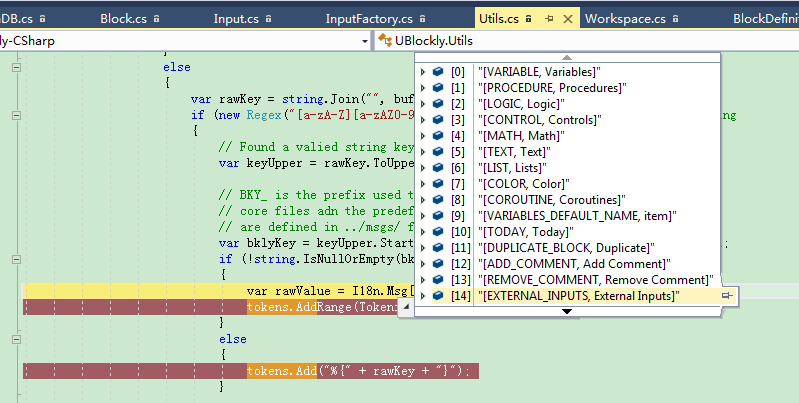
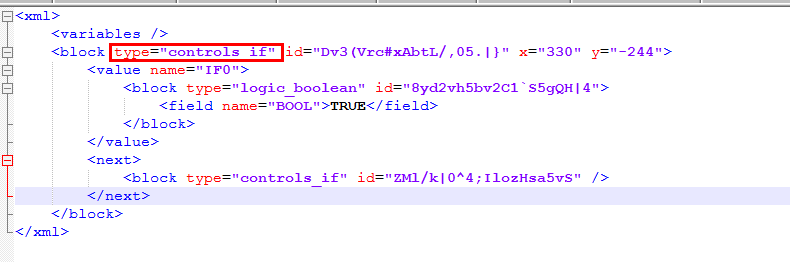
# 杂

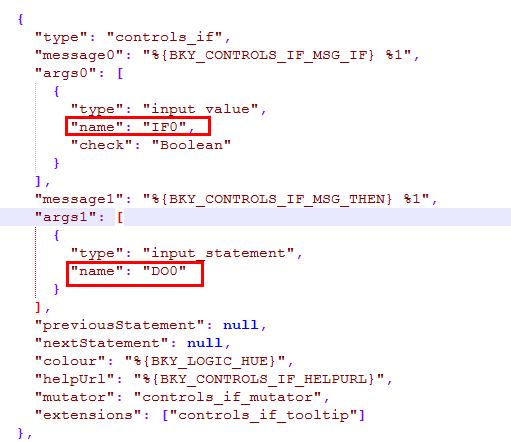


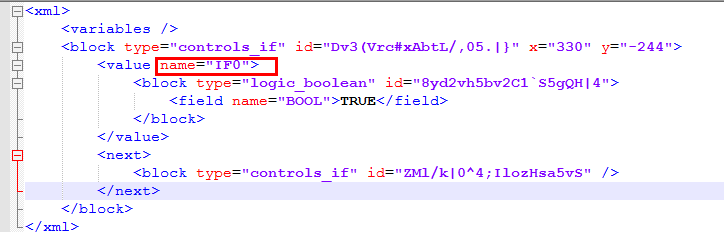
## 创建

根据type创建block:



Input的名字：





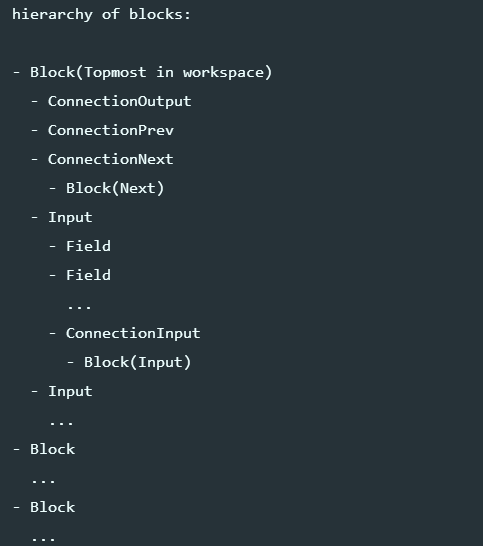
# Blocky Model

组合（搭积木）

## Workspace

Workspace相当于一个容器，包含Blocks, Variables,以及连接关系，可以保存为一份XML,点击组件的**”逻辑编程”**打开一个workspace

## Block结构



### Connection

Connection是实现Blocks之间的连接的关键，方式如下



### Input

## 字段定义

### type

定义积木的类型名称：字符串

每个积木就相当于一个函数function,这个就相当于函数名称

命名:是 **类名\_函数名**

**数据结构说明：**

private Dictionary<string, BlockDefinition> mDefinitions = new Dictionary<string, BlockDefinition>();

**type是**BlockFactory**.**Instance**.mDefinitions的key**

private Dictionary<string, List<string>> mPrefixCategories = new Dictionary<string, List<string>>();

**mPrefixCategories按类名保存所有函数**

列表：.



