
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/WorldCreator>

一、文档规范

1.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

2.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(rgba(128,128,128,0.5))。

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

3.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(rgba(128,128,128,0.5))

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

二、软件简介

1.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。

2.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。

3.开发技术及软件运行目标环境

a) 开发技术

- C#（C#7.0）
- WPF（WPF4.5）

-
- .NET Framework (.NET Framework 4.7.2)

b) 开发环境

- Visual Studio 2017 Community

c) 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 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

4.参考文献

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）

5.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>

三、软件逻辑设计

1.工程管理逻辑

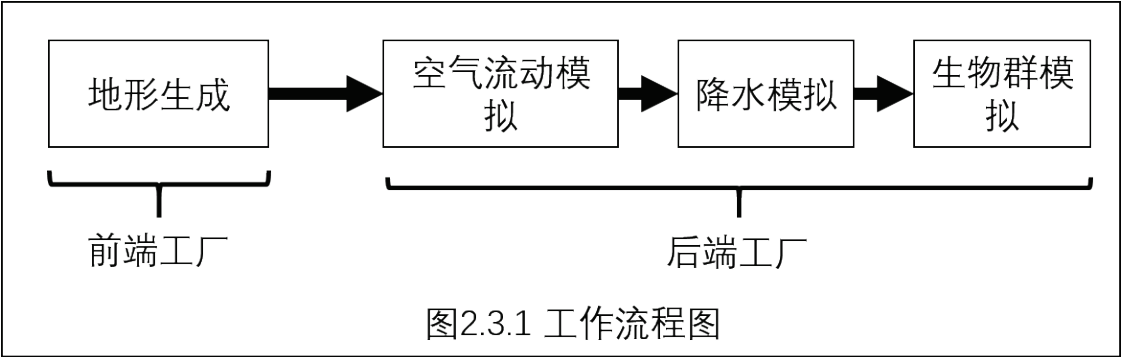
软件工程逻辑为树形分层逻辑，根节点为工程节点（Project）。工程节点下分为多个工作节点（Work），每一个工作节点对应一个地形从生成到最终的环境模拟演变结束的整个过程。在同一工程内的各个工作是相互关联而又相互独立的，他们既可以是同一个项目的不同部分的地形，也可以通过更强的合成器将多个地形合成到一起，实现更加复杂的效果控制。

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。

2.功能使用操作方式

软件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：



同一节点只依赖前一节点产生的数据，不依赖前一节点的算法类型，故可根据需求自行选择合适的算法或自行扩展。

四、工程文件格式及存储结构

工程是相关工作的集合，包含一个或多个工作，这些工作彼此相互独立但又相互联系。

工作是一个设定参数的具体实施过程和相关资源的管理和集合。每一个工作都可以产生一块地形，并且根据流程进行后端的环境模拟及结果的导出。

1.工程文件格式

工程涉及到两种文件存储，为工程文件和工作文件。

a) 工程文件

工程文件后缀名为 **mriwcpro**（MiRaI World Creator 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-47e4-9f1a-85cf72f19deb" name="DemoPro">
  <work guid="46a1d881-f42d-4a7e-a1cf-f779325cc017" dictionary="demowork"
file="demowork.mriwcw" name="测试工作集" />
</project>
```

b) 工作文件

工作文件后缀名为 **mriwcw**（MiRaI World Creator 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 creator="[记录 Creator 的 ProgramSet 信息]">
    <setting>
      [该节点为相应类产生其具体内容]
    </setting>
  </FrontEndFactory>
</work>
```

```

    <images>
        <image key="[引用的资源 key]">
    </images>
</FrontEndFactory>

<BackEndFactory>
    <Factory name="[后端工厂展示名称]" progress="[该后端工厂所处理的内容]"
factory="[后端工厂的 ProgramSet]">
    [该节点为相应类产生其具体内容]
    <images>
        <image key="[引用的资源]">
    </images>
    </Factory>
</BackEndFactory>
</work>

