlue(string)

```
public static string MediumSlateBlue(string v)
```

Parameters

v [string ↗](#)

Returns

[string ↗](#)

MediumSpringGreen(string)

```
public static string MediumSpringGreen(string v)
```

Parameters

v [string ↗](#)

Returns

[string ↗](#)

MediumTurquoise(string)

```
public static string MediumTurquoise(string v)
```

Parameters

v [string ↗](#)

Returns

[string ↗](#)

MediumVioletRed(string)

```
public static string MediumVioletRed(string v)
```

Parameters

v [string ↗](#)

Returns

[string ↗](#)

MidnightBlue(string)

```
public static string MidnightBlue(string v)
```

Parameters

v [string ↗](#)

Returns

[string ↗](#)

MintCream(string)

```
public static string MintCream(string v)
```

Parameters

v [string ↗](#)

Returns

[string ↗](#)

MistyRose(string)

```
public static string MistyRose(string v)
```

Parameters

v [string ↗](#)

Returns

[string ↗](#)

Moccasin(string)

```
public static string Moccasin(string v)
```

Parameters

v [string ↗](#)

Returns

[string ↗](#)

NavajoWhite(string)

```
public static string NavajoWhite(string v)
```

Parameters

v [string ↗](#)

Returns

[string ↗](#)

NavyBlue(string)

```
public static string NavyBlue(string v)
```

Parameters

v [string ↗](#)

Returns

[string ↗](#)

OffClr(string)

```
public static string OffClr(string v)
```

Parameters

v [string ↗](#)

Returns

[string ↗](#)

OldLace(string)

```
public static string OldLace(string v)
```

Parameters

v [string ↗](#)

Returns

[string ↗](#)

Olive(string)

```
public static string Olive(string v)
```

Parameters

v [string ↗](#)

Returns

[string ↗](#)

OliveDrab(string)

```
public static string OliveDrab(string v)
```

Parameters

v [string ↗](#)

Returns

[string ↗](#)

OnClr(string)

```
public static string OnClr(string v)
```

Parameters

v [string ↗](#)

Returns

[string ↗](#)

OnMsg(string, bool)

```
public static string OnMsg(string text, bool on)
```

Parameters

text [string](#) ↗

on [bool](#) ↗

Returns

[string](#) ↗

Orange(string)

```
public static string Orange(string v)
```

Parameters

v [string](#) ↗

Returns

[string](#) ↗

OrangeRed(string)

```
public static string OrangeRed(string v)
```

Parameters

v [string](#) ↗

Returns

[string](#) ↗

Orchid(string)

```
public static string Orchid(string v)
```

Parameters

v [string ↗](#)

Returns

[string ↗](#)

PaleGoldenRod(string)

```
public static string PaleGoldenRod(string v)
```

Parameters

v [string ↗](#)

Returns

[string ↗](#)

PaleGreen(string)

```
public static string PaleGreen(string v)
```

Parameters

v [string ↗](#)

Returns

[string ↗](#)

PaleTurquoise(string)

```
public static string PaleTurquoise(string v)
```

Parameters

v [string ↗](#)

Returns

[string ↗](#)

PaleVioletRed(string)

```
public static string PaleVioletRed(string v)
```

Parameters

v [string ↗](#)

Returns

[string ↗](#)

PapayaWhip(string)

```
public static string PapayaWhip(string v)
```

Parameters

v [string ↗](#)

Returns

[string ↗](#)

PeachPuff(string)

```
public static string PeachPuff(string v)
```

Parameters

v [string ↗](#)

Returns

[string ↗](#)

Peru(string)

```
public static string Peru(string v)
```

Parameters

v [string ↗](#)

Returns

[string ↗](#)

Pink(string)

```
public static string Pink(string v)
```

Parameters

v [string ↗](#)

Returns

[string ↗](#)

Plum(string)

```
public static string Plum(string v)
```

Parameters

v [string](#)

Returns

[string](#)

PowderBlue(string)

```
public static string PowderBlue(string v)
```

Parameters

v [string](#)

Returns

[string](#)

Purple(string)

```
public static string Purple(string v)
```

Parameters

v [string](#)

Returns

[string](#)

RebeccaPurple(string)

```
public static string RebeccaPurple(string v)
```

Parameters

v [string](#)

Returns

[string](#)

Red(string)

```
public static string Red(string v)
```

Parameters

v [string](#)

Returns

[string](#)

RosyBrown(string)

```
public static string RosyBrown(string v)
```

Parameters

v [string](#)

Returns

[string](#)

RoyalBlue(string)

```
public static string RoyalBlue(string v)
```

Parameters

v [string ↗](#)

Returns

[string ↗](#)