===============================================

colour\_picker

colour\_blend

===============================================

coroutine\_wait\_time

coroutine\_wait\_frame

===============================================

lists\_create\_empty

lists\_create\_with

lists\_repeat

lists\_reverse

lists\_isEmpty

lists\_length

lists\_indexOf

lists\_getIndex

lists\_removeIndex

lists\_setIndex

lists\_getSublist

lists\_sort

lists\_split\_from\_text

lists\_join\_text

===============================================

logic\_boolean

controls\_if

controls\_ifelse

logic\_compare

logic\_operation

logic\_negate

logic\_ternary

===============================================

controls\_repeat

controls\_whileUntil

controls\_for

controls\_forEach

controls\_flow\_statements

===============================================

math\_number

math\_arithmetic

math\_single

math\_trig

math\_constant

math\_number\_property

math\_round

math\_on\_list

math\_modulo

math\_constrain

math\_random\_int

math\_random\_float

===============================================

procedures\_defnoreturn

procedures\_defreturn

procedures\_callnoreturn

procedures\_callreturn

=======================Text========================

Text

text\_join

text\_append

text\_length

text\_isEmpty

text\_indexOf

text\_charAt

text\_getSubstring

text\_changeCase

text\_trim

text\_count

text\_replace

text\_reverse

=======================variables========================

variables\_get

variables\_set

variables\_change

### output

mHasOutput是否有输出

### previousStatement

mHasPreviousStatement 是否有前置代码段(函数)

