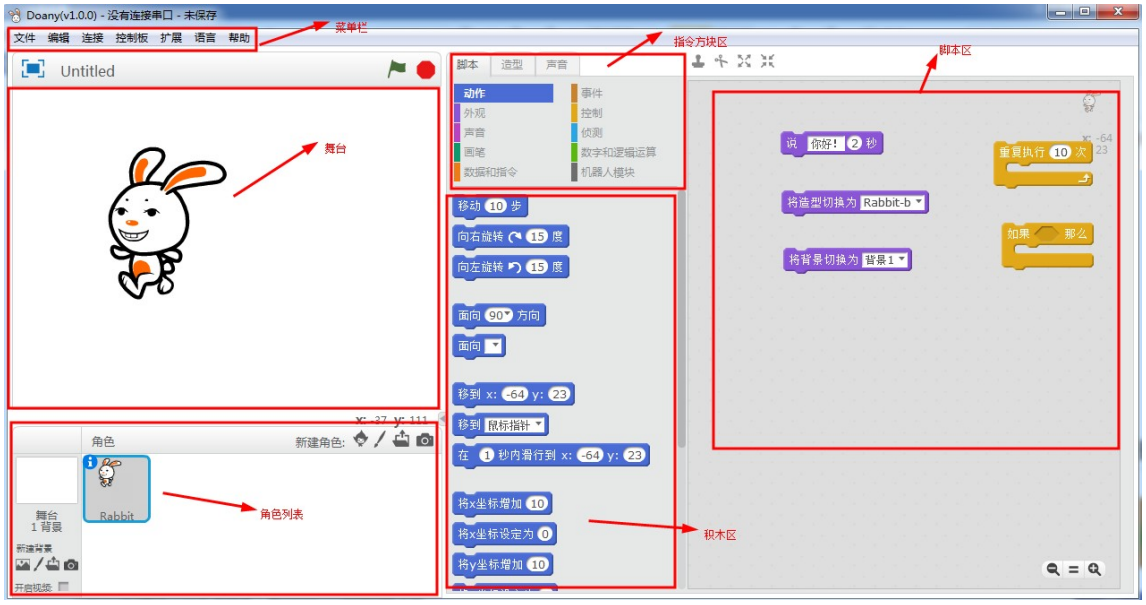
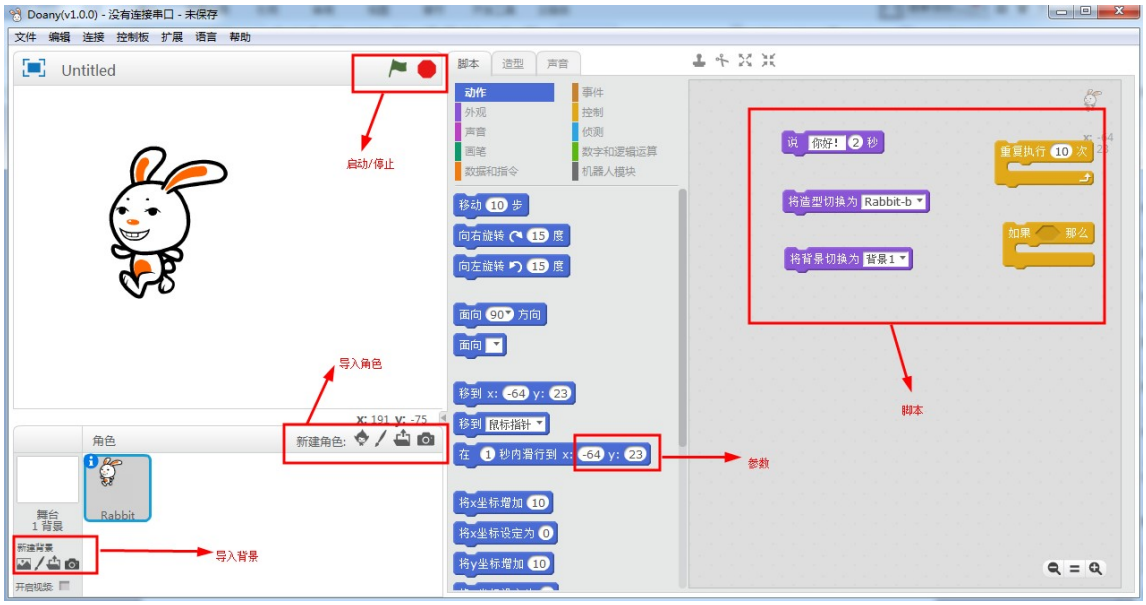
界面

Doany 介绍

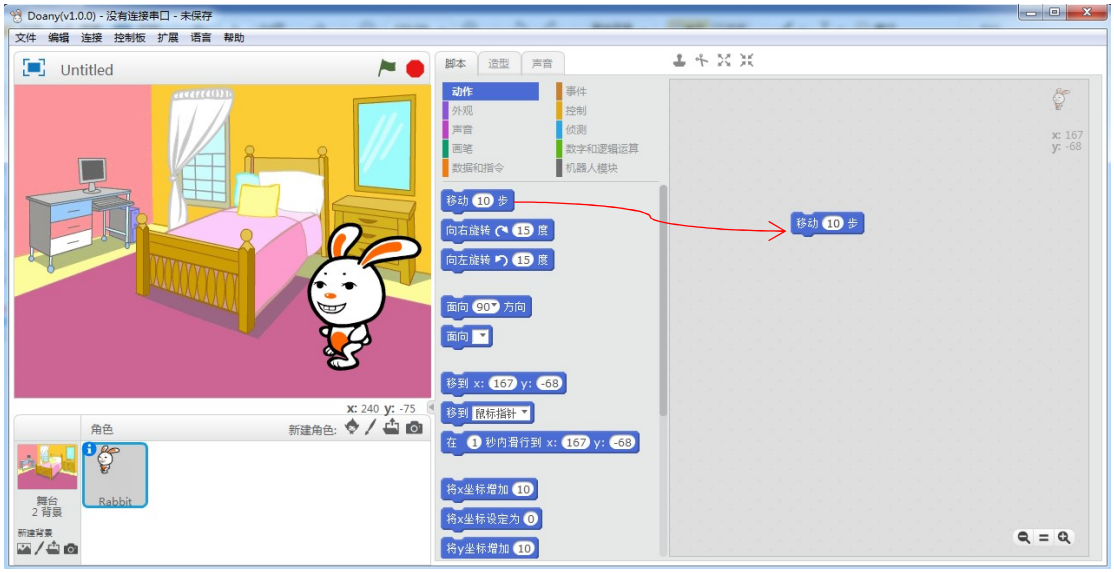
Doany 是一款基于 Scratch 2.0 开发的图形化编程软件，它拥有简单的程序编写方式，可以兼容 Arduino 的主板并加以控制。





图形化编程方式

Doany 的编程方式非常简单，只需将积木拖拽到脚本区即可：



然后通过积木之间相互卡合的方式来实现程序块之间的连接：

积木的形状

触发式积木：任何脚本的第一块积木由事件触发执行
命令积木：执行相关的命令
参数模块：不能单独使用，需要将其放入其他积木中
C 形积木：内部可以卡和其他模块，实现其特有功能

参数的形状与数据类型

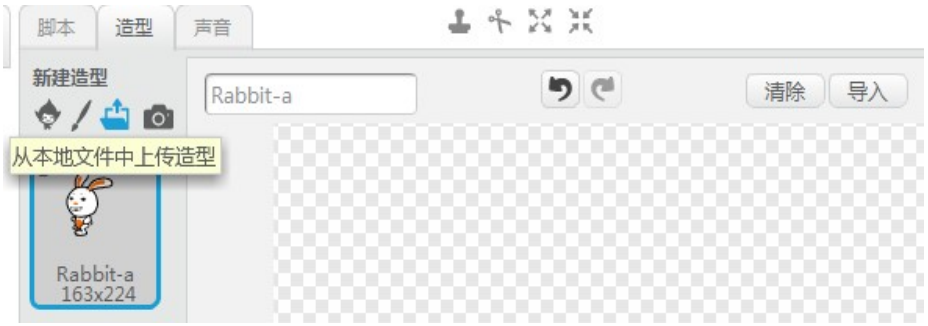
第一块积木中的圆角矩形里的参数在设置时只能放入数字，不能放入其他参数。
第二块积木中的六边形框内只能放形状为六边形的积木模块。六边形积木返回的布尔值，就是真和假。
第三块积木中的矩形框内的参数既可以放数字也可以放字符串（即数字，字母，字符）。