```

如

```

<work guid="">
    <images>
        <image key="res_1" name="this is a name" file="path base on project
flodar/images" description="description"/>
        <image key="res_2" name="Random Map" file="f-random.png"
description="the radom value of whole map"/>
    </images>

    <FrontEndFactory creator="MiRaI.RandomTend.RandomTend|0.1">
        <setting>
            <add key="" value="" />
        </setting>
        <images>
            <image key="res_1">
        </images>
    </FrontEndFactory>

    <BackEndFactory>
        <Factory name="空气运动" progress="AirMotion"
factory="BE.MiRaI.AirMotion|0.1">
            <data file="airmotion.data" />
            <images>
                <image key="res_2">
            </images>
        </Factory>
    </BackEndFactory>
</work>

```

2.工程文件的存储逻辑

```
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
|  |  |-.....
|  |-RandomMap.dataraw [随机数图]
|  |-HeightMap.dataraw [高度图]
|  |-TerrainMap.dataraw [地势图]
|  |-AirMotion.dataraw [空气流动图]
|  |-Precipitation.dataraw [降水]
|  |- [其他产生的过程数据]
|-Geo2
|-.....
```

3.工程的展示逻辑

```
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】
```

五、模块简介及功能概览

1.WorldCreatorStudio

a) 模块功能

软件 UI 提供模块，将各模块功能整合提供完整的操作流程。

2.WorldCreatorStudio_Core

a) 模块功能

WorldCreatorStudio 模块的核心算法、核心类，并为算法功能模块定义基本的开发接口。

3.WorldCreatorStudio_Resouses

a) 模块功能

存储 WorldCreatorStudio 所使用到的资源。

4.^[FE]FrontEndFactorys.RandomTend

a) 模块功能

随机趋势化算法核心代码。且实现 WorldCreatorStudio_Core 模块定义的接口。

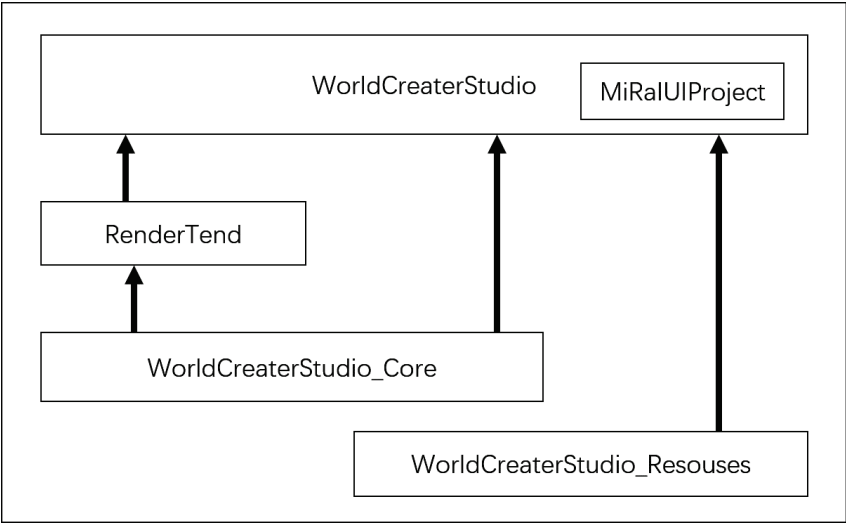
5.MiRaUIProject

该模块现已独立为单独的工程，并已通过 MIT 协议开源。开原地址：
<https://github.com/ijrys/MiRaUIProject>

b) 模块功能

提供 WPF 程序的 UI 样式。

6.模块依赖关系图



六、ValueToImage 模块功能细则

1.[static]ValueToGrayImage

将高度数据转换为可视的灰度图

a) 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

2.[static] ValueToColorImage

将高度数据转换为可视的彩色图

a) 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
-----	------

3.[static]AtmosphericMotionToImage

配合 AtmosphericMotion 的值转换器。

a) 方法

● GetBitmap

```
public static WriteableBitmap GetBitmap (PointData[,] datas)
```

获取区域风向图

datas	风向数据
-------	------

4.[static]RainfallMotionToImage

配合 RainfallMotion 的值转换器。

a) 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
-------	------

5.SolarIlluminance

配合 SolarIlluminance 的值转换器。

a) 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
-------	------

6.BiomesToImage

配合 BiomesToImage 的值转换器。

a) 方法

● GetBitmap

```
public static WriteableBitmap GetBitmap (BiomesType[,] value)
```