### nextStatement

mHasNextStatement是否有后置代码段（函数）

### mOutputChecks

输出类型检查用， 输出类型转换为字符串保存在这

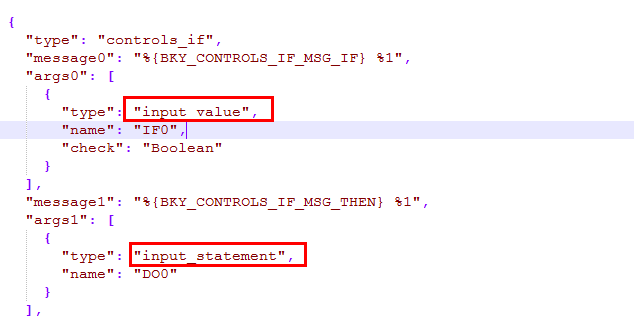
### inputsInline

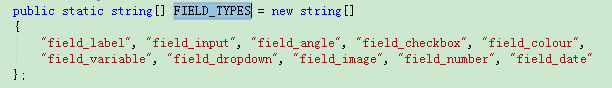
mInputsInlineDefault输入是否在同一行

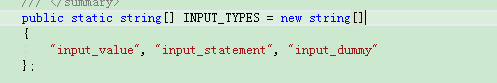
### mutator

### extensions

### 参数type







# 生成器、解析器

## 生成器

## 解析器

# UI

# 需求

## 运动

向左移动

向右移动  
向上移动

向下移动



## 外观

显示

隐藏

换造型





## 声音

播放声音



## 事件

当接收到消息

广播消息



## 控制

条件

循环

等待

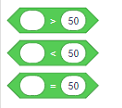








## 运算



## 变量

1. 游戏里导出的变量（如游戏时间倒计时、杀敌数）
2. 组件里的临时变量

## 生成器

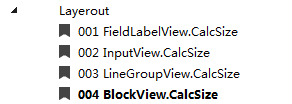
# 实现

## IEnumerator

要StartCoroutine才真正执行



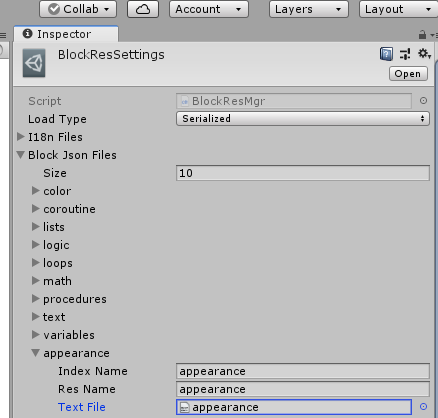