Doany 的程序结构

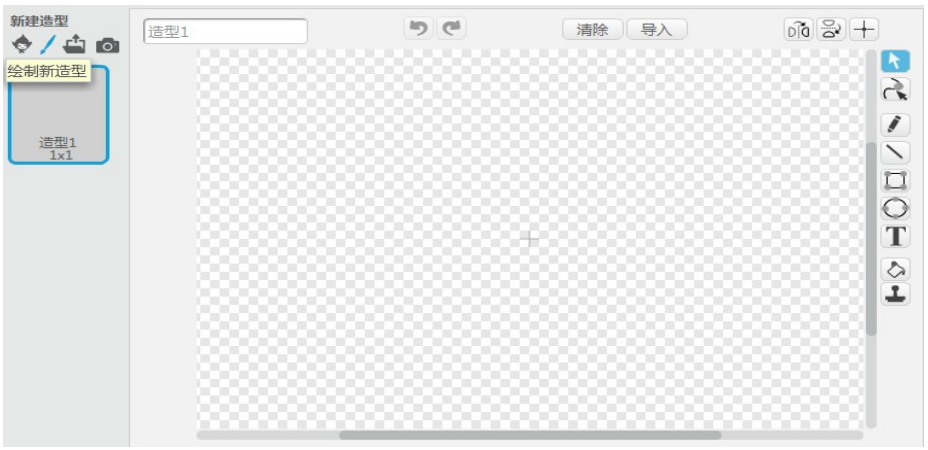
Doany 程序要么是纯软件的形式让很多角色在小舞台里面互动，要么是通过脚本以硬件的形式实现其功能，或者两种都有。一个 Doany 程序可以拥有众多的角色，每一个都拥有其自己独特的脚本、声音；每个程序还可以设置一个背景，背景也可拥有声音。脚本是积木的结合，是实现每一个程序的核心，因为它实现了硬件和软件的逻辑。

