该项目已使用MIT开源协议（https://opensource.org/licenses/mit-license.php）开源，软件开源地址：<https://github.com/ijrys/WorldCreater>

# 文档规范

## 字体及样式使用规范

不使用非商用免费字体

中文使用思源黑体、黑体、隶书等免费字体

英文使用consolas

## 文档内编程语句规范

非必要情况下，不出现代码语句，只展示定义。

类型使用浅色标注(rgb(128,128,128))。

定义的名称使用加粗字体。

私有内容使用斜体。

方法定义，若参数列表为空，括号与函数名保持在同一行；若参数列表不为空，参数列表单独一行。。

代码使用“代码”格式。

如：

public static string Fun1()

private string Fun2

(string str, int start)

单独的代码块使用外侧框线框起来，配合表格使用的代码段（如模块具体功能中的方法的声明和解释）可忽略外框线

## 注释规范

可使用中括号括起来的内容对内容做注释，并对内容以浅色颜色(rgb(128,128,128))

如

[已完成][优先]文档自动更新模块

* 统一注释：

[WPF]：使用了WPF技术的模块

[static]：静态的模块

[i]：interface，接口

[a]：abstract，抽象类

[FE]：前端工厂（FrontEndFactory）相关

[BE]：前端工厂（BackEndFactory）相关

# 软件需求说明

## 开发目的

整合三维地形生成及环境模拟运算，为非精确的大规模地形建模工程提供便利，为快速的地形生成算法做实验性研究平台。 并且基于此软件提供的平台实验自行设计的随机趋势化地形生成算法，为以后的实验提供便利。

## 软件功能要求

提供一个统一的、易扩展的平台环境，为相关算法的实验、应用提供一个可行的环境。 Studio要有对工程、项目和相关资源管理能力。

算法分为前端工厂部分和后端工厂部分。前端工厂为地形生成器，该部分要通过接收一组设定值而产生一个合理的二维的高度数据，作为地形的高度信息。后端工厂要根据用户的设定和其他后端工厂的产出数据，对前端工厂产出的地形进行合理的环境模拟。

软件将实现前端工厂的随机趋势化算法和后端工厂中的空气流动模拟、降水模拟及生态系统模拟。 Studio要求能对工厂产出的数据进行整合，并且可以导出为三维模型或是二维贴图，以方便后续对产出数据的使用。

## 开发技术及软件运行目标环境

### 开发技术

* C# （C#7.0）
* WPF （WPF4.5）

### 开发环境

* .NET Framework （.NET Framework 4.7.2）
* Visual Studio 2017 Community 及兼容产品

### 软件运行目标环境

* .NET Framework 4.7.2 及兼容版本
* Windows 10 1607及后支持.NET Framework 4.7.2及WPF的操作系统
* 显示器800 \* 600 及更大分辨率
* 内存：剩余1G以上可用的内存空间
* 磁盘：剩余2G以上可用的磁盘空间，磁盘可读写

## 参考文献

1. 《WPF编程宝典——使用c#2012和.NET 4.5》（第4版） （清华大学出版社 [美]Matthew MacDonald 著，王德才 译）

## 相关链接

#### C#7.0

<https://docs.microsoft.com/zh-cn/dotnet/csharp/whats-new/csharp-7>

#### WPF4.5

<https://docs.microsoft.com/zh-cn/dotnet/framework/wpf/getting-started/whats-new>

#### .NET Framework 4.7.2 Download Page

<https://dotnet.microsoft.com/download/dotnet-framework-runtime/net472>

#### Visual Studio

<https://visualstudio.microsoft.com/zh-hans/>

#### MIT开源协议

<https://opensource.org/licenses/mit-license.php>

# 软件逻辑设计

## 工程管理逻辑

软件工程逻辑为树形分层逻辑，根节点为工程节点（Project）。工程节点下分为多个工作节点（Work），每一个工作节点对应一个地形从生成到最终的环境模拟演变结束的整个过程。在同一工程内的各个工作是相互关联而又相互独立的，他们既可以是同一个项目的不同部分的地形，也可以通过更强大的合成器将多个地形合成到一起，实现更加复杂的效果控制。

工作分为图片资源管理器、前端工厂和后端工厂。图片资源管理器管理整个工作过程中产生的图片资源，进行统一管理。其他节点只能通过引用这些资源来进行图片资源的分类。

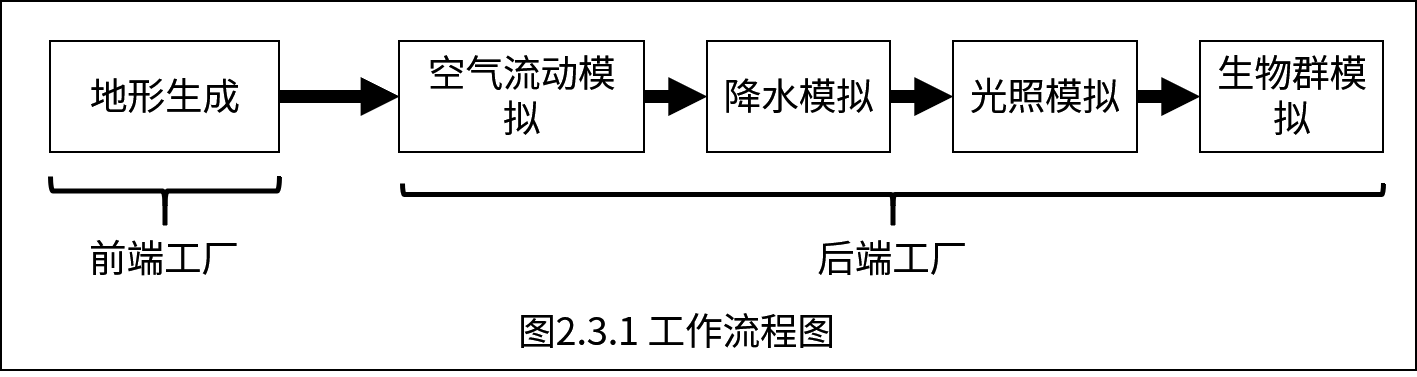
## 功能使用操作流程及方式

用户通过鼠标、键盘进行输入，并通过ui界面获得操作输出。

使用统一的ui平台对算法功能进行整合，用户通过控制ui平台来实现对算法的调用和数据的获取。

各算法按照算法目的分为五大部分，对应于工作的五个步骤。使用流水线工作方式，只有在依赖的前一工作点完成工作后才可使用当前工作点。

流程示意如下：



同一节点只依赖之前节点产生的数据，不依赖之前节点的数据生成算法类型，故可根据需求自行选择合适的算法或自行扩展。

# 软件数据存储结构及节点逻辑结构定义

## 工程文件存储格式

工程涉及到两种文件存储，为工程文件和工作文件。

### 工程文件

工程文件后缀名为mriwcpro（MiRaI World Creater Project），以标准xml文档格式存储。

工程文件记录了工程的基本信息（GUID、工程名）和工程拥有的工作信息（GUID、工作所在的文件夹、工作文件名，工作名）。

工程文件内容具体定义如下：

<project guid="[工程GUID]" name="[工程名称]">  
 <work guid="[工作1GUID]" dictionary="[工作1存储文件夹]" file="[工作1文件]" name="[工作1名]" />  
 <work guid="[工作2GUID]" dictionary="[工作2存储文件夹]" file="[工作2文件]" name="[工作2名]" />  
</project>

如

<project guid="c99a3e2b-58eb-47e4-1234-85cf72f19deb" name="DemoPro">  
 <work guid="46a1d881-f42d-4a7e-4567-0123456789ab" dictionary="demowork" file="demowork.mriwcw" name="测试工作集" />  
</project>

### 工作文件

工作文件后缀名为mriwcw（MiRaI World Creater Work），以标准xml文档格式存储。

工作文件记录了工作的基本信息（GUID）和工作拥有的图片资源(images节点)、工作的前端工厂信息、工作的后端工厂信息。

工作文件内容具体定义如下：

<work guid="[工作GUID]">  
 <images>  
 <image key="[资源1唯一键]" name="[资源1展示名称]" file="[资源1文件，根目录为工作文件夹下images文件夹]" description="[对资源1的描述信息]"/>  
 <image key="[资源2唯一键]" name="[资源2展示名称]" file="[资源2文件，根目录为工作文件夹下images文件夹]" description="[对资源2的描述信息]"/>  
 </images>  
  
 <FrontEndFactory creater="[记录Creater的ProgramSet信息]">  
 <setting>

[该节点为相应类产生其具体内容]

</setting>

<Resault name="" value="[运算结果的存储文件]" imgrefkey="[用于展示的图片资源key]" />

</FrontEndFactory>

<BackEndFactory>

<AMNode creater="[后端工厂的ProgramSet]" state="[节点的状态]">

<Resault dataName="[运算结果的存储文件]" imgrefkey="[用于展示的图片资源key]" />

<Config>

[该节点为相应类产生其具体内容]