## Layerout

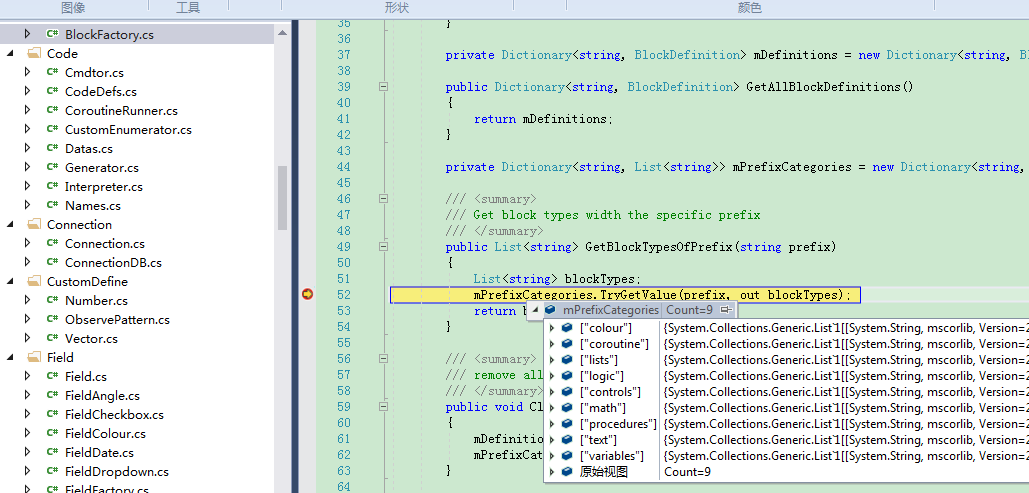


FieldLabelView.CalcSize🡺

InputView.CalcSize🡺

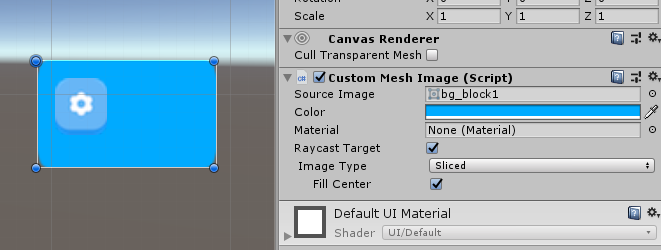
## Tab栏



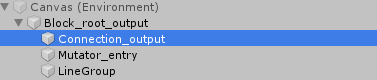


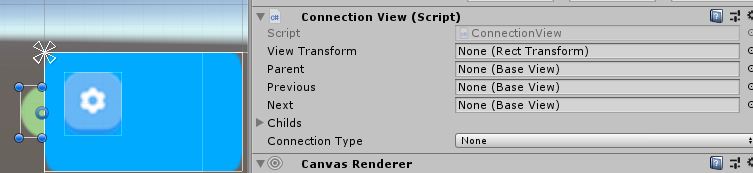
## 外观

### Block\_root

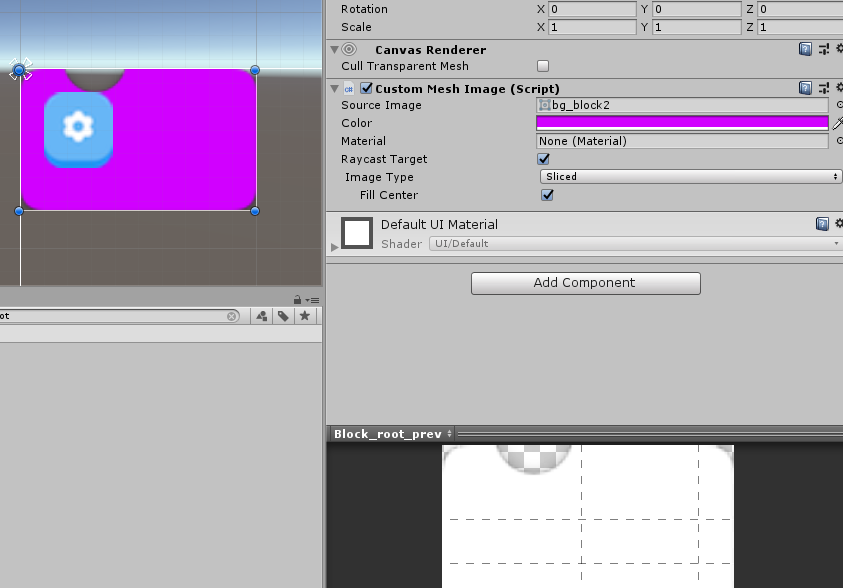


### Block\_root\_output

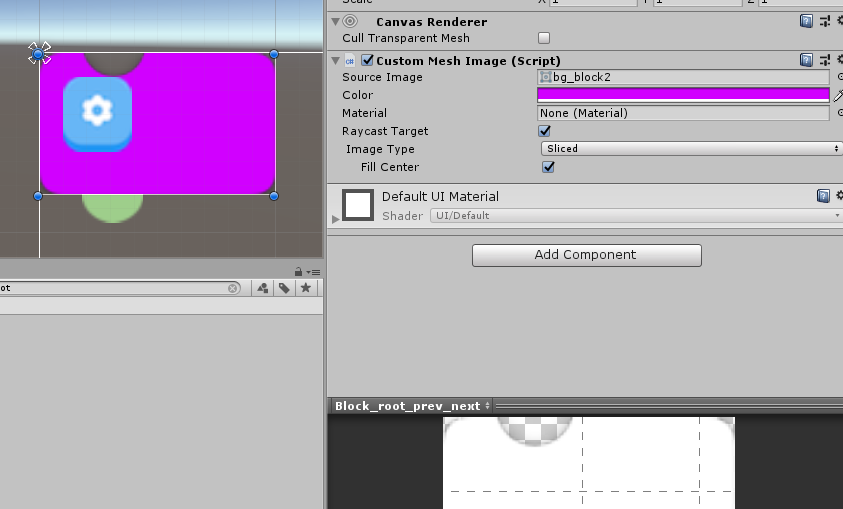




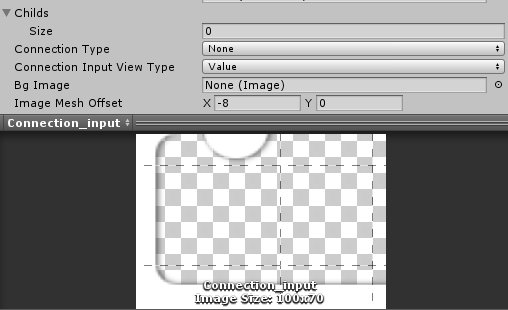
### Block\_root\_prev



### Block\_root\_prev\_next



### Input\_statement



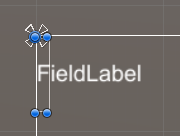
### Input\_value\_slot



### Input\_value

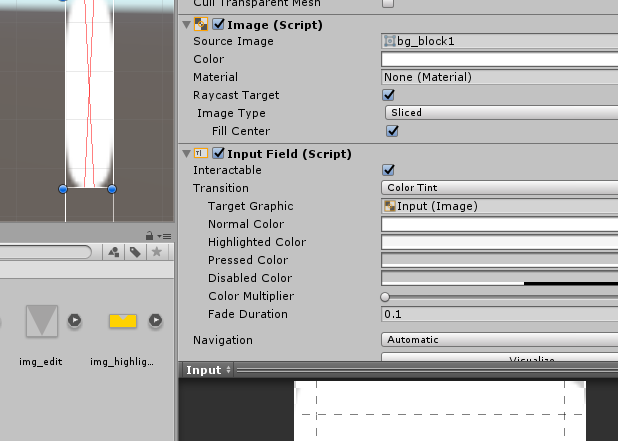


### Filed\_label

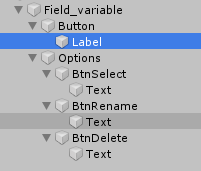
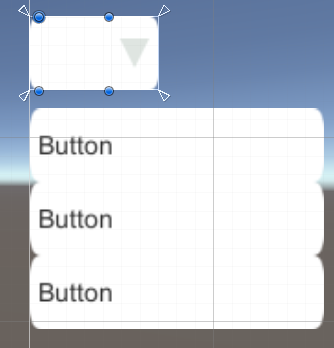