角色与造型的创建

Doany 的角色与造型的创建既可用导入图片的形式，也可根据自己的想象绘制。GIF 文件包含许多帧，我们既可以直接将其作为角色也可以将这些帧导入到造型中：



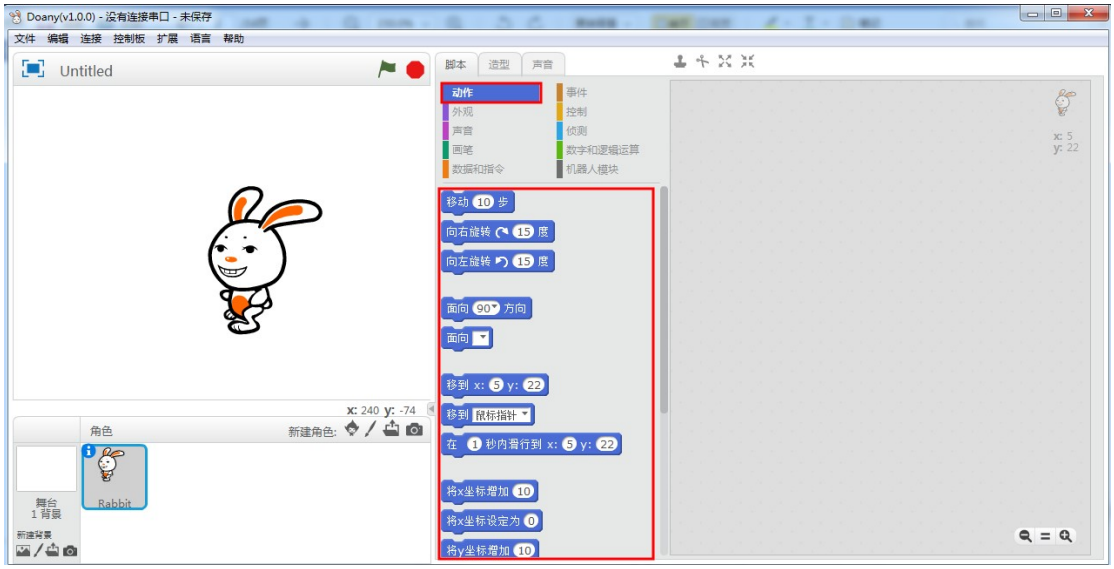
还可以根据自己的想法绘制造型：

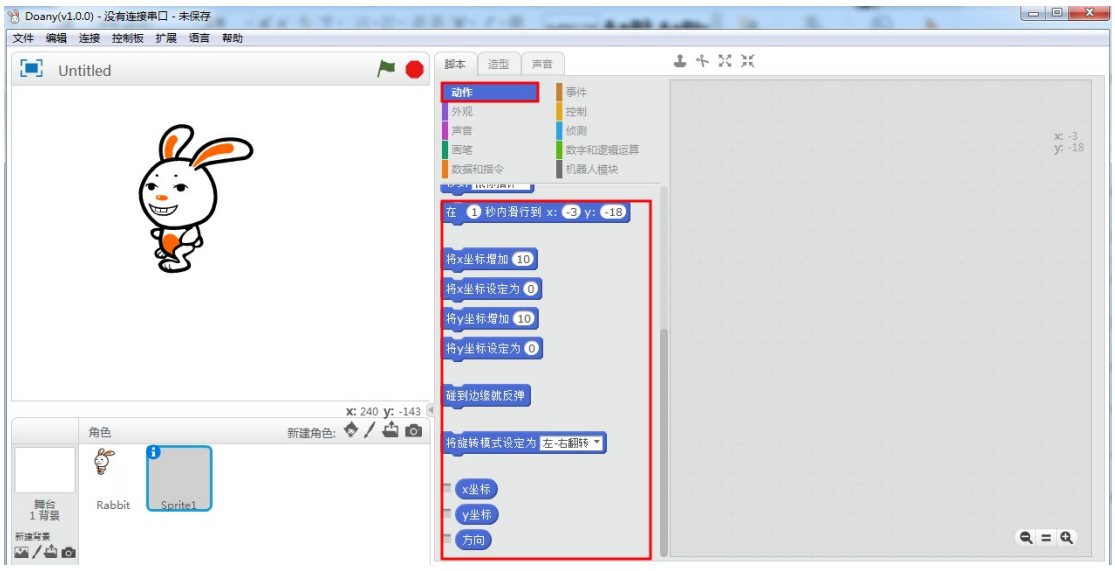


动作

动作指令区

下面红框区域就是 Doany 的动作指令：





通过上面这些指令，Doany 会做出与之相对应的动作，比如移动十步，移动到 XY(-3，-18)所在位置。

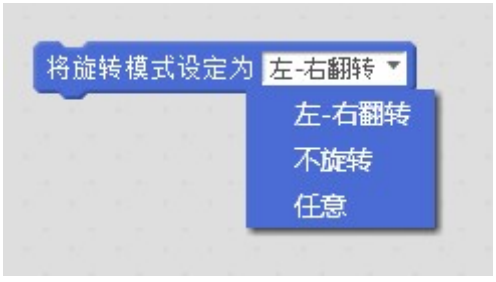
角色的方向

选择不同的方向，舞台中的角色就会根据你的选择实现面朝哪边，向哪边旋转：

旋转模式

角色的旋转模式有三种，可以在两个地方进行设置，第一处是在角色列表中，用鼠标点击角色左上角的感叹号，就可以选择不同的旋转模式：

另外一处就是使用积木块了

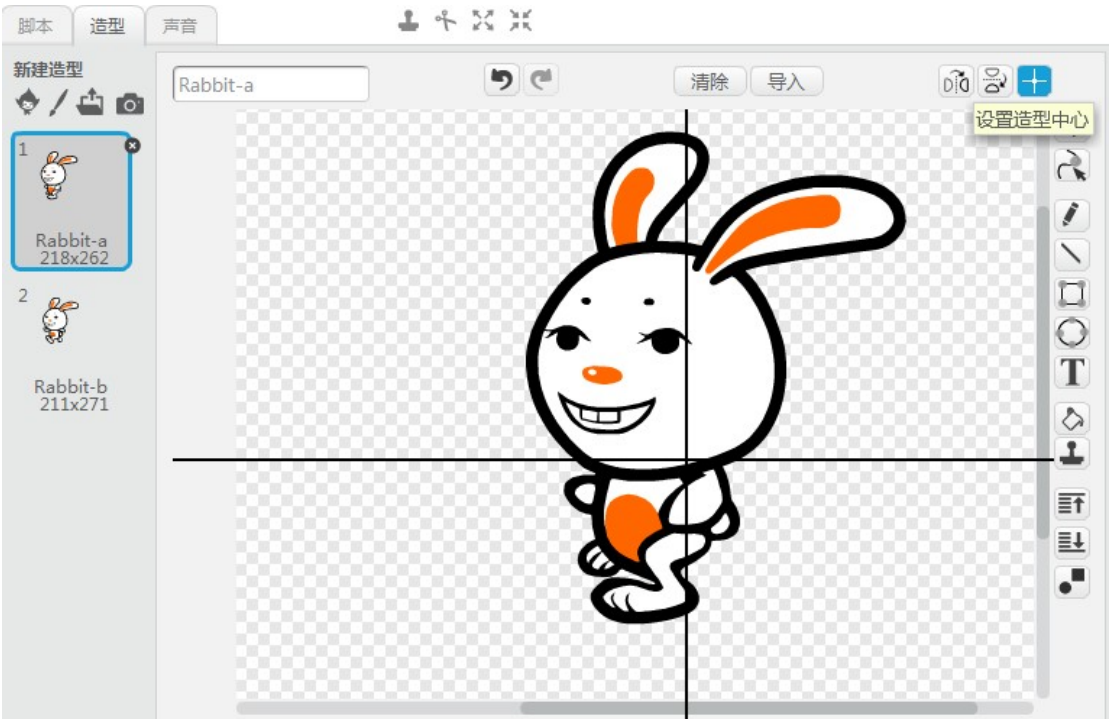