获取生物分布图

value	原始数据
-------	------

七、BackEndFactorys/AtmosphericMotion/MiRaI.BE.AM.SingleValue

e 模块功能细则

1.SingleValue

使用单值快速填充的空气运动模拟器。

继承：IAtmosphericMotionCalculaterAble

a) 属性

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

b) 委托

● OnProcessingChanged

```
public event DataCalculatingProcessingEventType OnProcessingChanged;
```

模拟过程产生新进度消息时事件

c) 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

2.SingleValueConfig

记录 SingleValue 进行模拟所需要的配置数据

继承：IAtmosphericMotionConfigAble

a) 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

b) 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

c) 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

3.SingleValueFactory

a) 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 Calculator 的

					CreatorName 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 CreatorProgramSet 相同
CalculaterGuid	Guid	F	T	typeof (SingleValue).GUID	获取模拟器的 GUID，与 Calculater 的 CreatorGuid 相同

b) 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

八、 BackEndFactorys/RainfallMotion/**MiRaI.BE.RM.SingleValue**
模块功能细则

1.SingleValue

使用单值快速填充的降水运动模拟器。
继承：IRainfallMotionCalculaterAble

a) 属性

名称	类型	可赋值	可获取	默认值	描述
CreatorName	string	F	T	"单一值快速设定"	获取模拟器名称

CreatorProgramSet	string	F	T	"MiRaI.BE.RM.SV 0.1"	获取模拟器程序集
CreatorGuid	Guid	F	T	typeof (SingleValue).GUID	获取模拟器 GUID

b) 委托

● OnProcessingChanged

```
public event DataCalculatingProcessingEventType OnProcessingChanged
```

模拟过程产生新进度消息时事件

c) 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

2.SingleValueConfig

记录 SingleValue 进行模拟所需要的配置数据

继承：IRainfallMotionConfigAble

a) 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

b) 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

c) 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 xmlDoc, 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

3.SingleValueFactory

用于创建、获取相应的 Calculator 和 Config 的工厂类

继承：IRainfallMotionCalculatorFactoryAble

a) 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 Calculator 的 CreatorName 相同
DisplayType	string	F	T	"BE.RM"	展示所属类型
CalculatorProgramSet	string	F	T	"MiRaI.BE.RM.SV 0.1"	获取模拟器的程序集，与 Calculator 的 CreatorProgramSet 相同
CalculatorGuid	Guid	F	T	typeof (SingleValue).GUID	获取模拟器的 GUID，与 Calculator 的 CreatorGuid 相同

b) 方法

● GetACalculator

```
public IAtmosphericMotionCalculatorAble GetACalculator ()
```

获取一个模拟器

● GetAConfiguration

```
public IAtmosphericMotionConfigAble GetAConfiguration ()
```

获取一个相应的配置对象

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

1.QuickCalculating

利用法线快速模拟光照强度模拟器。

继承：ISolarIlluminanceCalculatorAble

a) 属性

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

b) 委托

● OnProcessingChanged

```
public event DataCalculatingProcessingEventType OnProcessingChanged
```

模拟过程产生新进度消息时事件

c) 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

2.QuickCalculatingConfig

记录 QuickCalculating 进行模拟所需要的配置数据

继承：ISolarIlluminanceConfigAble

a) 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

b) 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

c) 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 xmlDoc, 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

3.QuickCalculatingFactory

用于创建、获取相应的 Calculator 和 Config 的工厂类

继承：IRainfallMotionCalculatorFactoryAble

a) 属性

名称	类 型	赋 值	获 取	默认值	描述
----	--------	--------	--------	-----	----

		值	取		
DisplayName	string	F	T	"反射面法线快速计算"	展示用的名称，与 Calclater 的 CreatorName 相同
DisplayType	string	F	T	"BE.SI"	展示所属类型
CalclaterProgramSet	string	F	T	"MiRaI.BE.SI.QC 0.1"	获取模拟器的程序集，与 Calclater 的 CreatorProgramSet 相同
CalclaterGuid	Guid	F	T	typeof (QuickCalculating).GUID	获取模拟器的 GUID，与 Calclater 的 CreatorGuid 相同

b) 方法

● GetACalclater

```
public IsolarIlluminanceCalclaterAble GetACalclater ()
```

获取一个模拟器

● GetAConfiguration

```
public IsolarIlluminanceConfigAble GetAConfiguration ()
```

