SkylarkWsp.DanmakuEngine

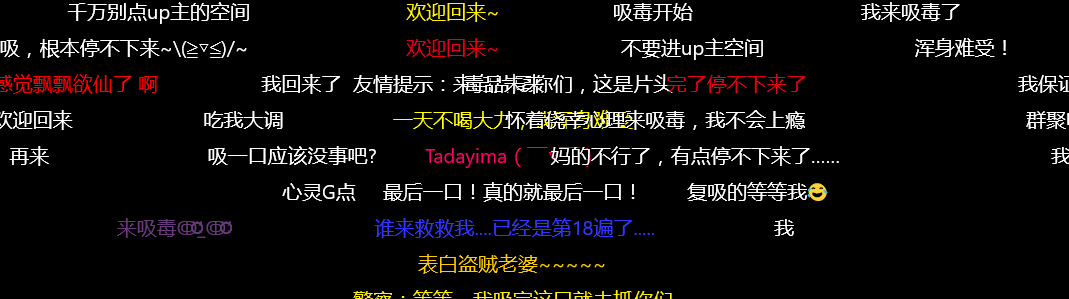
简介

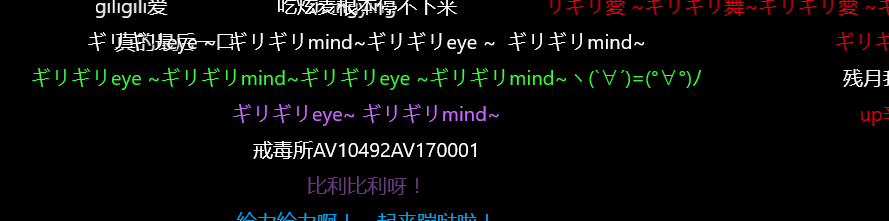
SkylarkWsp.DanmakuEngine（Codename HikariEngine）是一个为UWP平台设计的，弹幕引擎和控件库。由Skylark Workshop开发。

联系方式：

Kevin Gao：[gaojunxuanbox@live.com](mailto:gaojunxuanbox@live.com) （微博/Twitter：@Coding的高同学/@kevingjx）

Hao Ling：[lingmengc@outlook.com](mailto:lingmengc@outlook.com)（微博：@ Future\_Labs）





其中包含了

* DanmakuManager

提供了弹幕的布局、添加、控制等方法的实现

* DanmakuPresenter

一个根据DanmakuManager编写的弹幕容器，主体为一个Grid容器，用于显示弹幕

* DanmakuPlayer

一个可以根据弹幕Time属性定时显示弹幕的播放控件，主体由一个MediaElement、DanmakuPresenter和Logger组成。使用TimeLineMarker实现定时显示弹幕

* Logger

一个简单的日志记录控件，用于在屏幕上显示弹幕数据的加载信息

* DanmakuParser

包含两个Parser工厂类和一个工厂接口，用于将Bilibili和AcFun的弹幕数据成解析成引擎所使用的Danmaku类型

* ColorExtension

一个扩展类，包含两个扩展方法，用于将10进制颜色数据解析成Color类型

* ShadowTextBlock（已弃用）

一个包含阴影的TextBlock控件，引用自微软的Win2DSample，但由于渲染占用CPU较高和所需时间较长的缘故，并未在项目中使用，如果有需要，也可以根据需要使用。

* 实例应用和ギリギリEYE

一个演示DanmakuEngine使用方法的UWP应用，从Bilibili上获取样例弹幕数据并显示。（此应用内置了ギリギリEYE，请谨慎使用）

DanmakuManager

* 方法

1. 构造函数public DanmakuManager(Grid container)

将指定Grid作为容器，初始化DanmakuManager

1. private void ClearAllLine()

解除对所有行的占用

1. private int GetAvailableLine()

获取当前未被占用的行编号

1. private int GetAvailableLineForTopDanmaku()