</Config>

</AMNode>

</BackEndFactory>

</work>

如

<work guid="54fe1651-5c29-45c5-9a31-379e4b18b100">

<images>

<image key="FE.RandomValue" file="eec6e1f3e0a64d4c97962b64bf67.png" description="前端工厂的随机值图" />

<image key="FE.HeightValue" file="36089b82aacb4afa97544c5d5d8c.png" description="前端工厂高度值图" />

<image key="BE.AM.Map" file="684083b24a3a4b08ab8aa9c6da1c.png" description="AtmosphericMotion Visual Map" />

</images>

<FrontEndFactory creater="MiRaI.RandomTend.RandomTend|0.1">

<setting>

<add key="blockSize" value="5" />

<add key="wbn" value="1" />

<add key="hbn" value="1" />

<add key="seed" value="2019" />

</setting>

<Resault name="" value="HeightValue" imgrefkey="FE.HeightValue" />

</FrontEndFactory>

<BackEndFactory>

<AMNode creater="MiRaI.BE.AM.SV|0.1" state="ok">

<Resault dataName="AtmosphericMotion Visual Map" imgrefkey="BE.AM.Map" />

<Config power="76" dir="NW" />

</AMNode>

<RMNode creater="MiRaI.BE.RM.SV|0.1" state="ok">

<Resault dataName="RainfallMotion Visual Map" imgrefkey="BE.RM.RIMap" riimgrefkey="BE.RM.RIMap" atimgrefkey="BE.RM.ATMap" />

<Config ri="4020" sl="0" />

</RMNode>

<SINode creater="" state="ready" />

<BINode creater="" state="unable" />

</BackEndFactory>

</work>

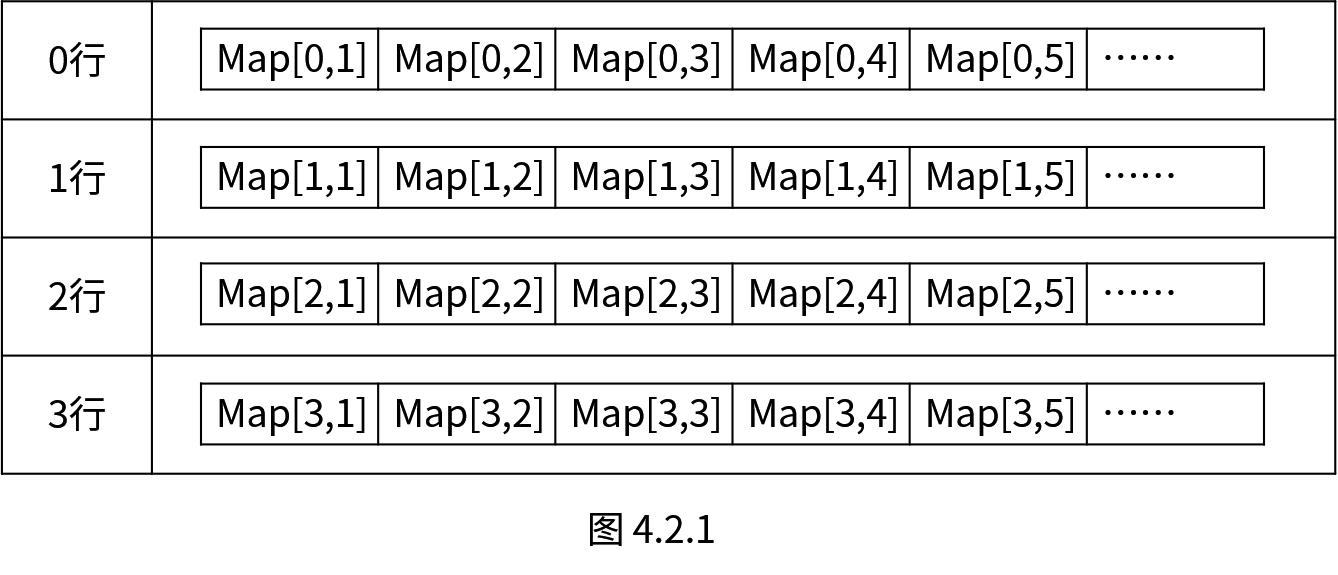
## 工程文件的物理存储逻辑

Project  
|-proj.mrimcproj [工程文件]  
|-Geo1  
| |-geo1.mrigeoproj [工作文件]  
| |-Images [工作中产生的图像资源]  
| | |-img1.png  
| | |-……  
| |-HeightValue [高度数据]  
| |-AtmosphericMotion Visual Map [空气流动数据]  
| |-RainfallMotion Visual Map [降数据]  
| |- [其他产生的过程数据]  
|-Geo2  
|-……

## 数据图结构定义

数据均为矩阵，内部操作时使用二维数组实现存储。

数组第一索引为行号，第二索引为列号，如下图(4.3.1)所示



数据存储时先存储数据图宽、高(int\_32)，再按行存储其中每一列数据。伪代码如下：

Write (map.Width);

Write (map.Height);

for (int h = 0; h < map.Height; h ++) {

for (int w = 0; w < map.Width; w ++) {

Write (map[h, w]);

}

}

数据图按0行在上，0列在左处理。方位上按上方为北，左方为西处理。

为了兼容各个流程，对于图的数据意义和长宽进行限制，规定如下：

* 生成的地形边长应均为n \* 2k，n、k均为正整数，且k>=4。
* 高度图中每个点代表的是一个正方形块的角的高度，则高度图的数据数量为(a + 1) \* (b + 1)，a、b分别为设定目标地图的长、宽。
* 后端工厂对数据的处理是基于块的，记录的是块的数据，则后端工厂生成的数据量应为 a \* b，a、b分别为设定目标地图的长、宽。

## 工程的展示逻辑

软件中对用户展示一个逻辑树，来展示整个工程。用户通过对逻辑树的相应节点进行操作从而控制整个工程。

Project【不可点击】  
|-Geo1【工程设置】  
| |-RandomMap【随机数图】  
| |-HeightMap【高度图】  
| | |-SubMap子图1  
| | |-SubMap子图2  
| |-TerrainMap【地势图】  
| |-LaterFactory【后期工厂】  
| |-AirMotion【空气流动图】  
| |-Precipitation【降水】  
| |-【水域】  
| |-【生物群】  
|-Geo2  
| |-RandomMap【随机数图】  
| |-HeightMap【高度图】  
| |-TerrainMap【地势图】  
| |-LaterFactory【后期工厂】  
| |-AirMotion【空气流动图】  
| |-Precipitation【降水】  
| |-【水域】  
| |-【生物群】

# 软件模块划分及关系

## 模块功能简介

### WorldCreaterStudio

软件UI提供模块，将各模块功能整合提供完整的操作流程，是用户与WorldCreaterStudio\_Core模块的桥梁。

### WorldCreaterStudio\_Core

定义整个软件平台的核心算法、核心类，并为算法功能模块定义基本的开发接口，为整个平台的基础和各个组件联通的纽带。

### WorldCreaterStudio\_Resouses

存储WorldCreaterStudio所使用到的资源。

### MiRaIUIProject

该模块为WPF框架的主题项目，基于WPF控件定义了一套视觉效果，现已独立为单独的项目，并已通过MIT协议开源。开原地址：<https://github.com/ijrys/MiRaIUIProject>