它的旋转模式含义：

- 左-右翻转就是角色只能左右翻转，即 90 和-90 度。
- 不旋转就是无论给角色设定什么值，角色的方向不做任何变化，始终面朝 90 度方向。
- 任意即可以做任何方向的旋转。

造型的中心点

一个角色的每一个造型都有一个中心点，在造型页面中可以进行打开并设置：



中心点的选择有三个作用：

- 可以作为角色坐标的依据
- 作为画笔的中心点
- 作为旋转中心点

即坐标的依据就是积木 

的坐标点，画笔中心就是指 

模块中 

的积木在画线时的位置，而旋转中心就是指 

积木模块旋转时的中心点。

角色的移动

角色有许多种的移动方式，常见的有：水平位置随机运动，垂直位置随机运动，移动到舞台任意位置：

左右摇摆

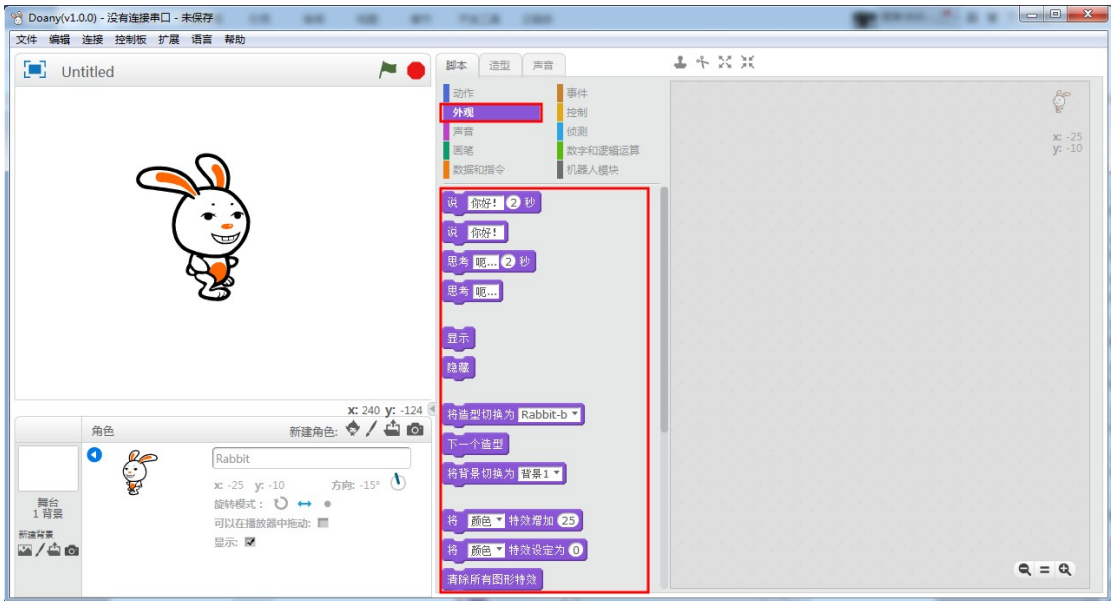
按照三角形轨迹移动

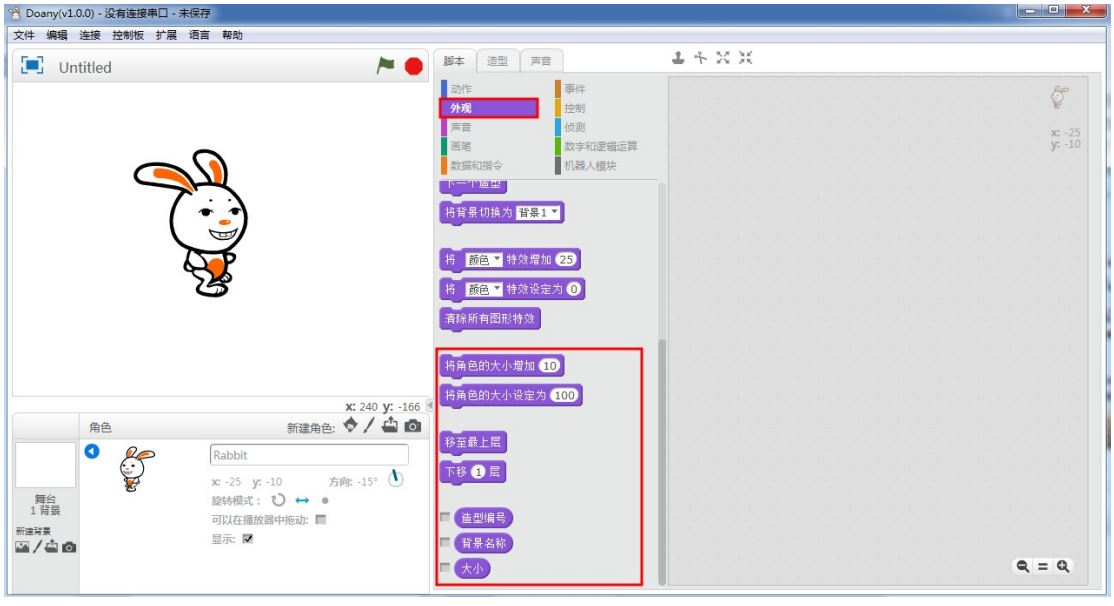
通过以上按键的方法控制角色时无法实现组合按键（即上移和右移一起按下）且会有一点延时，如有需要，可以参考下面的方式进行移动。