### Filed\_input

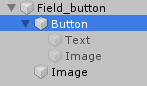
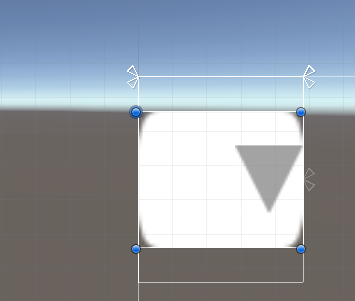




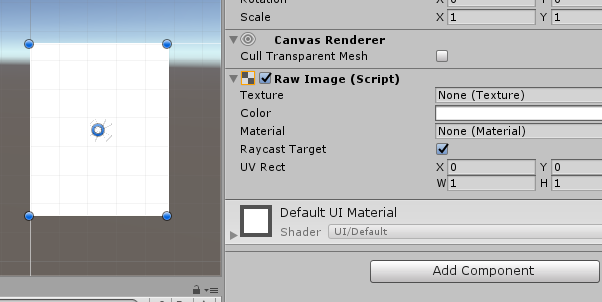
### Filed\_variable

### Filed\_button

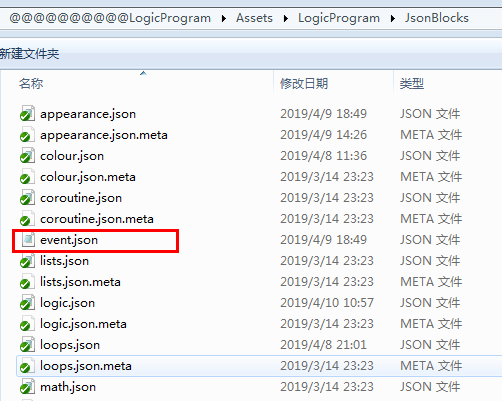
### Filed\_image

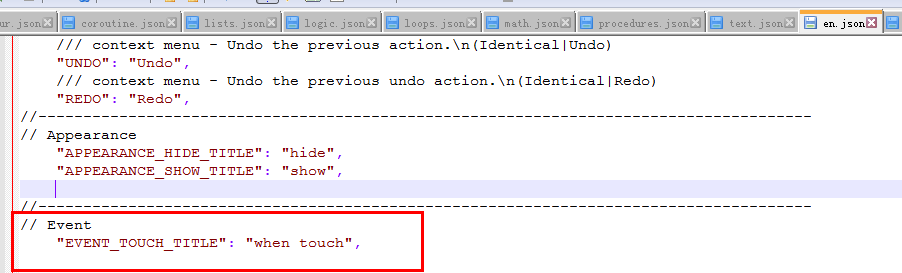