该模块为WorldCreaterStudio模块提供WPF程序的的UI样式。

### [FE]FrontEndFactorys.RandomTend

前端工厂相关算法的实现。随机趋势化算法核心代码。

### [BE]MiRaI.BE.AM.SingleValue

后端工厂空气运动模拟相关算法的实现。提供一个快速的、简单的实现，方便必要的需求和软件整体框架测试。

### [BE]MiRaI.BE.RM.SingleValue

后端工厂降水运动模拟相关算法的实现。提供一个快速的、简单的实现，方便必要的需求和软件整体框架测试。

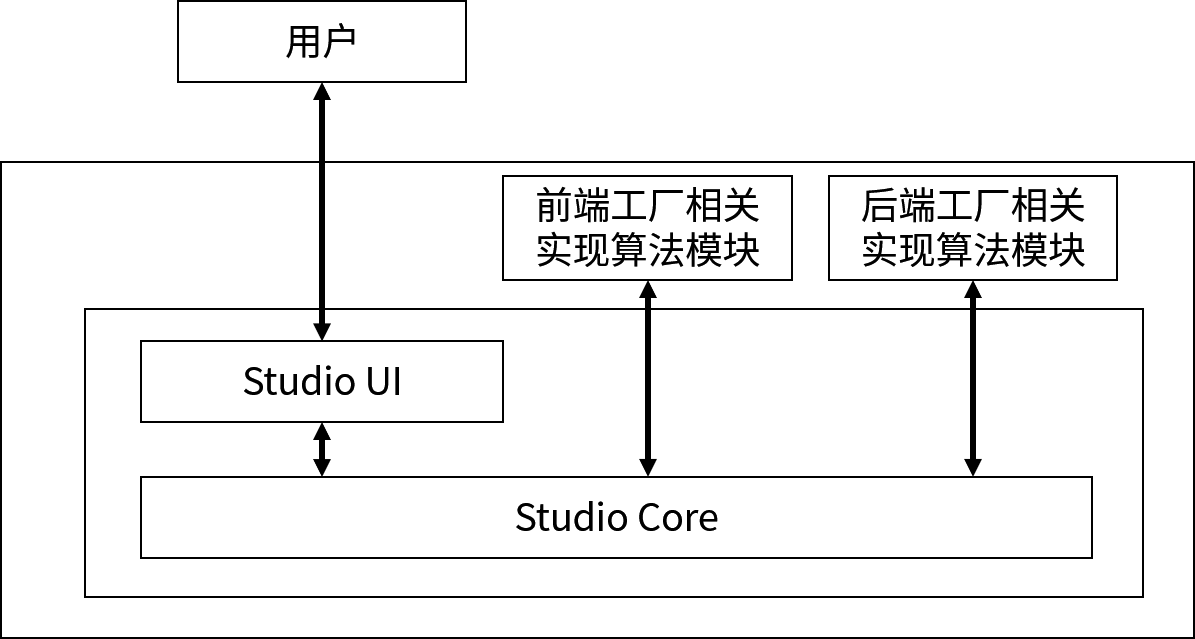
### [BE]MiRaI.BE.SI.QuickCalculating

后端工厂光照强度模拟相关算法的实现。提供一个快速的、简单的实现，方便必要的需求和软件整体框架测试。

### [BE]MiRaI.BE.BI.QuickCalc

后端工厂生物群模拟相关算法的实现。提供一个快速的、简单的实现，方便必要的需求和软件整体框架测试。

## 模块依赖关系图



# 代码规范规约

## 命名风格

* 程序集、命名空间、类名、方法名、属性名使用UpperCamelCase风格，如FrontEndFactory
* 参数名、成员变量、局部变量使用lowerCamelCase风格，如inputName
* 私有字段使用下划线开始，如\_nodeName
* 异常类命名使用Exception结尾
* 接口以I开头，以Able结尾，如IwriteToFileAble

## 代码格式

* 大括号的使用约定。

如果是大括号内为空，则简洁地写成{}即可，不需要换行；

如果是非空代码块则 左大括号前不换行，左大括号后换行。右大括号前换行。右大括号后还有else等代码则不换行；表示终止的右大括号后必须换行。

* 左右小括号和字符之间不出现空格。
* 缩进

级别缩进使用tab，统一级别的代码块应保证前边有相同数量的tab

内容缩进在使用tab实现级别缩进后按需使用空格进行内容对齐

* 换行

在一行代码过长时进行换行保证阅读方便。换行时应保证语句块不被拆分（如调用方法时不将方法名与参数列表拆在两行），调用操作符应跟在新行，其他操作符保持在旧行。

* 代码格式示例如下：

public static void Function (int id,

string name,

string nickName) {

if (id < 10 &&

name != null) {

// do something

} else {

nickName.DoFun1(' ')

.DoFun2("||")

.DoFun3();

}

}

# ValueToImage模块功能细则

## [static]ValueToGrayImage

将高度数据转换为可视的灰度图

### 方法

#### GetBitmap

public static WriteableBitmap GetBitmap(int minvalue, int maxvalue, byte mingray, byte maxgray, int[,] map)

获取一个灰度图。

|  |  |
| --- | --- |
| minvalue | 高度的最小值，小于或等于该值的将显示为最小亮度 |
| maxvalue | 高度的最大值，大于或等于该值的将显示为最大亮度 |
| mingray | 设定最小亮度 |
| maxgray | 设定最大亮度 |
| map | 高度数据 |

#### GetBitmapWithError

public static WriteableBitmap GetBitmapWithError (int minvalue, int maxvalue, int[,] map)

获取一个带有高度超出检查的灰度图，用于检查生成的地形高度数据是否在合法范围内。

|  |  |
| --- | --- |
| minvalue | 高度数据的下限，小于这个值的区域将设置为红色(255,0,0) |
| maxvalue | 高度数据的上限，大于这个值的区域将设置为青色(0,255,255) |
| map | 高度数据 |

#### GetPixel

public static byte GetPixel (int minvalue, int maxvalue, byte mingray, byte maxgray, int value)

获取一个值在高度范围内的转换结果

|  |  |
| --- | --- |
| minvalue | 高度的最小值，小于或等于该值的将返回mingray |
| maxvalue | 高度的最大值，大于或等于该值的将返回maxgray |
| mingray | 设定最小亮度 |
| maxgray | 设定最大亮度 |
| value | 高度数据 |

## [static]ValueToColorImage

将高度数据转换为可视的彩色图

### 方法

#### **GetBitmap**

public static WriteableBitmap GetBitmap (int minvalue, int maxvalue, int[,] map)

获取一张彩色图。海平面值为0。

|  |  |
| --- | --- |
| minvalue | 高度数据的下限，小于这个值的区域将视为minvalue |
| maxvalue | 高度数据的上限，大于这个值的区域将视为maxvalue |
| map | 高度数据 |

## [static]AtmosphericMotionToImage

配合AtmosphericMotion的值转换器。

### 方法

#### GetBitmap

public static WriteableBitmap GetBitmap (PointData[,] datas)

获取区域风向图

|  |  |
| --- | --- |
| datas | 风向数据 |

## [static]RainfallMotionToImage

配合RainfallMotion的值转换器。

### 方法

#### GetRainfallIntensityBitmap

public static WriteableBitmap GetRainfallIntensityBitmap (PointData[,] datas)

获取降雨强度可视化图。

|  |  |
| --- | --- |
| datas | 原始数据 |

#### GetAreaTpyeBitmap

public static WriteableBitmap GetAreaTpyeBitmap (PointData[,] datas)

获取区域类型可视化图

|  |  |
| --- | --- |
| datas | 原始数据 |

#### GetWaterDeepBitmap

public static WriteableBitmap GetWaterDeepBitmap (PointData[,] datas)

获取水深可视化图

|  |  |
| --- | --- |
| datas | 原始数据 |

## SolarIlluminance

配合SolarIlluminance的值转换器。

### 方法

#### GetBitmap

public static WriteableBitmap GetBitmap (byte[,] value)

获取光照强度图，返回值为Gray8

|  |  |
| --- | --- |
| value | 原始数据 |

## BiomesToImage

配合BiomesToImage的值转换器。

### 方法

#### GetBitmap

public static WriteableBitmap GetBitmap (BiomesType[,] value)

获取生物分布图

|  |  |
| --- | --- |
| value | 原始数据 |

# BackEndFactorys/AtmosphericMotion/MiRaI.BE.AM.SingleValue模块功能细则

## SingleValue

使用单值快速填充的空气运动模拟器。

继承：IAtmosphericMotionCalculaterAble

### 属性

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 名称 | 类型 | 可赋值 | 可获取 | 默认值 | 描述 |
| CreaterName | string | F | T | "单一值风力设定" | 获取模拟器名称 |
| CreaterProgramSet | string | F | T | "MiRaI.BE.AM.SV|0.1" | 获取模拟器程序集 |
| CreaterGuid | Guid | F | T | typeof (SingleValue).GUID | 获取模拟器GUID |

### 委托

#### OnProcessingChanged

public event DataCalculatingProcessingEventType OnProcessingChanged;

模拟过程产生新进度消息时事件

### 方法

#### GetAtmosphericMotionDatasBySpecialConfig

public AtmosphericMotionResault GetAtmosphericMotionDatasBySpecialConfig (SingleValueConfig config, int[,] heightMap, Work work)

使用SingleValueConfig类型的配置项进行模拟

|  |  |
| --- | --- |
| config | 配置设置 |
| heightMap | 高度数据 |
| work | 执行方法的工作集 |

#### GetAtmosphericMotionDatas

public AtmosphericMotionResault GetAtmosphericMotionDatas (IAtmosphericMotionConfigAble config, int[,] heightMap, Work work)

使用一个IAtmosphericMotionConfigAble类型的配置项进行模拟。

若不为SingleValueConfig类型，抛出IncongruentConfigurationException

|  |  |
| --- | --- |
| config | 配置设置 |
| heightMap | 高度数据 |
| work | 执行方法的工作集 |

## SingleValueConfig

记录SingleValue进行模拟所需要的配置数据

继承：IAtmosphericMotionConfigAble

### 属性

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 名称 | 类型 | 可赋值 | 可获取 | 默认值 | 描述 |
| Direction | Direction | T | T | "单一值风力设定" | 获取模拟器名称  会改变九个IsToX的属性的值 |
| IsToC | bool | T | T |  | 【不推荐对其进行赋值】  用于配合绑定前端交互的九个按钮的属性。  值会互相影响，并且会改变Direction的值。  set不会主动触发任何事件，但通过改变Direction的值来触发相应事件 |
| IsToNW | bool | T | T |  |
| IsToN | bool | T | T |  |
| IsToNE | bool | T | T |  |
| IsToE | bool | T | T |  |
| IsToSE | bool | T | T |  |
| IsToS | bool | T | T |  |
| IsToSW | bool | T | T |  |
| IsToW | bool | T | T |  |
| Power | byte | T | T |  | 获取或设置风力 |
| DisplayPower | double | F | T | Power/20.0 | 获取用于界面展示的风力 |
| ShowPanel | ControlTemplate | F | T |  | 获取相配套的界面模板 |