外观

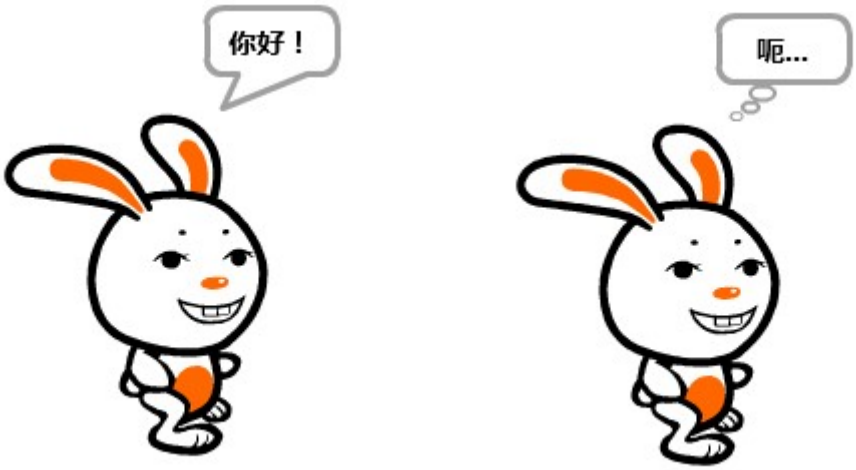
外观指令区

下图中红色区域就是要介绍的外观指令：



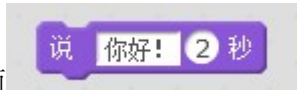


说和思考



通过这两种积木，用于角色之间的交流。

这四块积木都用于角色交互，不同在于 

这个积木会一直说你好，要将内容清除就在矩形框内将其内容删除执行就好了，而 

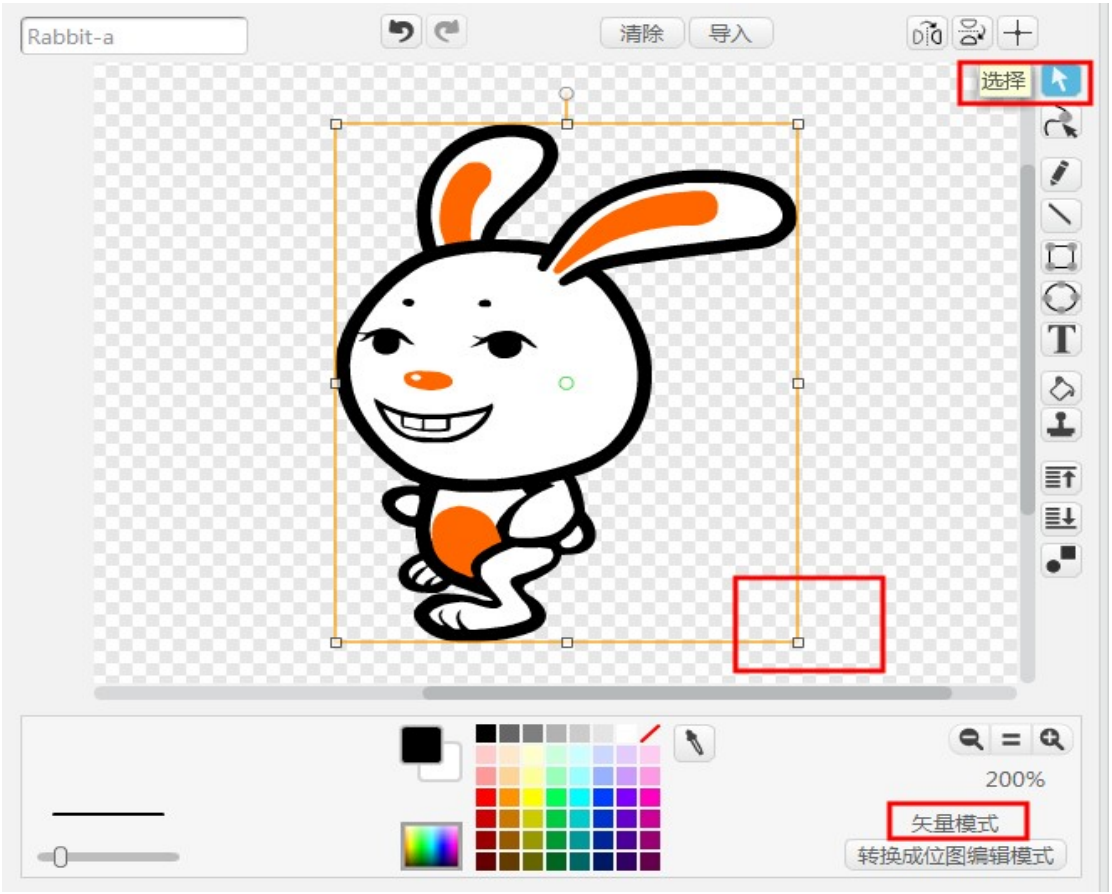
积木将会在规定时间后自动清除效果。

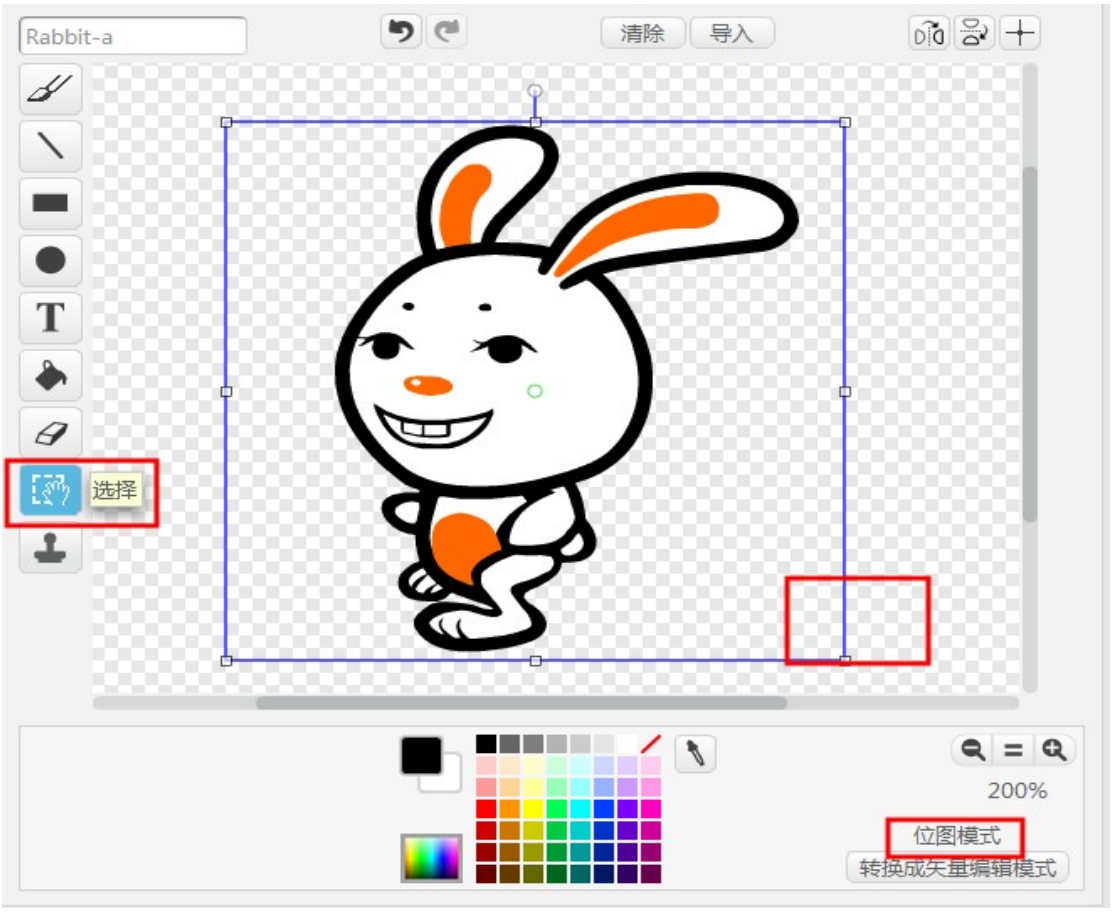
角色的放大和缩小

改变角色的方法有以下三种：

- 造型里面的绘图编辑器
- 工具栏上的  图标，使用会改变角色中所有造型的大小
- 使用积木进行放大和缩小

第一种方法：首先看造型处于何种方式，造型处于矢量模式，就选用选择工具，在点击矢量图即可；若造型处于位图模式，也选择使用工具，选中希望改变大小的区域即可。


Rabbit-a
清除
导入
选择
200%
矢量模式
转换成位图编辑模式


Rabbit-a
清除
导入
选择
200%
位图模式
转换成矢量编辑模式

第二种方法：用鼠标点击选择工具栏上的放大和缩小选项，再点击对应的角色即可：

第三种方法：就是使用 

模块里的 

和 

就可以在脚本里动态的实现改变造型的大小，设定参数 100 为原始大小，参数 150 就是原来的 1.5 倍。使用 

积木可以在程序中动态的获得角色当前的大小。

造型与动画

每一个角色都至少有一个造型，在任意时刻都要处于一个造型之中。与造型相关的积木如下所示：

第一个积木可以直接切换到自己想要的造型，第二个积木会自动切换到下个造型，如果当前处于最后一个造型的话，那么自动切换到第一个造型，第三个积木则会告诉你当前处于第几个造型。

角色的造型可以在造型里面添加，修改，删除：

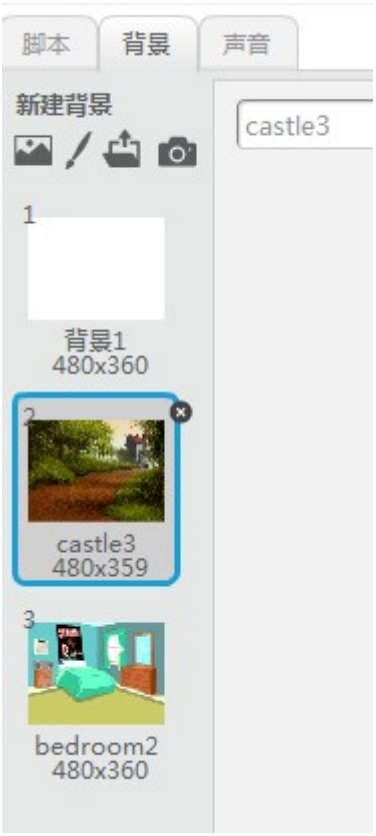
我们可以制作一个简单的动画来展示效果：

背景与场景

任何角色，包括舞台本身都可以切换背景，相关积木如下：

第一个积木可以切换到具体的背景；第二个积木可以获得当前背景的名称。

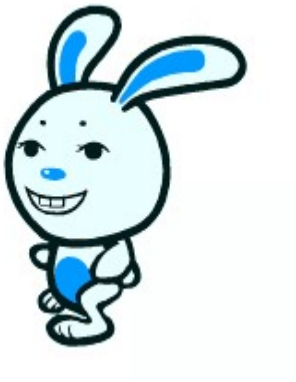
舞台的背景往往是和背景有关的，下面演示一下简单的场景切换，首先要从素材库中添加两个背景，也可以自己绘制，然后再编写脚本。