SaddleBrown(string)

```
public static string SaddleBrown(string v)
```

Parameters

v [string ↗](#)

Returns

[string ↗](#)

Salmon(string)

```
public static string Salmon(string v)
```

Parameters

v [string ↗](#)

Returns

[string ↗](#)

SandyBrown(string)

```
public static string SandyBrown(string v)
```

Parameters

v [string ↗](#)

Returns

[string ↗](#)

SeaGreen(string)

```
public static string SeaGreen(string v)
```

Parameters

v [string ↗](#)

Returns

[string ↗](#)

Seashell(string)

```
public static string Seashell(string v)
```

Parameters

v [string ↗](#)

Returns

[string ↗](#)

Sienna(string)

```
public static string Sienna(string v)
```

Parameters

v [string](#)

Returns

[string](#)

Silver(string)

```
public static string Silver(string v)
```

Parameters

v [string](#)

Returns

[string](#)

Size<T>(T, int)

```
public static string Size<T>(T v, int size)
```

Parameters

v T

size [int](#)

Returns

[string](#)

Type Parameters

T

SkyBlue(string)

```
public static string SkyBlue(string v)
```

Parameters

v [string ↗](#)

Returns

[string ↗](#)

SlateBlue(string)

```
public static string SlateBlue(string v)
```

Parameters

v [string ↗](#)

Returns

[string ↗](#)

SlateGray(string)

```
public static string SlateGray(string v)
```

Parameters

v [string ↗](#)

Returns

[string ↗](#)

Snow(string)

```
public static string Snow(string v)
```

Parameters

v [string ↗](#)

Returns

[string ↗](#)

SoftBlue(string)

```
public static string SoftBlue(string v)
```

Parameters

v [string ↗](#)

Returns

[string ↗](#)

SoftGreen(string)

```
public static string SoftGreen(string v)
```

Parameters

v [string ↗](#)

Returns

[string](#)

SoftRed(string)

```
public static string SoftRed(string v)
```

Parameters

v [string](#)

Returns

[string](#)

SoftYellow(string)

```
public static string SoftYellow(string v)
```

Parameters

v [string](#)

Returns

[string](#)

SpringGreen(string)

```
public static string SpringGreen(string v)
```

Parameters

v [string](#)

Returns

[string](#) ↗

StateClr(string)

```
public static string StateClr(string v)
```

Parameters

v [string](#) ↗

Returns

[string](#) ↗

SteelBlue(string)

```
public static string SteelBlue(string v)
```

Parameters

v [string](#) ↗

Returns

[string](#) ↗

Tan(string)

```
public static string Tan(string v)
```

Parameters

v [string](#) ↗

Returns

[string](#)

Teal(string)

```
public static string Teal(string v)
```

Parameters

v [string](#)

Returns

[string](#)

Thistle(string)

```
public static string Thistle(string v)
```

Parameters

v [string](#)

Returns

[string](#)

ToHex(Color)

```
public static string ToHex(Color color)
```

Parameters

color [Color](#)

Returns

[string ↗](#)

ToggleMsg(string, bool, string, string)

```
public static string ToggleMsg(string text, bool on, string onText, string offText)
```

Parameters

text [string ↗](#)

on [bool ↗](#)

onText [string ↗](#)

offText [string ↗](#)

Returns

[string ↗](#)

Tomato(string)

```
public static string Tomato(string v)
```

Parameters

v [string ↗](#)

Returns

[string ↗](#)

TransformClr(string)

```
public static string TransformClr(string v)
```

Parameters

v [string ↗](#)

Returns

[string ↗](#)

Turquoise(string)

```
public static string Turquoise(string v)
```

Parameters

v [string ↗](#)

Returns

[string ↗](#)

Violet(string)

```
public static string Violet(string v)
```

Parameters

v [string ↗](#)

Returns

[string ↗](#)

VioletRed(string)

```
public static string VioletRed(string v)
```

Parameters

v [string](#)

Returns

[string](#)

Wheat(string)

```
public static string Wheat(string v)
```

Parameters

v [string](#)

Returns

[string](#)

White(string)

```
public static string White(string v)
```

Parameters

v [string](#)

Returns

[string](#)

WhiteSmoke(string)

```
public static string WhiteSmoke(string v)
```

Parameters

v [string](#)

Returns

[string](#)

Yellow(string)

```
public static string Yellow(string v)
```

Parameters

v [string](#)

Returns

[string](#)

YellowGreen(string)

```
public static string YellowGreen(string v)
```

Parameters

v [string](#)

Returns

[string](#)

YellowNice(string)

```
public static string YellowNice(string v)
```

Parameters

v [string](#) ↗

Returns

[string](#) ↗