### 委托

#### ValueChanged

public event NodeValueChangedEventType ValueChanged

内容发生更改时事件

#### PropertyChanged

public event PropertyChangedEventHandler PropertyChanged

属性发生更改时通知绑定目标

### 方法

#### LoadFromXMLNode

public void LoadFromXMLNode (XmlElement xmlnode)

从XML节点中加载配置数据

|  |  |
| --- | --- |
| xmlnode | 数据来源的XML节点 |

#### XmlNode

public XmlElement XmlNode (XmlDocument xmlDocument, bool save = false)

获取配置的XML描述节点

|  |  |
| --- | --- |
| xmlDocument | XML节点所在文档的根节点 |
| save | 是否为保存动作 |

## SingleValueFactory

### 属性

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 名称 | 类型 | 可赋值 | 可获取 | 默认值 | 描述 |
| DisplayName | string | F | T | 单一值风力设定 | 展示用的名称，与Calculater的CreaterName相同 |
| DisplayType | string | F | T | BE.AM | 展示所属类型 |
| CalculaterProgramSet | string | F | T | MiRaI.BE.AM.SV|0.1 | 获取模拟器的程序集，与Calculater的CreaterProgramSet相同 |
| CalculaterGuid | Guid | F | T | typeof (SingleValue).GUID | 获取模拟器的GUID，与Calculater的CreaterGuid相同 |

### 方法

#### GetACalculater

public IAtmosphericMotionCalculaterAble GetACalculater ()

获取一个模拟器

#### GetAConfiguration

public IAtmosphericMotionConfigAble GetAConfiguration ()

获取一个相应的配置对象

# BackEndFactorys/RainfallMotion/MiRaI.BE.RM.SingleValue模块功能细则

## SingleValue

使用单值快速填充的降水运动模拟器。

继承：IRainfallMotionCalculaterAble

### 属性

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 名称 | 类型 | 可赋值 | 可获取 | 默认值 | 描述 |
| CreaterName | string | F | T | "单一值快速设定" | 获取模拟器名称 |
| CreaterProgramSet | string | F | T | "MiRaI.BE.RM.SV|0.1" | 获取模拟器程序集 |
| CreaterGuid | Guid | F | T | typeof (SingleValue).GUID | 获取模拟器GUID |

### 委托

#### OnProcessingChanged

public event DataCalculatingProcessingEventType OnProcessingChanged

模拟过程产生新进度消息时事件

### 方法

#### GetAtmosphericMotionDatasBySpecialConfig

public AtmosphericMotionResault GetAtmosphericMotionDatasBySpecialConfig (SingleValueConfig config, int[,] heightMap, Work work)

使用SingleValueConfig类型的配置项进行模拟

|  |  |
| --- | --- |
| config | 配置设置 |
| heightMap | 高度数据 |
| work | 执行方法的工作集 |

#### GetAtmosphericMotionDatas

public AtmosphericMotionResault GetAtmosphericMotionDatas (IAtmosphericMotionConfigAble config, int[,] heightMap, Work work)

使用一个IAtmosphericMotionConfigAble类型的配置项进行模拟。

若不为SingleValueConfig类型，抛出IncongruentConfigurationException

|  |  |
| --- | --- |
| config | 配置设置 |
| heightMap | 高度数据 |
| work | 执行方法的工作集 |

## SingleValueConfig

记录SingleValue进行模拟所需要的配置数据

继承：IRainfallMotionConfigAble

### 属性

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 名称 | 类型 | 可赋值 | 可获取 | 默认值 | 描述 |
| RainfallIntensity | int | T | T |  | 获取或设置降水强度。每单位时间降水0.01个全局高度单位 |
| DisplayRainfallIntensity | double | F | T | RainfallIntensity / 100 | 用于展示的真实降水强度 |
| SeaLevel | int | T | T |  | 获取或设置海平面高度，与高度图高度同单位 |
| ShowPanel | ControlTemplate | F | T |  | 获取相配套的界面模板 |

### 委托

#### ValueChanged

public event NodeValueChangedEventType ValueChanged

内容发生更改时事件

#### PropertyChanged

public event PropertyChangedEventHandler PropertyChanged

属性发生更改时通知绑定目标

### 方法

#### LoadFromXMLNode

public void LoadFromXMLNode (XmlElement xmlnode)

从XML节点中加载配置数据

|  |  |
| --- | --- |
| xmlnode | 数据来源的XML节点 |

#### XmlNode

public XmlElement XmlNode (XmlDocument xmlDocument, bool save = false)

获取配置的XML描述节点

|  |  |
| --- | --- |
| xmlDocument | XML节点所在文档的根节点 |
| save | 是否为保存动作 |

## SingleValueFactory

用于创建、获取相应的Calculater和Config的工厂类

继承：IRainfallMotionCalculaterFactoryAble

### 属性

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 名称 | 类型 | 可赋值 | 可获取 | 默认值 | 描述 |
| DisplayName | string | F | T | "单一值快速设定" | 展示用的名称，与Calculater的CreaterName相同 |
| DisplayType | string | F | T | "BE.RM" | 展示所属类型 |
| CalculaterProgramSet | string | F | T | "MiRaI.BE.RM.SV|0.1" | 获取模拟器的程序集，与Calculater的CreaterProgramSet相同 |
| CalculaterGuid | Guid | F | T | typeof (SingleValue).GUID | 获取模拟器的GUID，与Calculater的CreaterGuid相同 |

### 方法

#### GetACalculater

public IAtmosphericMotionCalculaterAble GetACalculater ()

获取一个模拟器

#### GetAConfiguration

public IAtmosphericMotionConfigAble GetAConfiguration ()

获取一个相应的配置对象

# BackEndFactorys/SolarIlluminance/MiRaI.BE.SI.QuickCalculating模块功能细则

## QuickCalculating

利用法线快速模拟光照强度模拟器。

继承：ISolarIlluminanceCalculaterAble

### 属性

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 名称 | 类型 | 可赋值 | 可获取 | 默认值 | 描述 |
| CreaterName | string | F | T | "反射面法线快速计算" | 获取模拟器名称 |
| CreaterProgramSet | string | F | T | "MiRaI.BE.SI.QC|0.1" | 获取模拟器程序集 |
| CreaterGuid | Guid | F | T | typeof (QuickCalculating).GUID | 获取模拟器GUID |

### 委托

#### OnProcessingChanged

public event DataCalculatingProcessingEventType OnProcessingChanged

模拟过程产生新进度消息时事件

### 方法

#### GetSolarIlluminanceResaultDatasBySpecialConfig

public SolarIlluminanceResault GetSolarIlluminanceResaultDatasBySpecialConfig (QuickCalculatingConfig config, int[,] heightMap, Work work)

使用QuickCalculatingConfig类型的配置项进行模拟

|  |  |
| --- | --- |
| config | 配置设置 |
| heightMap | 高度数据 |
| work | 执行方法的工作集 |

#### GetAtmosphericMotionDatas

public SolarIlluminanceResault GetSolarIlluminanceDatas (ISolarIlluminanceConfigAble config, int[,] heightMap, Work work)

使用一个ISolarIlluminanceConfigAble类型的配置项进行模拟。

若不为ISolarIlluminanceConfigAble类型，抛出IncongruentConfigurationException

|  |  |
| --- | --- |
| config | 配置设置 |
| heightMap | 高度数据 |
| work | 执行方法的工作集 |

## QuickCalculatingConfig

记录QuickCalculating进行模拟所需要的配置数据

继承：ISolarIlluminanceConfigAble

### 属性

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 名称 | 类型 | 可赋值 | 可获取 | 默认值 | 描述 |
| Angle | double | T | T |  | 获取或设置光照角度 |
| ShowPanel | ControlTemplate | F | T |  | 获取相配套的界面模板 |

### 委托

#### ValueChanged

public event NodeValueChangedEventType ValueChanged

内容发生更改时事件

#### PropertyChanged

public event PropertyChangedEventHandler PropertyChanged

属性发生更改时通知绑定目标

### 方法

#### LoadFromXMLNode

public void LoadFromXMLNode (XmlElement xmlnode)

从XML节点中加载配置数据

|  |  |
| --- | --- |
| xmlnode | 数据来源的XML节点 |

#### XmlNode

public XmlElement XmlNode (XmlDocument xmlDocument, bool save = false)