图形特效

图形特效是为了让角色更加的生动，我们可以更改外观上的效果，一些相关的积木如下：

将颜色特效设定值改为 0 就是还原到最初的效果，也可以使用“清除所有图形特效”积木，就可以还原。各个效果如下图所示：



将 颜色 特效设定为 100

将 超广角镜头 特效设定为 100

将 旋转 特效设定为 100

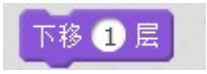
将 像素滤镜 特效设定为 100

将 马赛克 特效设定为 50

将 亮度 特效设定为 50

角色之间的层次关系

当有多个角色在一起时，就会发生重叠，这就涉及到了谁在谁上面的情况：

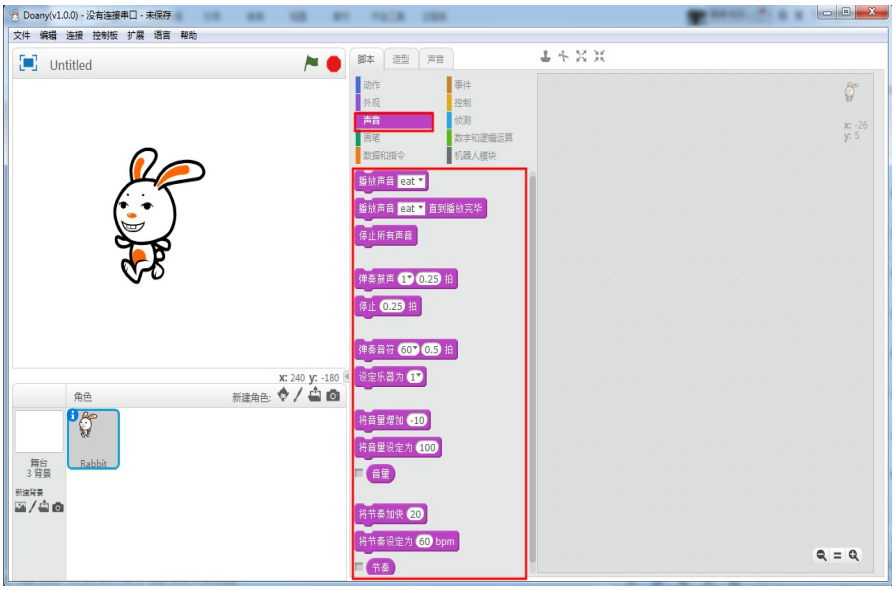


使用上面两块积木可以进行调整角色之间的层次关系。

声音

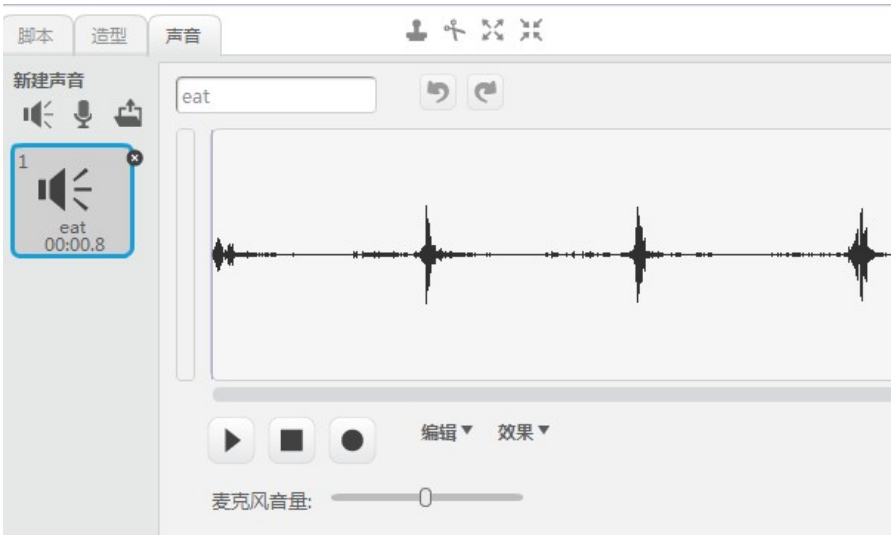
外观指令区

下图中的红框区域就是相对应的声音指令：



播放声音

每一个角色都拥有声音，你可以在声音页面里自己录制声音，导入外部声音或者从库中选取声音都可以：

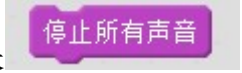


播放声音使用模块 

里的积木 

，前两个积木的差别在于第一个积木播放声音后会进行下个积木的内容，而第二个积木要等到声音播放完毕后才能进入下一个积木。我们可以动手通过下面的例子来说明：

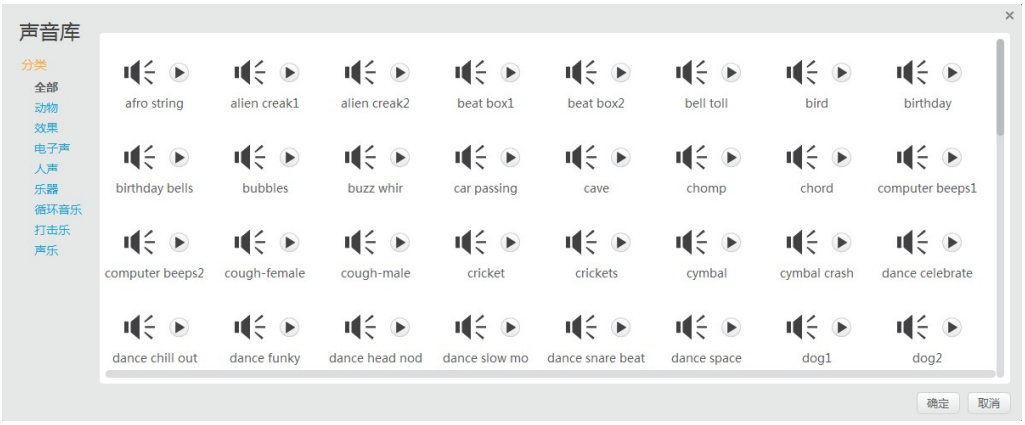
如上面的脚本的效果可以看出第一个脚本的效果是一遍播放声音一边移动，第二个脚本的效果是声音播放完毕后才开始移动。

使用第三块积木 

时，所有的声音都将停止播放。

添加声音库

除了一些自己制作的声音外，还可以从库中调取，来制作适合程序的音乐：



制作乐曲

制作一个乐曲除了需要设置音符和节拍外，还需要考虑音量的大小与速度，使用的音色等：

设置弹奏有音符的音色，设置停止音符的时间

弹奏鼓声和音符

音量 100 为原始音量

bpm 为每分钟的节拍数

通过下面脚本了解 bpm：

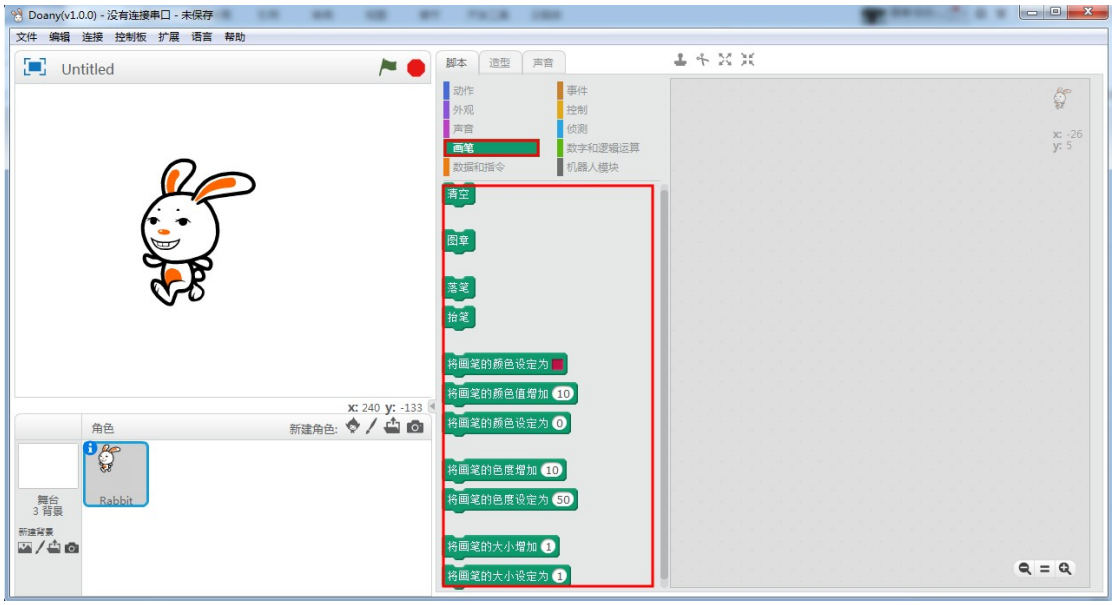
当 bpm 为 120，1 分钟演奏 120 拍，4 拍时间为 2S

下面动手制作粉刷匠的主奏：

画笔

外观指令区

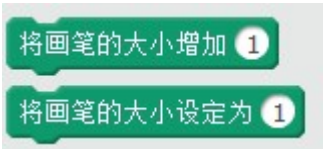
下图中红色区域就是要介绍的画笔指令：



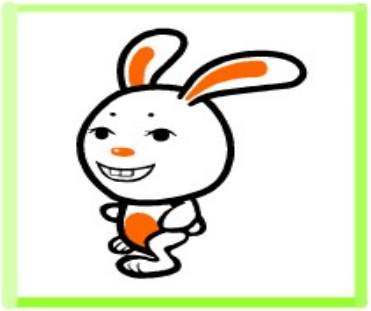
绘图功能

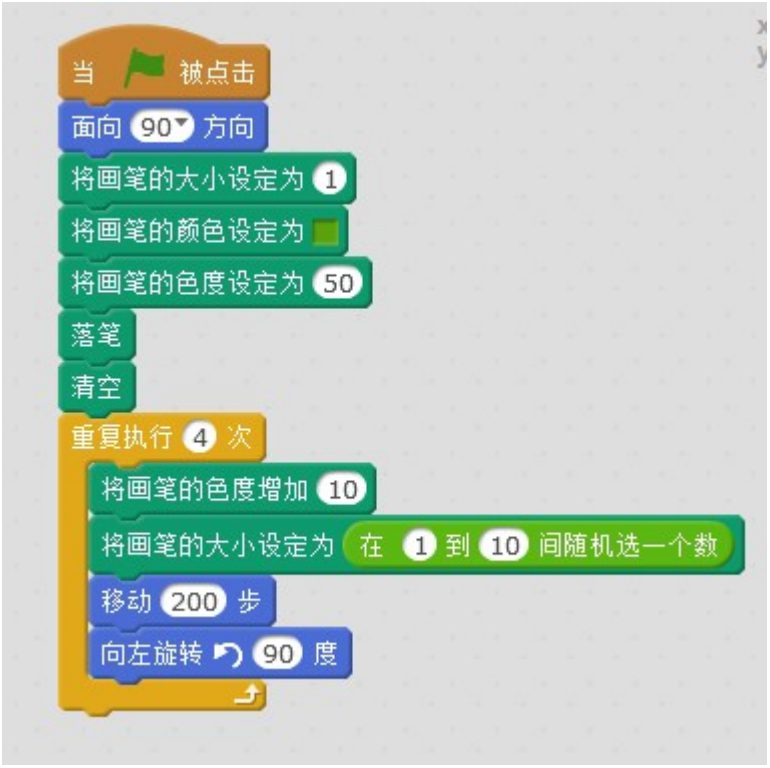
每个角色都自带了一个隐形的画笔，画笔的笔尖就是改造型的中心点，而画笔只有两种状态，即抬起和落下，建立角色时画笔默认处于“抬笔”的状态：