获取一个相应的配置对象

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

1.QuickCalc

根据已有快速模拟光照强度模拟器。
继承：IBiomesCalclaterAble

a) 属性

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

b) 委托

● OnProcessingChanged

```
public event DataCalculatingProcessingEventType OnProcessingChanged
```

模拟过程产生新进度消息时事件

c) 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

2.QuickCalcConfig

记录 QuickCalc 进行模拟所需要的配置数据

继承：IBiomesConfigAble

a) 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

b) 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

c) 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

3.QuickCalcFactory

用于创建、获取相应的 Calculator 和 Config 的工厂类

继承：IBiomesCalculatorFactoryAble

a) 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 Calculator 的 CreatorName 相同
DisplayType	string	F	T	"BE.BI"	展示所属类型
CalculatorProgramSet	string	F	T	"MiRaI.BE.BI.QC 0.1"	获取模拟器的程序集，与 Calculator 的 CreatorProgramSet 相同
CalculatorGuid	Guid	F	T	typeof (QuickCalc).GUID	获取模拟器的 GUID，与 Calculator 的 CreatorGuid 相同

b) 方法

● GetACalculator

```
public IsolarIlluminanceCalculatorAble GetACalculator ()
```

获取一个模拟器

● GetAConfiguration

```
public IsolarIlluminanceConfigAble GetAConfiguration ()
```

获取一个相应的配置对象

十一、FrontEndFactorys/RandomTend 模块功能细则

1.RandomTendCreator

随机趋势化地形生成算法

继承：MapCreator

a) 属性

名称	类型	可赋值	可获得	默认值	描述
CreatorProgramSet	string	F	T	"MiRaI.RandomTend.RandomTend 0.1"	
CreatorGuid	Guid	F	T	typeof(RandomTendCreator).GUID	

b) 方法

● CreatAMap

```
public override WorldCreatorStudio_Core.FrontendNode.CreatingResault
CreatAMap(Configuration configuration, WorldCreatorStudio_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
WorldCreatorStudio_Core.Exceptions.IncongruentConfigurationException

configuration	配置对象
work	目标工作集

2.RTConfiguration

RandomTendCreator 的配置类型

继承：Configuration

a) 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

b) 委托

```
public override event PropertyChangedEventHandler PropertyChanged
```

c) 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

3.RandomTendCreaterFactory

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

继承：MapCreaterFactory

d) 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
CalclaterProgramSet	string	F	T	"MiRaI.RandomTend.RandomTend 0.1"	获取创建器的程序集，与 MapCreator 的 CreatorProgramSet 相同
CalclaterGuid	Guid	F	T	typeof (RandomTendCreator).GUID	获取创建器的 GUID，与 MapCreator 的 CreatorGuid 相同

e) 方法

● GetACreator