# 说明

## 增加tab栏(比如”事件”)

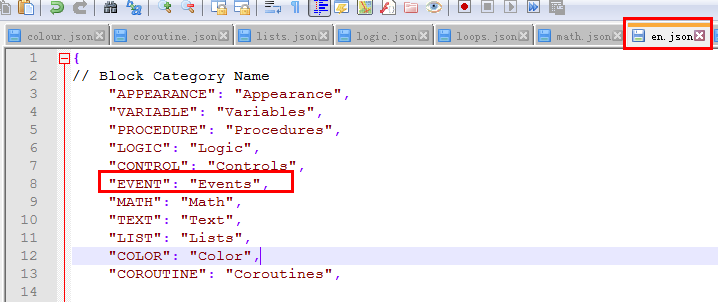
### 增加json定义文件



### 积木的描述从描述文件获取（目的支持多语言）

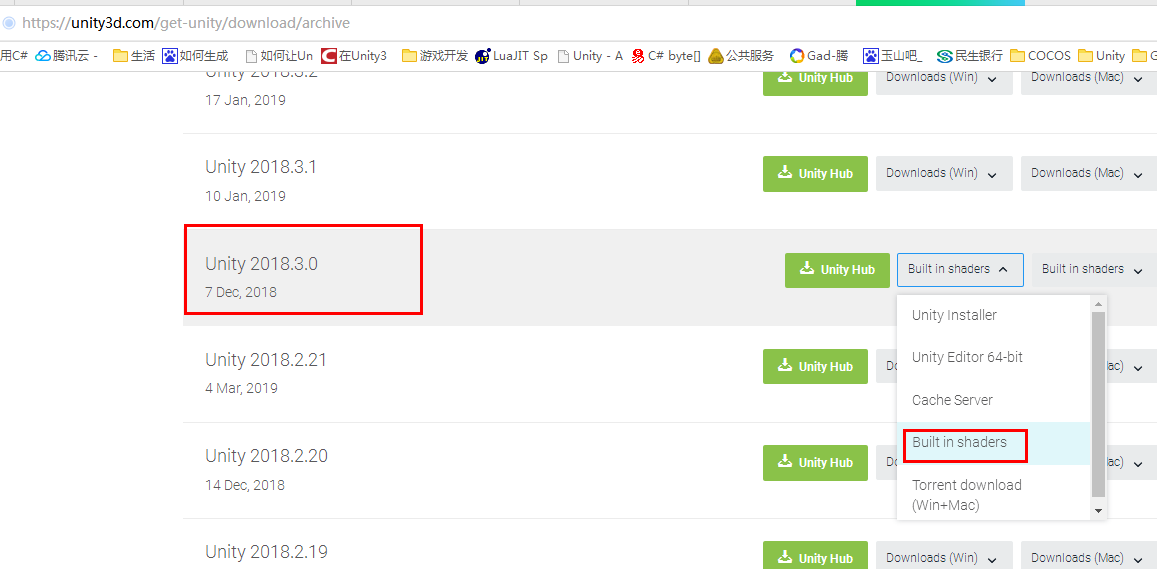


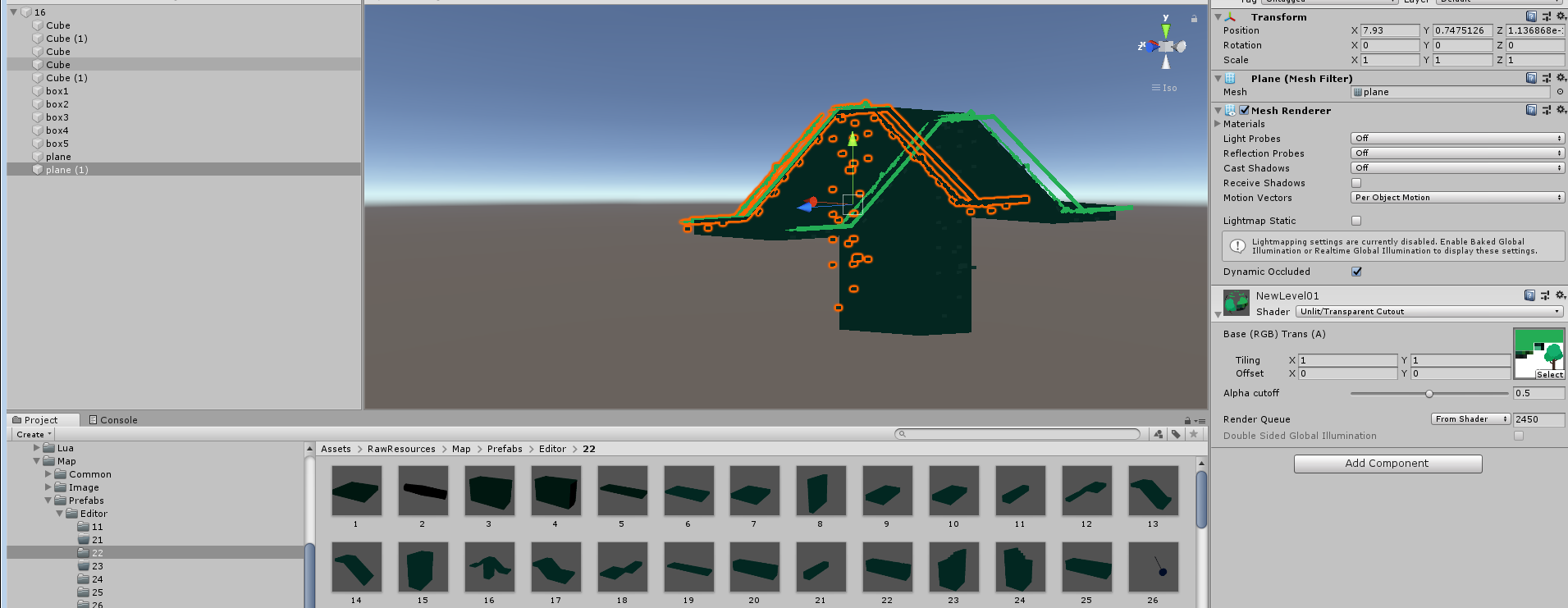
### Tab栏的名字



## Shade

Shade官网可以下载：





### Cull Front //关闭背面剔除