画笔还有三个属性：颜色，色度，大小。它们相对应的积木为：



以上的积木，通过改变这些属性我们可以绘制出非常漂亮的图案：





图章

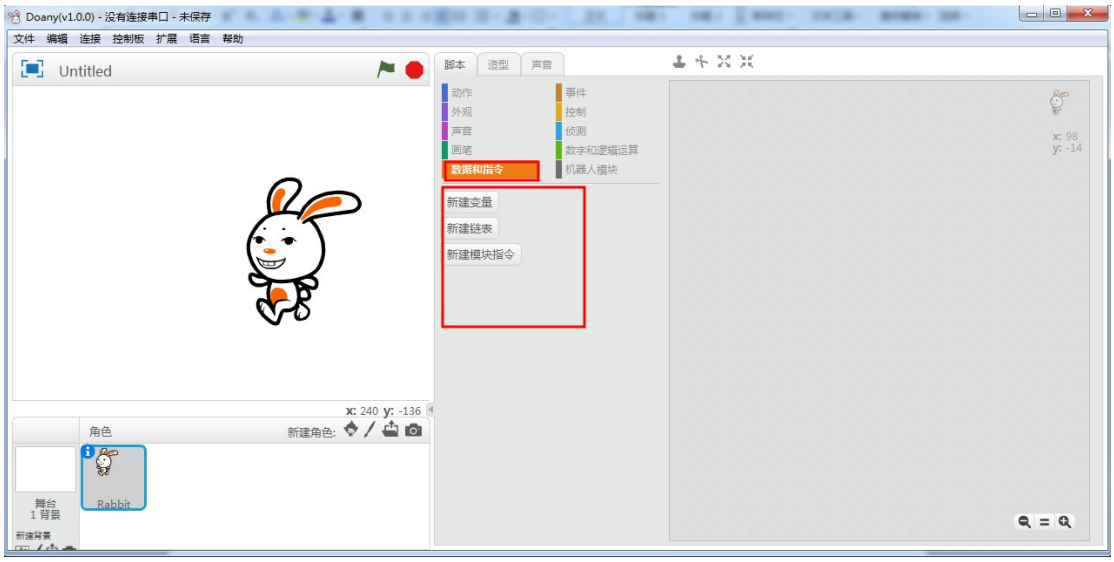
图章和画笔一样，都可以在舞台中留下印记，不同的是，图章留下来的是角色当前造型本身，而不是一条线，并且不需要执行落笔和抬笔的操作，因为图章积木本身就包含了落笔和抬笔的操作，我们可以通过例子来了解图章的这个功能：

通过上图可以看出，图章就是复制了角色，如果使用画笔比较麻烦的话，可以先在编辑器中将角色绘制好，然后在使用图章积木进行绘制。

数据和指令

外观指令区

下图中红色区域就是要介绍的数据和指令：



变量

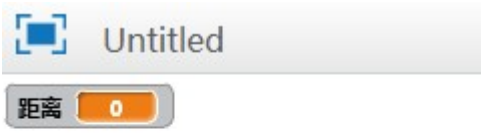
变量是什么？简单的说，我们可以把变量看做是一个盒子，可以将一些物品放在盒子中。

我们可以在 

模块中新建变量：

在弹出的窗口中可以添加需要的变量名,其中“适用于所有角色”和“仅适用于当前角色”的区别在于前者的变量是所有角色所通用的，即全局变量。而后者仅仅可以在当前角色内部中使用，并且也可以作为克隆体的变量使用，即局部变量。还有一个区别则在于重命名的问题，全局变量的命名是唯一的，不能重复，而局部变量是在角色内部唯一的，其他的角色可以拥有相同的名称。

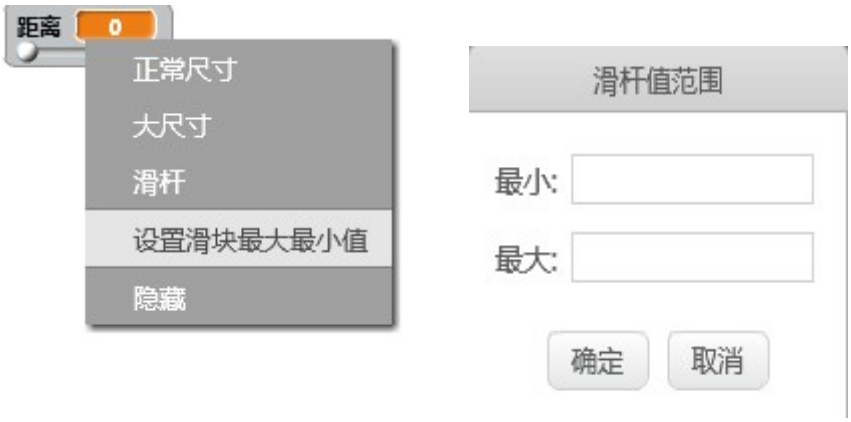
再输入变量名确定后，积木区就会出现新的积木，同时舞台上出现“变量值显示器”：



“将变量设定为”积木可以给变量一个初始的值，“将变量的值增加”积木可以对变量增加或减少一定的数值，最后“显示和隐藏变量”就是将舞台中的变量显示器进行隐藏和显示。

舞台上的变量显示器有三种展示的方式：

默认的就是正常尺寸，有时候会调成大尺寸，也会调成滑杆，这样对程序的交互会有很大的帮助，在滑杆模式下还可以点击右键进行范围的调整，默认范围是{0,100}，还可以显示小数形式：



变量名也可以进行操作，不需要了可以将它删掉，名字也可以进行修改，右键变量名可以进行修改和删除。

一个变量对程序的使用会很重要，使用变量的次数会很频繁，例如记录步数，时间，进行一些运算啊，可以看下面的一些程序对于变量的使用：

链表

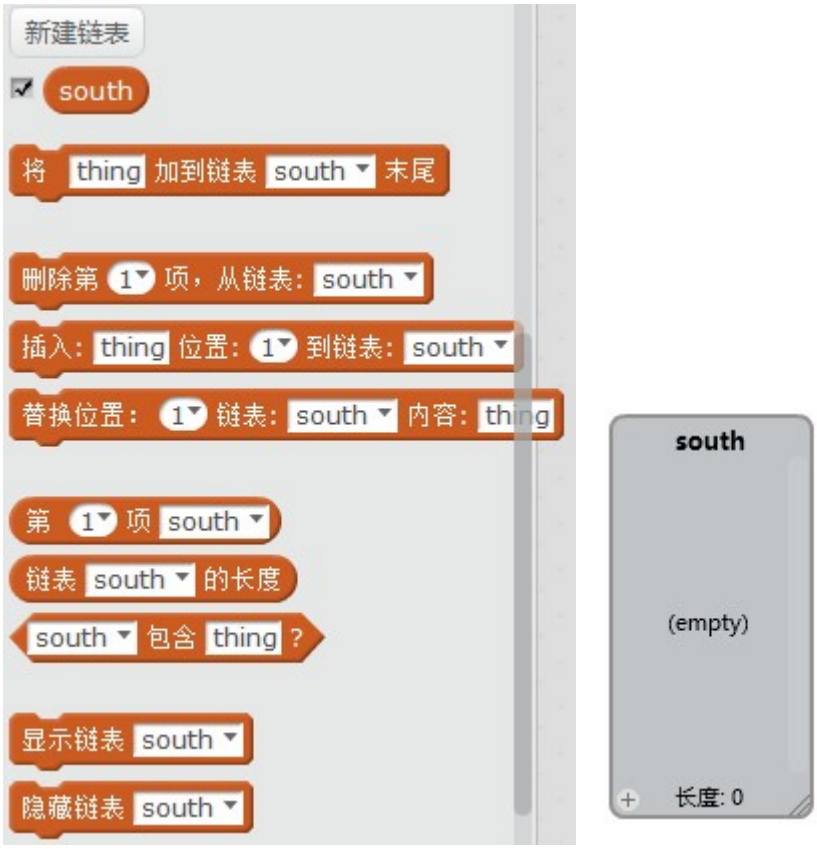
链表是变量的容器，每个变量在容器中都有一个唯一的编号。第一个容器中的变量，第二个是 2 号变量，第三个是 3 号变量，以此往后类推，如果现在 1 号变量不需要，将它删除，

那么 2 号变量就自动变成 1 号变量，3 号变成 2 号变量，以此往上类推。我们都知道变量的引用方法是通过变量名，而链表中变量的引用就是通过唯一的编号。换言之，链表可以在程序运行时动态的管理变量。

鼠标点击新建链表 

新建链表：

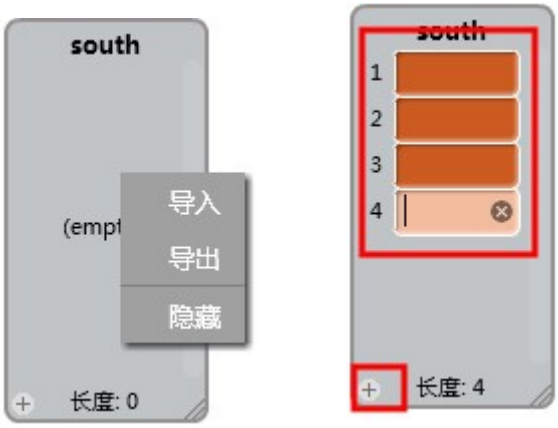
其中“适用于所有角色”和“仅适用于当前角色”的区别跟“变量”是一样的。添加过后积木区会出现相关积木，舞台模块出现链表值显示器：