```
public override MapCreator GetACreator()
```

获取一个创建器

● GetAConfiguration

```
public override Configuration GetAConfiguration()
```

获取一个相应的配置对象

十二、WorldCreatorStudio 模块功能细则

1.[static]Commands

Studio 使用到的自定义命令

a) 构造函数

static Commands()

b) 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

2.[static]Resouses/StoreRoom

a) 属性

声明	描述
public static List<NewWork.MapCreatorTypeNode> MapCreatorCollection	获取注册到程序中的获取 MapCreatorTypeNode 列表
private static Dictionary<string, NewWork.MapCreatorTypeNode>	【私有】存储

<code>_mapCreatorTypeMayToCollection</code>	MapCreatorTypeNode 与其 Type 的关联规则，方便根据 Type 查找 MapCreatorTypeNode
<code>private static Dictionary<string, WorldCreatorStudioCore.MapCreator.MapCreatorFactory> programSetToCreatorFactory</code>	【私有】存储 Creator 与其 ProgramSet 的对应规则，方便根据 ProgramSet 查找 Creator

b) 方法

声明	描述
<code>public static void RegisterACreatorFactory (WorldCreatorStudio_Core.MapCreator.MapCreatorFactory creatorFactory)</code>	注册一个 CreatorFactory creatorFactory : 要 注 册 的 CreatorFactory
<code>[DEL]public static WorldCreatorStudio_Core.MapCreator.MapCreatorFactory GetACreatorFactory (string programSet)</code>	【已删除】根据 programSet 获取一个 CreatorFactory programSet : 所 根据的 programSet 字符串

3.[static]Resouses/Icons

提供所有用到的的图标资源。

a) 属性

名称	描述
<code>public static BitmapSource DarkIconPro</code>	暗色主题 工程图标
<code>public static BitmapSource DarkIconWork</code>	暗色主题 工作图标

<code>public static BitmapSource DarkIconRes</code>	暗色主题 资源图标
<code>public static BitmapSource DarkIconResLib</code>	暗色主题 资源库图标
<code>public static BitmapSource DarkIconBackEndWork</code>	暗色主题 后端工作图标

4.[WPF]Resouses/ControlTemplates [WPF]Resouses/Theme

MiRaUIProject 模块提供的内容

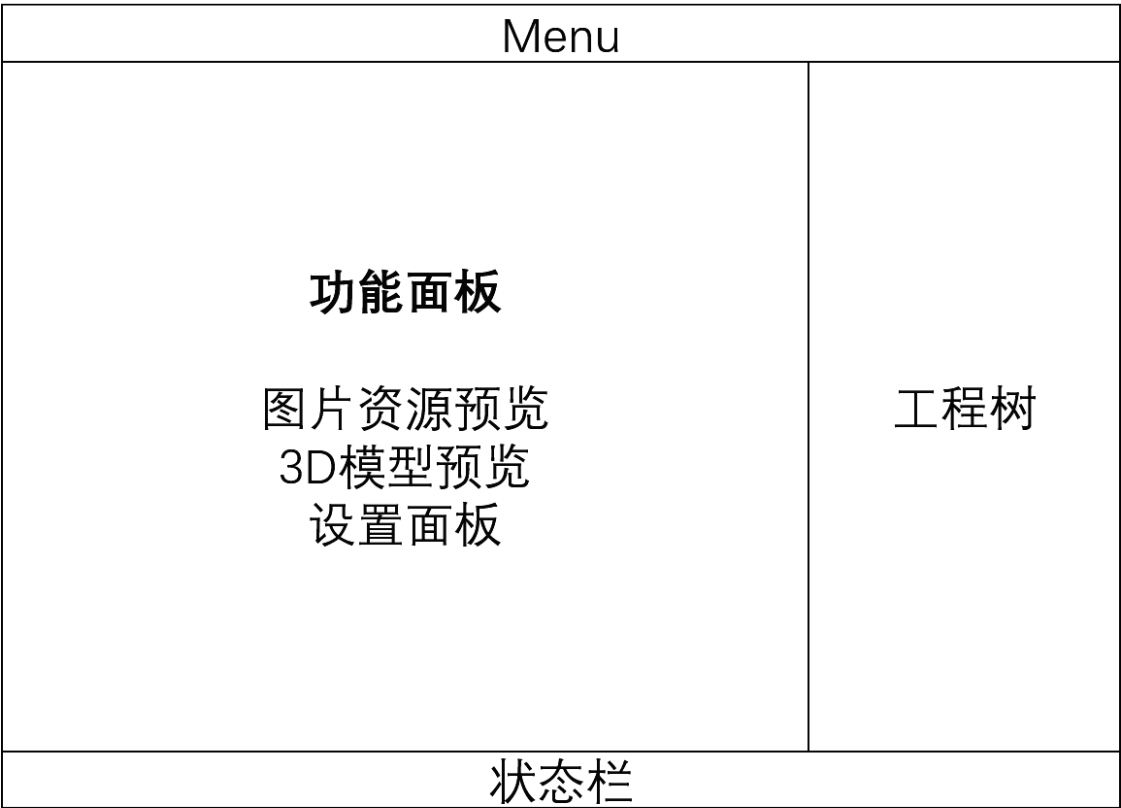
5.[WPF]Windows/MainWindow

主功能窗体

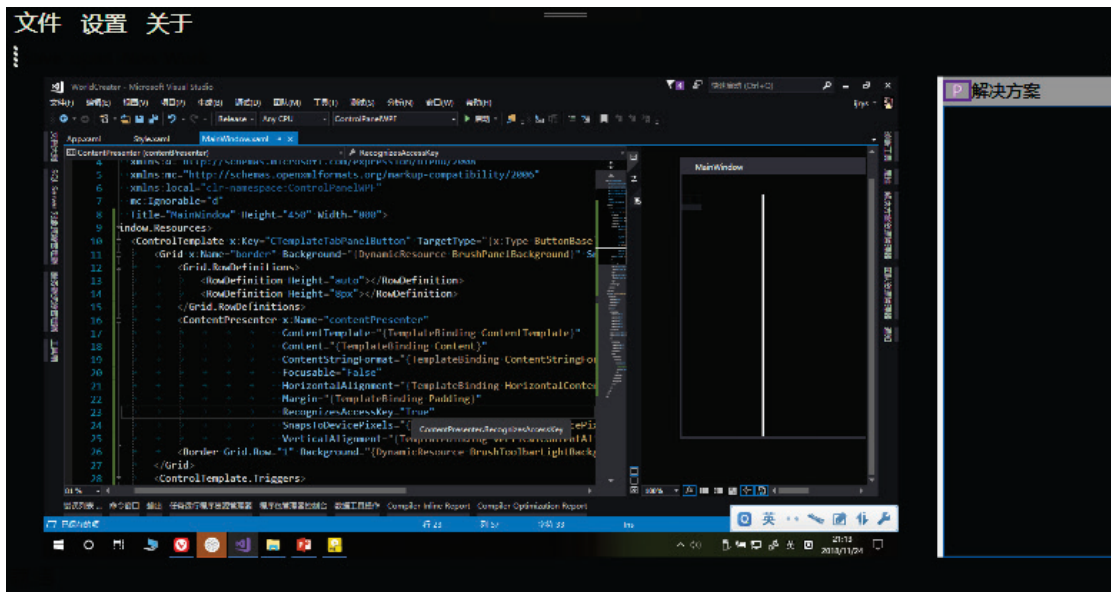
- 继承：Window

a) UI 设计

- 原型设计图



● 最终产品图



b) 委托和事件

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

c) 字段和属性

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

d) 方法

名称	描述
<code>private void CommandNewWorkExecuted (object</code>	命令 NewWork 执行的方法

<code>sender, ExecutedRoutedEventArgs e)</code>	sender: 调用者 e: 参数
<code>private void CommandNewProjectExecuted (object sender, ExecutedRoutedEventArgs e)</code>	命令 NewProject 执行的方法 sender: 调用者 e: 参数
<code>private void ProjectChangedFunction(Project value)</code>	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
<code>private void ShowFunctionPanel (object dataprovider, FrameworkElement panel)</code>	[功能面板相关]展示一个功能面板
<code>private void Tree_Project_Item_DoubleClick (object sender, MouseButtonEventArgs e)</code>	[功能面板相关]Tree_Project_Item 的双击时间
<code>private void RegShowPanel()</code>	[功能面板相关]向 Core 注册展示板