为顶部弹幕获取可用行编号

1. private int GetAvailableLineForBottomDanmaku()

为底部弹幕获取可用行编号

1. private int GetY(int line)

根据提供的行编号，计算出弹幕所处位置的Y坐标值

1. private void \_container\_SizeChanged(object sender, SizeChangedEventArgs e)

响应控件大小变化

1. public void AddScrollableDanmaku(string text, Color foreground, double size, bool shadow)

添加一个滚动弹幕，text为弹幕内容，foreground为弹幕颜色，size为弹幕字体大小，shadow为指定是否为弹幕添加阴影的布尔值

1. public void AddScrollableDanmaku(string text, Color foreground, double size, int speed,bool shadow)

添加一个滚动弹幕，text为弹幕内容，foreground为弹幕颜色，size为弹幕字体大小，speed为弹幕运动速度，shadow为指定是否为弹幕添加阴影的布尔值

1. public async void AddTopMotionlessDanmaku(string text, Color foreground, double size,bool shadow)

添加位于屏幕顶端的固定弹幕。text为弹幕内容，foreground为弹幕颜色，size为弹幕字体大小，shadow为指定是否为弹幕添加阴影的布尔值

1. public async void AddBottomMotionlessDanmaku(string text, Color foreground, double size,bool shadow)

添加位于屏幕底端的固定弹幕。text为弹幕内容，foreground为弹幕颜色，size为弹幕字体大小，shadow为指定是否为弹幕添加阴影的布尔值

1. public void AddDanmaku(string text, Color foreground, double size,DanmakuMode mode,bool shadow,int speed=5000)

添加一条弹幕。text为弹幕内容，foreground为弹幕颜色，size为弹幕字体大小，mode为弹幕类型（见下），speed为弹幕运动速度，shadow为指定是否为弹幕添加阴影的布尔值

1. public void SwitchVisibility()

显示或隐藏容器

1. public void Pause()

暂停容器中正在运动的弹幕

1. public void Resume()

恢复被暂停运动的弹幕

1. public void Clear()

清空容器，解除对所有行的占用

1. public void SetLimit(int count)

限制容器中最大弹幕数量

1. private double GetDefaultFontSize(double size)

获取默认字体大小（未定）

* 枚举

1. DanmakuMode

指定弹幕类型，分别为：顶部固定、从右至左滚动、从左至右滚动（未实现）、底部固定、高级（未实现）、脚本（未实现）

1. DanmakuSource

指定了弹幕数据的源，分别为：AcFun、Bilibili、Niconico和其他

DanmakuPresenter

无特殊的方法，使用姿势和Manager类似。

DanmakuPlayer

* 方法

1. public void SetDanmakuSource(ObservableCollection<Danmaku> source)

将当前弹幕播放器的弹幕源设定为指定的ObservableCollection（正考虑使用List替代ObservableCollection）

1. public async void LoadDanmaku(string danmakuId,DanmakuSource source)

调用DanmakuParser获取弹幕数据

1. public void Play()

播放视频和弹幕

1. public void Stop()

停止播放视频（弹幕也不会继续加载）

1. public void Pause()

暂停视频和弹幕

1. public void WriteLine(string msg)

在屏幕日志上显示内容

1. public void SetLimit(int count)

设定DanmakuPresenter的最大弹幕数量限制

* 事件

1. 多数MediaElement自带的事件及其处理
2. public event RoutedEventHandler DanmakuLoadStart

在弹幕开始加载时触发

1. public event RoutedEventHandler DanmakuLoadComplete

在弹幕加载完成时触发

1. public event DanmakuExceptionRoutedEventHandler DanmakuLoadFailed

在弹幕加载出现错误或播放过程中因弹幕出现问题而引发异常时触发的事件

* 属性和Dependency Properties

1. 多数MediaElement的内置属性，例如Source、Volume等
2. ShowLogger

DependencyProperty，指定了是否显示日志控件

DanmakuFactory

1. 通过创建IDanmuFactory来获取弹幕Model，提供BiliDanmuFactory和AcDanmuFactory两个工厂类，同时可以继续拓展实例工厂类。