获取配置的XML描述节点

|  |  |
| --- | --- |
| xmlDocument | XML节点所在文档的根节点 |
| save | 是否为保存动作 |

## QuickCalculatingFactory

用于创建、获取相应的Calculater和Config的工厂类

继承：IRainfallMotionCalculaterFactoryAble

### 属性

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 名称 | 类型 | 可赋值 | 可获取 | 默认值 | 描述 |
| DisplayName | string | F | T | "反射面法线快速计算" | 展示用的名称，与Calculater的CreaterName相同 |
| DisplayType | string | F | T | "BE.SI" | 展示所属类型 |
| CalculaterProgramSet | string | F | T | "MiRaI.BE.SI.QC|0.1" | 获取模拟器的程序集，与Calculater的CreaterProgramSet相同 |
| CalculaterGuid | Guid | F | T | typeof (QuickCalculating).GUID | 获取模拟器的GUID，与Calculater的CreaterGuid相同 |

### 方法

#### GetACalculater

public IsolarIlluminanceCalculaterAble GetACalculater ()

获取一个模拟器

#### GetAConfiguration

public IsolarIlluminanceConfigAble GetAConfiguration ()

获取一个相应的配置对象

# BackEndFactorys/Biomes/MiRaI.BE.BI.QuickCalc模块功能细则

## QuickCalc

根据已有快速模拟光照强度模拟器。

继承：IBiomesCalculaterAble

### 属性

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 名称 | 类型 | 可赋值 | 可获取 | 默认值 | 描述 |
| CreaterName | string | F | T | "快速运算" | 获取模拟器名称 |
| CreaterProgramSet | string | F | T | " MiRaI.BE.BI.QC|0.1" | 获取模拟器程序集 |
| CreaterGuid | Guid | F | T | typeof (QuickCalc).GUID | 获取模拟器GUID |

### 委托

#### OnProcessingChanged

public event DataCalculatingProcessingEventType OnProcessingChanged

模拟过程产生新进度消息时事件

### 方法

#### GetBiomesDatasBySpecialConfig

public BiomesResault GetBiomesDatasBySpecialConfig (QuickCalcConfig config, int[,] heightMap, Work work)

使用QuickCalcConfig类型的配置项进行模拟

|  |  |
| --- | --- |
| config | 配置设置 |
| heightMap | 高度数据 |
| work | 执行方法的工作集 |

#### GetBiomesDatas

public BiomesResault GetBiomesDatas (IBiomesConfigAble config, int[,] heightMap, Work work)

使用一个IBiomesConfigAble类型的配置项进行模拟。

若不为QuickCalcConfig类型，抛出IncongruentConfigurationException

|  |  |
| --- | --- |
| config | 配置设置 |
| heightMap | 高度数据 |
| work | 执行方法的工作集 |

## QuickCalcConfig

记录QuickCalc进行模拟所需要的配置数据

继承：IBiomesConfigAble

### 属性

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 名称 | 类型 | 可赋值 | 可获取 | 默认值 | 描述 |
| ShowPanel | ControlTemplate | F | T |  | 获取相配套的界面模板 |

### 委托

#### ValueChanged

public event NodeValueChangedEventType ValueChanged

内容发生更改时事件

#### PropertyChanged

public event PropertyChangedEventHandler PropertyChanged

属性发生更改时通知绑定目标

### 方法

#### LoadFromXMLNode

public void LoadFromXMLNode (XmlElement xmlnode)

从XML节点中加载配置数据

|  |  |
| --- | --- |
| xmlnode | 数据来源的XML节点 |

#### XmlNode

public XmlElement XmlNode (XmlDocument xmlDocument, bool save = false)

获取配置的XML描述节点

|  |  |
| --- | --- |
| xmlDocument | XML节点所在文档的根节点 |
| save | 是否为保存动作 |

## QuickCalcFactory

用于创建、获取相应的Calculater和Config的工厂类

继承：IBiomesCalculaterFactoryAble

### 属性

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 名称 | 类型 | 可赋值 | 可获取 | 默认值 | 描述 |
| DisplayName | string | F | T | "快速运算" | 展示用的名称，与Calculater的CreaterName相同 |
| DisplayType | string | F | T | "BE.BI" | 展示所属类型 |
| CalculaterProgramSet | string | F | T | "MiRaI.BE.BI.QC|0.1" | 获取模拟器的程序集，与Calculater的CreaterProgramSet相同 |
| CalculaterGuid | Guid | F | T | typeof (QuickCalc).GUID | 获取模拟器的GUID，与Calculater的CreaterGuid相同 |

### 方法

#### GetACalculater

public IsolarIlluminanceCalculaterAble GetACalculater ()

获取一个模拟器

#### GetAConfiguration

public IsolarIlluminanceConfigAble GetAConfiguration ()

获取一个相应的配置对象

# FrontEndFactorys/RandomTend模块功能细则

## RandomTendCreater

随机趋势化地形生成算法

继承：MapCreater

### 属性

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 名称 | 类型 | 可赋值 | 可获取 | 默认值 | 描述 |
| CreaterProgramSet | string | F | T | "MiRaI.RandomTend.RandomTend|0.1" |  |
| CreaterGuid | Guid | F | T | typeof(RandomTendCreater).GUID |  |

### 方法

#### CreatAMap

public override WorldCreaterStudio\_Core.FrontendNode.CreatingResault CreatAMap(Configuration configuration, WorldCreaterStudio\_Core.Work work)

利用配置进行地图创建

若configuration不是想要的配置类型，抛出异常WorldCreaterStudio\_Core.Exceptions.IncongruentConfigurationException

|  |  |
| --- | --- |
| configuration | 配置对象 |
| work | 目标工作集 |

## RTConfiguration

RandomTendCreater的配置类型

继承：Configuration

### 属性

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 名称 | 类型 | 可赋值 | 可获取 | 默认值 | 描述 |
| BlockSize | int | T | T | 3 | 获取或设置块大小，块边长为(2^value + 1)。值为[3,16]。 |
| WidthBlockNum | int | T | T | 1 | 横向块数量 |
| HeightBlockNum | int | T | T | 1 | 纵向块数量 |
| Width | int | F | T | (9) | 获取地图宽度 |
| Height | int | F | T | (9) | 获取地图高度 |
| MaxBlockNum | int | F | T | 1 << (maxLength - 3) | 获取最大可设定的块数量 |
| RandomSeed | int | T | T | 2019 | 随机值种子 |
| ShowPanel | ControlTemplate | F | T |  | 配套的展示面板 |

### 委托

public override event PropertyChangedEventHandler PropertyChanged

### 方法

#### LoadFromXMLNode

public override void LoadFromXMLNode(XmlElement xmlnode)

从XML节点中加载配置

|  |  |
| --- | --- |
| xmlnode | 数据来源的XML节点 |

#### XmlNode

public override XmlElement XmlNode(XmlDocument xmlDocument, bool save = false)

获取配置的XML描述节点

|  |  |
| --- | --- |
| xmlDocument | XML节点所在文档的根节点 |
| save | 是否为保存动作 |

#### GetWidth

public override int GetWidth()

获取地图宽度

#### GetHeight

public override int GetHeight()

获取地图高度

#### GetRandomSeed

public override int GetRandomSeed()

获取地图随机种子

#### ToString

public override string ToString()

用于测试使用

## RandomTendCreaterFactory

用于创建、获取相应的MapCreater和Configuration的工厂类

继承：MapCreaterFactory

### 属性

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 名称 | 类型 | 可赋值 | 可获取 | 默认值 | 描述 |
| DisplayName | string | F | T | " Random Tend" | 获取展示用的名称 |
| DisplayType | string | F | T | "Random Tend" | 获取展示所属类型 |
| CalculaterProgramSet | string | F | T | "MiRaI.RandomTend.RandomTend|0.1" | 获取创建器的程序集，与MapCreater的CreaterProgramSet相同 |
| CalculaterGuid | Guid | F | T | typeof (RandomTendCreater).GUID | 获取创建器的GUID，与MapCreater的CreaterGuid相同 |

### 方法

#### GetACreater

public override MapCreater GetACreater()

获取一个创建器

#### GetAConfiguration

public override Configuration GetAConfiguration()

获取一个相应的配置对象

# WorldCreaterStudio模块功能细则

## [static]Commands

Studio使用到的自定义命令

### 构造函数

static Commands()

### 属性

|  |  |
| --- | --- |
| 声明 | 描述 |
| public static RoutedUICommand NewWork | 命令·新建工作 |
| public static RoutedUICommand NewProject | 命令·新建工程 |

## [static]Resouses/StoreRoom

### 属性

|  |  |
| --- | --- |
| 声明 | 描述 |
| public static List<NewWork.MapCreaterTypeNode> **MapCreaterCollection** | 获取注册到程序中的获取MapCreaterTypeNode列表 |
| private static Dictionary<string, NewWork.MapCreaterTypeNode> **\_**mapCreaterTypeMayToCollection | 【私有】存储MapCreaterTypeNode与其Type的关联规则，方便根据Type查找MapCreaterTypeNode |
| private static Dictionary<string, WorldCreaterStudioCore.MapCreater.MapCreaterFactory> programSetToCreaterFactory | 【私有】存储Creater与其ProgramSet的对应规则，方便根据ProgramSet查找Creater |