6.[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

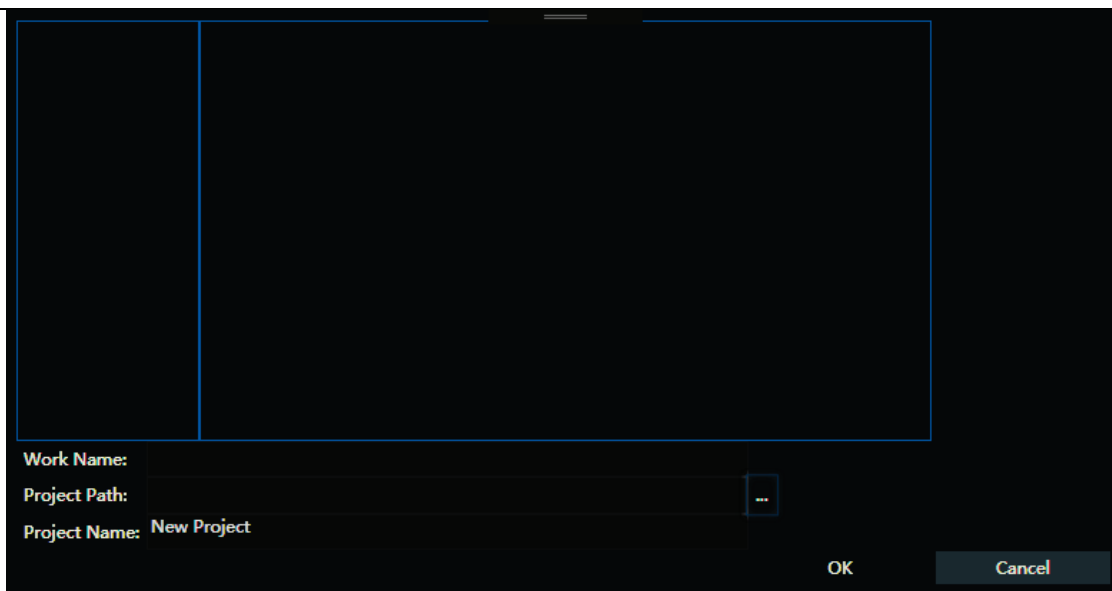
- 继承：Window

a) UI 设计

- 原型设计图



- 最终产品图



b) 字段和属性

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

c) 方法

名称	描述
<code>public new void Show()</code>	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()
<code>public new void ShowDialog()</code>	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()
<code>private void ShowDialog (Project project)</code>	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
<code>public Project GetNewProject()</code>	获取一个新的工程，会根据需要自动弹出窗体
<code>public Work GetNewWork (Project project)</code>	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
<code>private void BtnDirSelect_Click (object sender, RoutedEventArgs e)</code>	前端按钮 BtnDirSelect 点击事件
<code>private void BtnOk_Click (object sender, RoutedEventArgs e)</code>	前端按钮 BtnOk 点击事件
<code>private void BtnCancel_Click (object sender, RoutedEventArgs e)</code>	前端按钮 BtnCancel 点击事件
<code>private void ListCreatorTypeSelectionChanged (object sender, SelectionChangedEventArgs e)</code>	前端列表 List_CreatorType 选择项目改变事件

十三、WorldCreatorStudio_Core 模块功能细则

1.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 WorldCreatorStudio_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。
- MapCreator
FrontendFactory 的 Creator 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。

2.[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

a) 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

b) 事件

- **NodeValueChanged**

event NodeValueChangedEventType NodeValueChanged
--

节点值发生改变事件，用于通知需要保存

c) 方法

● XmlNode

XmlElement

XmlNode(XmlDocument xmlDoc, 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

3.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

a) 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
ProjectDirectory	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

b) 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

c) 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

d) 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

4.ListNode/**Work**

表示一个工作

继承：IWorkLogicNodeAble, InotifyPropertyChanged

a) 属性和字段

名称	类型	可赋值	可获取	默认值	描述
[private]_workDictionary	DirectoryInfo	N/A	N/A		记录工作目录
[private]_workResousesDirectory	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 GUI D
[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
WorkDirectory	DirectoryInfo	F	T	_workDirectory	获取工作目录信息

					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

b) 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

c) 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,  
MapCreator.MapCreatorFactory creatorFactory)
```