例如：

IDanmuFactory fac = new BiliDanmuFactory();

1. 通过接口提供的方法CreateDanMuModel来创建Model：

例如：

DanmuModelBase biliModel = await fac.CreateDanMuModel("5212471");

1. 可以向下转型来获得实例模型额外的数据：

例如：

BiliDanMuModel biliModel2 = (BiliDanMuModel)biliModel;

DanmakuParser

调用DanmakuFactory实现对弹幕数据的Parsing

* 方法

1. public static async Task<BiliDanmakuModel> GetBiliDanmaku(string id)

使用指定的ID从Bilibili获取弹幕数据

1. public static async Task<BiliDanmakuModel> GetBiliDanmaku(string id,string uri)

使用指定的ID从Bilibili获取弹幕数据，并使用指定URI作为弹幕源服务器地址

1. public static async Task<AcDanmakuModel> GetAcDanmaku(string id)

使用指定的ID从AcFun获取弹幕数据

1. public static async Task<AcDanmakuModel> GetAcDanmaku(string id,string uri)

使用指定的ID从AcFun获取弹幕数据，并使用指定URI作为弹幕源服务器地址

Danmaku类型

所有弹幕数据的类型，包含如下属性

1. public string Content { get; set; }

弹幕的内容

1. public IList<string> PositionData { get; set; }

弹幕的详细信息，例如位置、颜色等

1. public double Size { get; set; }

弹幕的文本大小

1. public int Speed { get; set; }

弹幕的运动速度

1. public double Time { get; set; }

弹幕出现的时间（以毫秒计）

1. public DanmakuMode Mode { get; set; }

弹幕模式

1. public Color ForegroundColor { get; set; }

弹幕的颜色

1. public string UserId { get; set; }

弹幕发送者的用户ID

1. public string DanmakuId { get; set; }

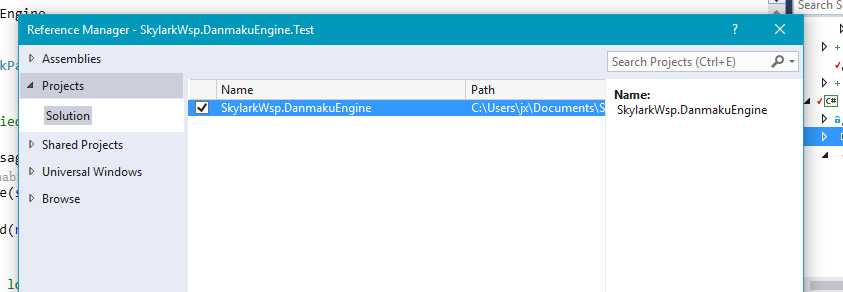
弹幕ID

1. public bool IsShowed { get; set; }

指示弹幕是否已经在DanmakuPlayer中显示过，以避免重复显示

如何使用

首先，在项目中引用SkylarkWsp.DanmakuEngine



在XAML中，引用SkylarkWsp.DanmakuEngine命名空间

xmlns:Danmaku="using:SkylarkWsp.DanmakuEngine"

添加如下XAML代码，来在UI中放置一个DanmakuPlayer，并设定其视频源

<Danmaku:DanmakuPlayer x:Name="player" Source="Assets/test-girigiri.mp4" />

在后台代码中为player设定弹幕源

调用 player.Play(); 开始播放

Contributing

各位菊苣，dalao啊，欢迎提出各种吐槽、意见和bug，您可以发送PULL请求来提交您的贡献，定会感激不尽orz。

Submit Pull Requests to contribute to this project.

**License**

MIT License

Copyright (c) 2016 Skylark Workshop

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.