### 方法

|  |  |
| --- | --- |
| 声明 | 描述 |
| public static void RegisterACreaterFactory  (WorldCreaterStudio\_Core.MapCreater.MapCreaterFactory createrFactory) | 注册一个CreaterFactory  createrFactory：要注册的CreaterFactory |
| [DEL]public static WorldCreaterStudio\_Core.MapCreater.MapCreaterFactory GetACreaterFactory  (string programSet) | 【已删除】根据programSet获取一个CreaterFactory  programSet：所根据的programSet字符串 |

## [static]Resouses/Icons

提供所有用到的的图标资源。

### 属性

|  |  |
| --- | --- |
| 名称 | 描述 |
| public static BitmapSource DarkIconPro | 暗色主题 工程图标 |
| public static BitmapSource DarkIconWork | 暗色主题 工作图标 |
| public static BitmapSource DarkIconRes | 暗色主题 资源图标 |
| public static BitmapSource DarkIconResLib | 暗色主题 资源库图标 |
| public static BitmapSource DarkIconBackEndWork | 暗色主题 后端工作图标 |

## [WPF]Resouses/ControlTemplates [WPF]Resouses/Theme

MiRaIUIProject模块提供的内容

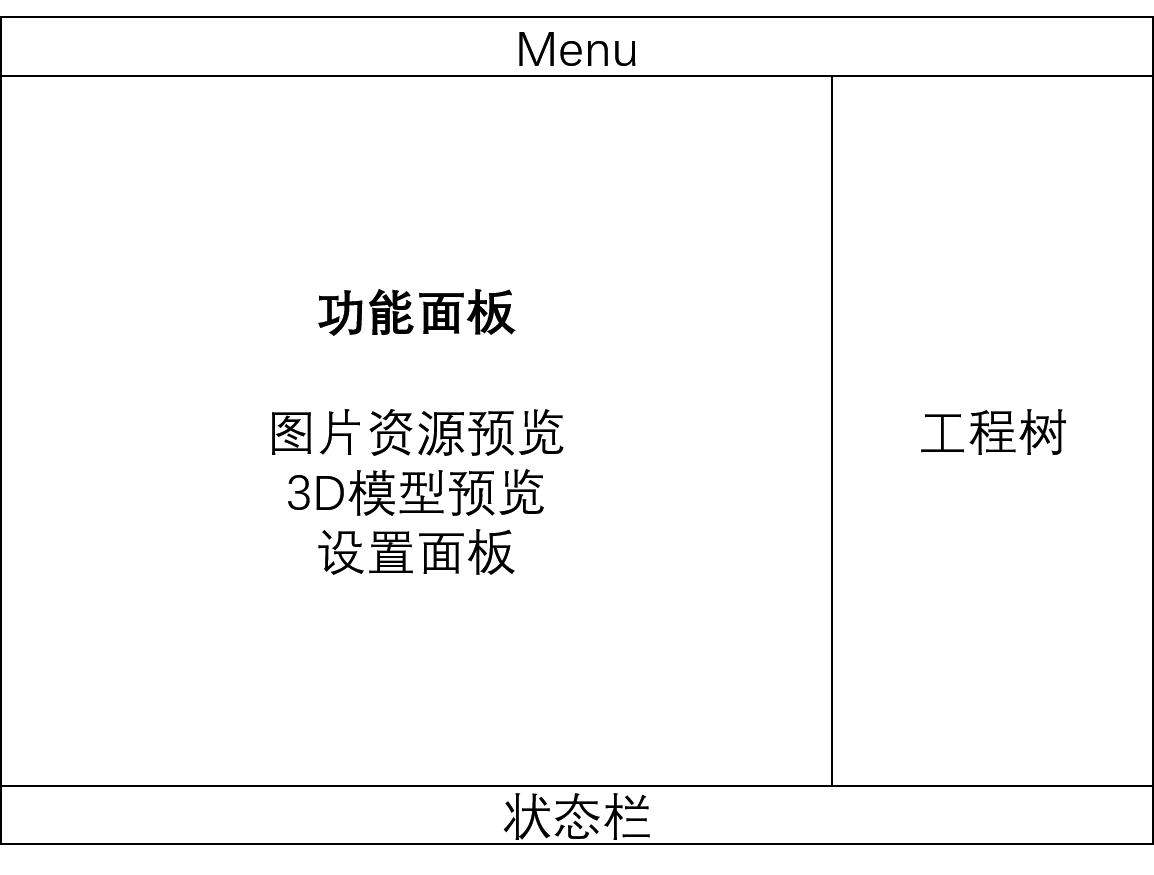
## [WPF]Windows/MainWindow

主功能窗体

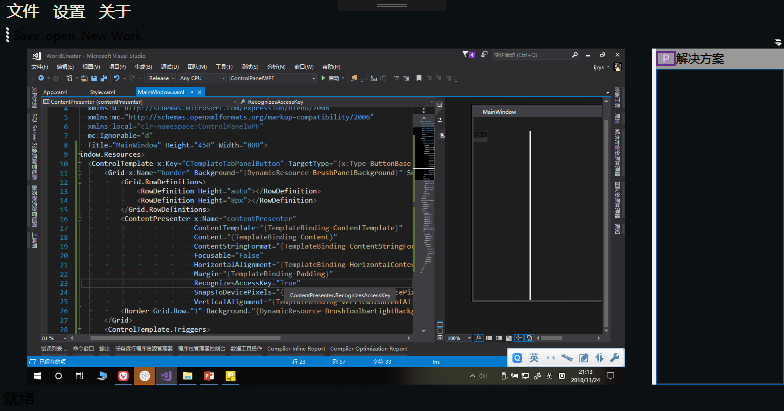
* 继承：Window

### UI设计

#### 原型设计图



#### 最终产品图



### 委托和事件

|  |  |
| --- | --- |
| 名称 | 描述 |
| public delegate void ProjectChangedEventType  (object sender, Project value); | 属性Project改变的Event基本类  sender：触发改变的对象  value：新的Project |
| public event ProjectChangedEventType ProjectChanged; | 属性Project改变后触发的事件 |

### 字段和属性

|  |  |
| --- | --- |
| 名称 | 描述 |
| private Project \_project | 【私有|字段】保存当前工程对象 |
| public Project Project | 获取和设置当前工程，并在更改后执行相应事件（ProjectChanged） |
| private FrameworkElement \_showingFunctionPanel | [功能面板相关]表示正在展示的面板 |

### 方法

|  |  |
| --- | --- |
| 名称 | 描述 |
| private void CommandNewWorkExecuted (object sender, ExecutedRoutedEventArgs e) | 命令NewWork执行的方法  sender：调用者  e：参数 |
| private void CommandNewProjectExecuted (object sender, ExecutedRoutedEventArgs e) | 命令NewProject执行的方法  sender：调用者  e：参数 |
| private void ProjectChangedFunction(Project value) | 更改属性Project后必然执行的函数，用于更新前端的数据绑定  Value：新的Project |
| private void ShowFunctionPanel  (object dataprovider, FrameworkElement panel) | [功能面板相关]展示一个功能面板 |
| private void Tree\_Project\_Item\_DoubleClick  (object sender, MouseButtonEventArgs e) | [功能面板相关]Tree\_Project\_Item的双击时间 |
| private void RegShowPanel() | [功能面板相关]向Core注册展示板 |