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
creatorFactory	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

d) 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
------	------

5.ListNode/**FrontEndFactory**

前端工厂管理节点。

继承：IWorkLogicNodeAble

a) 委托和事件

名称	描述
----	----

b) 属性和字段

名称	类型	可赋值	可获取	默认值	描述
Configuration	MapCreator.Configuration	F	T		获取用于生成地形的配置对象
Creator	MapCreator.MapCreator	F	T		获取用于地形生成的生成

					器
ImageReferenceMa nager	ImageResourceReference Manager	F	T		获 取 前 端 工 厂 的 图 片 资 源 引 用 管 理 器
ResaultHeightMap	FrontendNode.CreatingR esault	F	T		获 取 生 产 的 结 果
[IWorkLogicNodeA ble]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]ImageSource	ImageSource	F	T	WorldCreatorStudio_Resouses.Image	获

ble]Icon				s.Dark_Icon_FrontEndWork	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
--	--	--	--	--	------

c) 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

d) 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

e) 方法

● XmlNode

```
XmlElement XmlNode(XmlDocument xmlDoc, 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

● SetCreator

```
public void SetCreator (string programSet)
```

设置地形生成器

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

● SetCreator

```
public void SetCreator (MapCreator.MapCreatorFactory creatorFactory)
```

设置地形生成器

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

● **CreateAMap**

<code>public FrontendNode.CreatingResault CreateAMap ()</code>
--

执行地形创建过程

● **Children_NodeValueChanged**

<code>private void Children_NodeValueChanged (IWorkLogicNodeAble node)</code>

子节点值发生改变时事件方法

node	来源节点
------	------

● **Configuration_ValueChanged**

<code>private void Configuration_ValueChanged ()</code>

配置节点值发生改变时事件方法