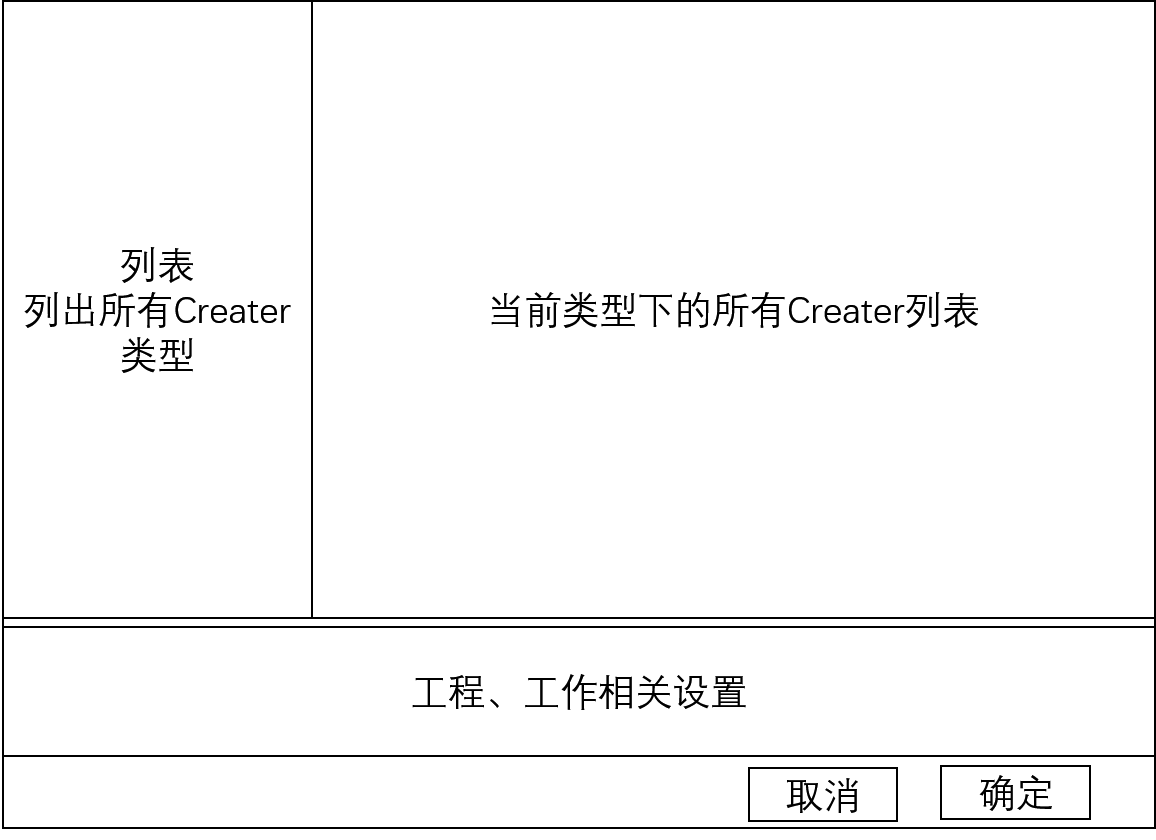
## [WPF]Windows/NewProject

新建工程、工作窗体 用于选择新建的工程、工作的类型，确定名称和存储位置等

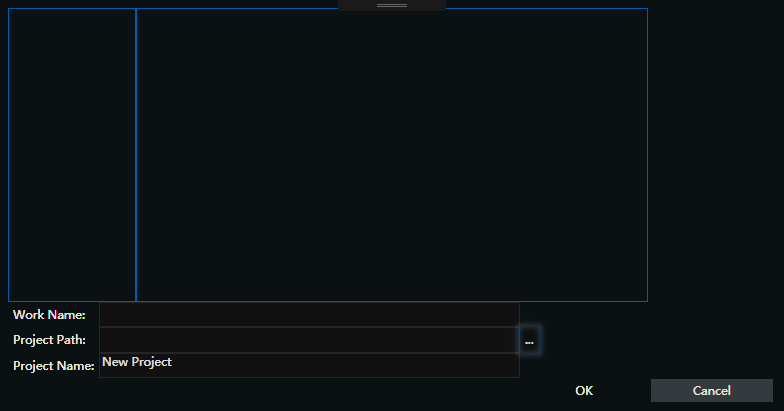
* 继承：Window

### UI设计

* 原型设计图



* 最终产品图



### 字段和属性

|  |  |
| --- | --- |
| 名称 | 描述 |
| private Project **Project** | 【私有】用于保存新创建的工程 |
| private Work **Work** | 【私有】用于保存新创建的工作 |
| private bool **CreateNewProject** | 【私有】 |
| public DialogResult **WindowResult** | 用于调用者获取用户最终选择的结果 |

### 方法

|  |  |
| --- | --- |
| 名称 | 描述 |
| public new void Show() | 重写了父类的Show方法，自动调用ShowDialog()  不建议使用，请合理选择使用GetNewProject()与GetNewWork() |
| public new void ShowDialog() | 重写了父类的ShowDialog方法，用于以独占方式弹出窗体  不建议使用，请合理选择使用GetNewProject()与GetNewWork() |
| private void ShowDialog (Project project) | 用于以独占方式弹出窗体，并记录传入的Project对象。用于创建一个新工作时确定工作所属的工程  Project：所属的工程 |
| public Project GetNewProject() | 获取一个新的工程，会根据需要自动弹出窗体 |
| public Work GetNewWork  (Project project) | 在已有的工程中创建一个新工作，会根据需要自动弹出窗体。  project：目标工程 |
| private void BtnDirSelect\_Click  (object sender, RoutedEventArgs e) | 前端按钮BtnDirSelect点击事件 |
| private void BtnOk\_Click  (object sender, RoutedEventArgs e) | 前端按钮BtnOk点击事件 |
| private void BtnCancel\_Click  (object sender, RoutedEventArgs e) | 前端按钮BtnCancel点击事件 |
| private void ListCreaterTypeSelectionChanged  (object sender, SelectionChangedEventArgs e) | 前端列表List\_CreaterType选择项目改变事件 |

# WorldCreaterStudio\_Core模块功能细则

## 子命名空间或逻辑文件夹功能描述

* ListNode

逻辑节点文件夹，内部元素直接存在于WorldCreaterStudio\_Core命名空间下。逻辑节点类型，提供主功能节点和逻辑节点接口。

* MapCreater

FrontendFactory的Creater相关基类的定义。

* FrontendNode

用于定义FrontendFactory的相关节点。

* BackendNode

BackendFactory的逻辑节点的相关定义，内包含AtmosphericMotion、RainfallMotion、SolarIlluminance、Biomes四个文件夹，用以区分四个过程的节点。

* Resouses

资源节点和资源管理管理节点的定义。

* StoreRoom

应用储藏室，存储各种注册到应用中的数据。

* Tools

各种工具类。

* Exceptions

用于定义所有所有应用可能用到的异常类型。

## [I]ListNode/IWorkLogicNodeAble

表示在程序中的逻辑节点，用以搭建层级逻辑关系。

继承：INotifyPropertyChanged

### 字段和属性

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 名称 | 类型 | 可赋值 | 可获取 | 默认值 | 描述 |
| Work | Work | N/A | T |  | 获取节点所在的工作 |
| ShowPanel | ControlTemplate | N/A | T |  | 获取节点可展示的面板 |
| NodeName | string | N/A | T |  | 获取节点在工程树形图中的展示名称 |
| Icon | ImageSource | N/A | T |  | 获取节点在工程树形图中的展示图标 |
| Childrens | ObservableCollection<IWorkLogicNodeAble> | N/A | T |  | 获取节点的子节点 |
| Changed | bool | N/A | T |  | 获取节点是否有值发生了改变 |

### 事件

#### NodeValueChanged

event NodeValueChangedEventType NodeValueChanged

节点值发生改变事件，用于通知需要保存

### 方法

#### XmlNode

XmlElement XmlNode(XmlDocument xmlDocument, bool save = false)

获取节点的XML节点

|  |  |
| --- | --- |
| xmlDocument | XML节点所在文档的根节点 |
| save | 是否为保存动作 |

## ListNode/Project

表示一个工程

对于工程文件的存储信息详见工程文件格式及存储结构=>工程文件格式

继承：IWorkLogicNodeAble

### 字段和属性

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 名称 | 类型 | 可赋值 | 可获取 | 默认值 | 描述 |
| [IWorkLogicNodeAble]Work | Work | F | T | null | 获取节点所属的工作 |
| [IWorkLogicNodeAble]ShowPanel | ControlTemplate | F | T | null | 获取节点可展示的面板 |
| [IWorkLogicNodeAble]NodeName | string | F | T |  | 获取节点在工程树形图中的展示名称 |
| [IWorkLogicNodeAble]Icon | ImageSource | F | T |  | 获取节点在工程树形图中的展示图标 |
| [IWorkLogicNodeAble]Childrens | ObservableCollection<IWorkLogicNodeAble> | F | T |  | 获取节点的子节点 |
| Guid | Guid | F | T |  | 获取工程的GUID |
| ProjectDirectionary | DirectoryInfo | F | T |  | 获取工程所在文件夹信息 |
| ProjectFile | FileInfo | F | T |  | 获取工程的文件信息 |
| [IWorkLogicNodeAble]Changed | bool | F | T |  | 获取工程是否发生了更改 |

### 事件

#### NodeValueChanged

public event NodeValueChangedEventType NodeValueChanged

节点值发生改变事件

#### PropertyChanged

public event PropertyChangedEventHandler PropertyChanged

属性发生改变时通知绑定元素

### 构造函数和静态获取方法

#### [private]Project

private Project(string projectPath, string filename, string proName)

构造函数

|  |  |
| --- | --- |
| projectPath： | 工程文件夹路径 |
| filename | 工程文件名 |
| proName | 工程名 |

#### OpenProject

public static Project OpenProject(string projectPath, string filename)

打开一个存在于存储器上的工程

|  |  |
| --- | --- |
| projectPath： | 工程文件夹路径 |
| filename | 工程文件名 |

#### NewProject

public static Project NewProject(string projectPath, string filename, string projectName)

在存储器上新建一个工程

|  |  |
| --- | --- |
| projectPath | 工程文件夹路径 |
| filename | 工程文件名 |
| proName | 工程名 |

### 方法

#### XmlNode

XmlElement XmlNode(XmlDocument xmlDocument, bool save = false)

获取节点的XML节点

|  |  |
| --- | --- |
| xmlDocument | XML节点所在文档的根节点 |
| save | 是否为保存动作 |

#### Save

public void Save(bool saveEvenUnchanged = false)

保存工程到本地文件

|  |  |
| --- | --- |
| saveEvenUnchanged | 指示在工程没有发生更改时是否强制保存。false：不强制保存,true：强制保存。 |

#### NewWork

public Work NewWork(string workPath, string filename, string workName)

在工程内新建一个工作

|  |  |
| --- | --- |
| projectPath： | 工程文件夹路径 |
| filename | 工程文件名 |
| proName | 工程名 |

## ListNode/Work

表示一个工作

继承：IWorkLogicNodeAble, InotifyPropertyChanged

### 属性和字段

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 名称 | 类型 | 可赋值 | 可获取 | 默认值 | 描述 |
| [private]\_workDirectionary | DirectoryInfo | N/A | N/A |  | 记录工作目录 |
| [private]\_workResousesDirectionary | DirectoryInfo | N/A | N/A |  | 记录工作的图片资源存储目录 |
| [private]\_workFile | FileInfo | N/A | N/A |  | 记录工作文件 |
| [private]\_guid | Guid | N/A | N/A |  | 工作的GUID |
| [private]\_changed | bool | N/A | N/A |  | 记录工作是否发生了更改 |
| [private]\_icon | ImageSource | N/A | N/A |  | 记录节点的图标 |
| [private]\_nodeName | string | N/A | N/A |  | 记录节点的名称 |
| Dispatcher | Dispatcher | F | T |  | 获取创建线程的工作队列 |
| WorkDirectionary | DirectoryInfo | F | T | \_workDirectionary | 获取工作目录信息 |
| Images | Resouses.ImageResourceManager | F | T | \_images | 获取工作的图片资源管理器 |
| FrontEndNodes | FrontEndFactory | F | T | \_frontEndNodes | 获取前端工厂节点 |
| BackEndNodes | BackEndFactory | F | T | \_backEndNodes | 获取后端工厂节点 |
| Guid | Guid | F | T | \_guid | 获取工作的GUID |
| [IWorkLogicNodeAble]Work | Work | F | T | null | 获取节点所属的工作 |
| [IWorkLogicNodeAble]ShowPanel | ControlTemplate | F | T | null | 获取节点可展示的面板 |
| [IWorkLogicNodeAble]NodeName | string | F | T |  | 获取节点在工程树形图中的展示名称 |
| [IWorkLogicNodeAble]Icon | ImageSource | F | T |  | 获取节点在工程树形图中的展示图标 |
| [IWorkLogicNodeAble]Childrens | ObservableCollection<IWorkLogicNodeAble> | F | T |  | 获取节点的子节点 |
| [IWorkLogicNodeAble]Changed | bool | F | T |  | 获取工程是否发生了更改 |

### 事件

#### NodeValueChanged

public event NodeValueChangedEventType NodeValueChanged

节点值发生改变事件

#### PropertyChanged

public event PropertyChangedEventHandler PropertyChanged

属性发生改变时通知绑定元素

### 构造函数和静态获取方法

#### Work

private Work(string workPath, string filename, string workName)

构造函数

|  |  |
| --- | --- |
| workPath | 工作文件夹路径 |
| filename | 工作文件名 |
| proName | 工作名 |

#### NewWork

public static Work NewWork(string workPath, string filename, string workName)

在存储器上创建一个新工作

|  |  |
| --- | --- |
| workPath | 工作文件夹路径 |
| filename | 工作文件名 |
| workName | 工作名 |

#### NewWork

public static Work NewWork(string workPath, string filename, string workName, MapCreater.MapCreaterFactory createrFactory)

在存储器上创建一个新工作

|  |  |
| --- | --- |
| workPath | 工作文件夹路径 |
| filename | 工作文件名 |
| workName | 工作名 |
| createrFactory | 前端工厂的地形创建器生产工厂 |

#### OpenWork

public static Work OpenWork(string workPath, string filename, string workName)

打开一个已有的工作

|  |  |
| --- | --- |
| workPath | 工作文件夹路径 |
| filename | 工作文件名 |
| workName | 工作名 |

### 方法

#### XmlNode

XmlElement XmlNode(XmlDocument xmlDocument, bool save = false)

获取节点的XML节点

|  |  |
| --- | --- |
| xmlDocument | XML节点所在文档的根节点 |
| save | 是否为保存动作 |

#### Save

public void Save(bool saveEvenUnchanged = false)

保存工作到存储器

|  |  |
| --- | --- |
| saveEvenUnchanged | 指示在工程没有发生更改时是否强制保存。false：不强制保存,true：强制保存。 |

#### [private]GetBitmapFromFile

private static BitmapImage GetBitmapFromFile(string path)

从文件中加载一张图片

|  |  |
| --- | --- |
| path | 图片路径 |

## ListNode/FrontEndFactory

前端工厂管理节点。

继承：IWorkLogicNodeAble

### 委托和事件

|  |  |
| --- | --- |
| 名称 | 描述 |

### 属性和字段

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 名称 | 类型 | 可赋值 | 可获取 | 默认值 | 描述 |
| Configuration | MapCreater.Configuration | F | T |  | 获取用于生成地形的配置对象 |
| Creater | MapCreater.MapCreater | F | T |  | 获取用于地形生成的生成器 |
| ImageReferenceManager | ImageResourceReferenceManager | F | T |  | 获取前端工厂的图片资源引用管理器 |
| ResaultHeightMap | FrontendNode.CreatingResault | F | T |  | 获取生产的结果 |
| [IWorkLogicNodeAble]Work | Work | F | T |  | 获取节点所属的工作 |
| [IWorkLogicNodeAble]ShowPanel | ControlTemplate | F | T | StoreRoom.ShowPanel.FrontEndFactoryPanel | 获取节点可展示的面板 |
| [IWorkLogicNodeAble]NodeName | string | F | T | "前端工厂" | 获取节点在工程树形图中的展示名称 |
| [IWorkLogicNodeAble]Icon | ImageSource | F | T | WorldCreaterStudio\_Resouses.Images.Dark\_Icon\_FrontEndWork | 获取节点在工程树形图中的展示图标 |
| [IWorkLogicNodeAble]Childrens | ObservableCollection<IWorkLogicNodeAble> | F | T | null | 获取节点的子节点 |
| [IWorkLogicNodeAble]Changed | bool | F | T |  | 获取工程是否发生了更改 |

### 事件

#### NodeValueChanged

public event NodeValueChangedEventType NodeValueChanged

节点值发生改变事件

#### PropertyChanged

public event PropertyChangedEventHandler PropertyChanged

属性发生改变时通知绑定元素

### 构造函数和静态获取方法

#### FrontEndFactory

public FrontEndFactory (Work parentWork)

构造函数

|  |  |
| --- | --- |
| parentWork | 所属的工作 |

#### LoadFromXmlNode

public static FrontEndFactory LoadFromXmlNode (XmlElement xmlnode, Work parentWork)

从XML节点中获取一个FrontEndFactory

|  |  |
| --- | --- |
| xmlnode | 信息来源的XML节点 |
| parentWork | 所属的工作 |

### 方法

#### XmlNode

XmlElement XmlNode(XmlDocument xmlDocument, bool save = false)

获取节点的XML节点

|  |  |
| --- | --- |
| xmlDocument | XML节点所在文档的根节点 |
| save | 是否为保存动作 |

#### InitByXMLNode

public bool InitByXMLNode (XmlElement xmlnode)

使用xml节点初始化前端工厂

|  |  |
| --- | --- |
| xmlnode | 信息来源的XML节点 |

#### Image\_Add

public void Image\_Add (string key, BitmapSource image, string description = "")

添加一个图片资源

|  |  |
| --- | --- |
| key | 资源的键 |
| image | 资源 |
| description | 描述信息 |

#### SetCreater

public void SetCreater (string programSet)

设置地形生成器

|  |  |
| --- | --- |
| programSet | 生成器的程序集 |

#### SetCreater

public void SetCreater (MapCreater.MapCreaterFactory createrFactory)

设置地形生成器

|  |  |
| --- | --- |
| createrFactory | 生成器的生成工厂 |

#### CreateAMap

public FrontendNode.CreatingResault CreateAMap ()

执行地形创建过程

#### Children\_NodeValueChanged

private void Children\_NodeValueChanged (IWorkLogicNodeAble node)

子节点值发生改变时事件方法

|  |  |
| --- | --- |
| node | 来源节点 |

#### Configuration\_ValueChanged

private void Configuration\_ValueChanged ()

配置节点值发生改变时事件方法