“将...加到链表末尾”积木直接将变量添加到容器的末尾，“删除第 n 项”积木除了可以删除某一项之外，还可以清空整个链表的变量：

“链表的长度”积木可以得到容器中变量的个数，“包含”积木可以检测容器中是否包含某些变量。其他积木块都很好理解，就不在一一描述了。链表值显示器支持导入导出文本文

件，每行代表一个变量：

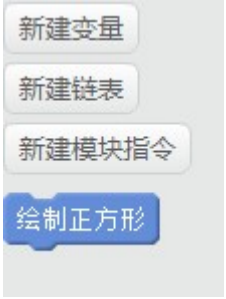


添加和删除链表除了使用积木添加变量，还可以使用链表值显示器中链表功能非常强大，可以说是高级程序中必不可少的要素，即使程序关闭链表中的数据也不会被删除，因此完成存档的功能，使数据不会丢失。

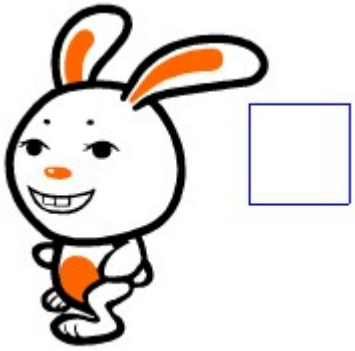
新建模块指令

新建功能块是一个非常重要的功能，不仅因为其功能强大，更重要的是它的一种编程思想就是模块化，这种思维方式不仅在编程中有体现，而且生活中也很常见，如我们做事一样会有一定的计划，分为几个步骤，每个步骤就是一个模块。不仅可以将脚本分为模块化，也可以将相同的脚本整合在一起。下面让我们来建立一个模块，点击新建模块指令会出现如下对话框：

每一个模块需要一个名称来说明它的主要作用，像绘制五角星，输入文字，点击确定后，在脚本区和积木区都会出现相对应的模块：



这样我们就可以直接使用并绘制正方形了，但是提前还需要完成“定义”部分：

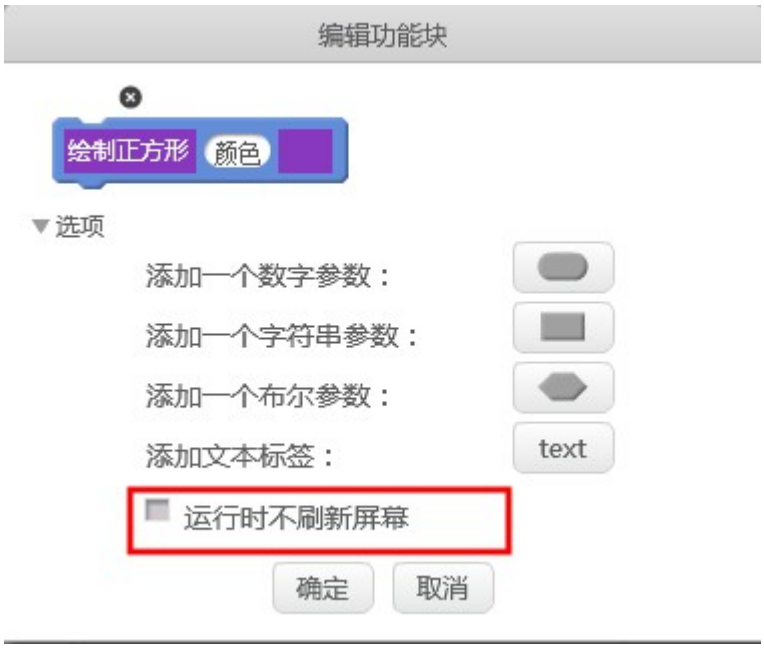


这样就绘制成功了，如何更加智能的绘制两个不同颜色的正方形呢，我们还可以使用参数，右键“绘制五角星积木”或者点击“定义绘制五角星积木”选择编辑，打开选项，添加一个数字参数，将参数修改为“颜色”，最后点击确定：

这时软件中的积木就会发生对应的变化：

这时我们只需要将程序修改为下图所以就可以：

还可以根据自己的需要在添加一些模块，方法跟上面介绍的一样，就不在操作了。下面介绍一下“运行时不刷新屏幕”的作用，所谓的不刷新屏幕就是指舞台直接展现出每个模块运行到最后的效果，下面举例说明一下：

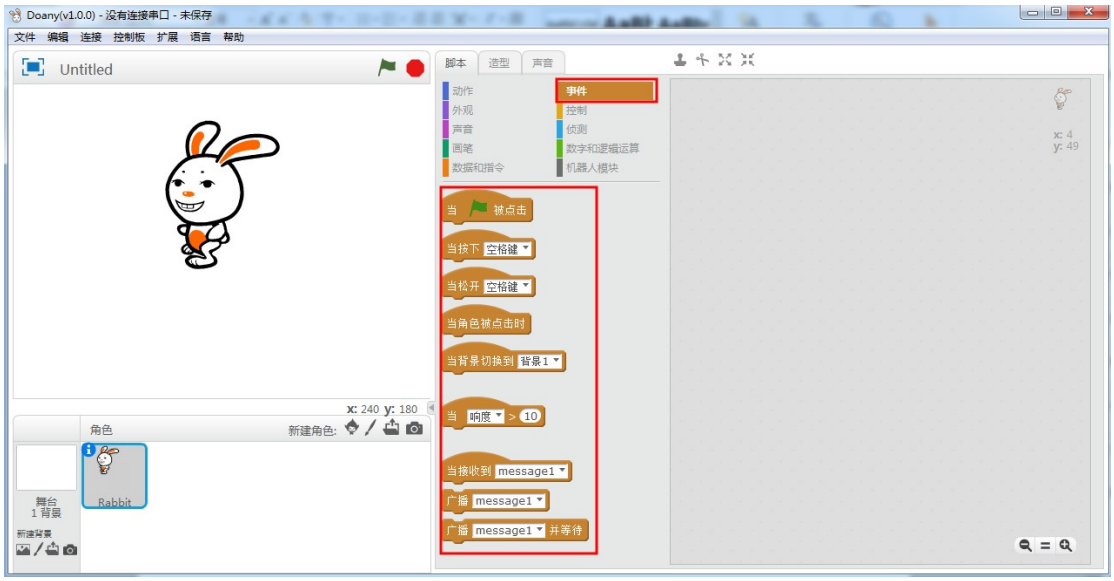


运行移动模块，此角色不会依次向前移动，而是直接移动到最终位置，如果某个模块运行时间过长，就可以使用此选项。

事件

何为事件

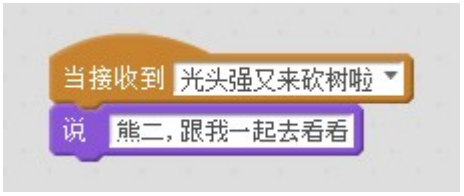
所谓的事件是指被动的接受由外部触发的事件，这是一种接近人类的思维方式。例如，大多数人都是通过一大早起床看天气穿什么衣服，下雨天带伞，天气就是一个外部事件，导致我们是否带伞，穿什么衣服的这种行为。就是被动的接受一个事件并做出行为。生活中有很多事情都是被动事件。那么在我们 Doany 软件中，事件是怎么样的，在软件中事件都是由一块块上方半圆形的积木构成：

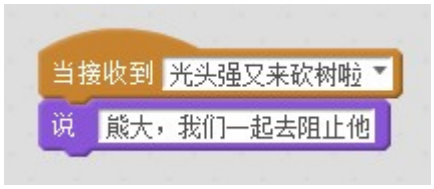


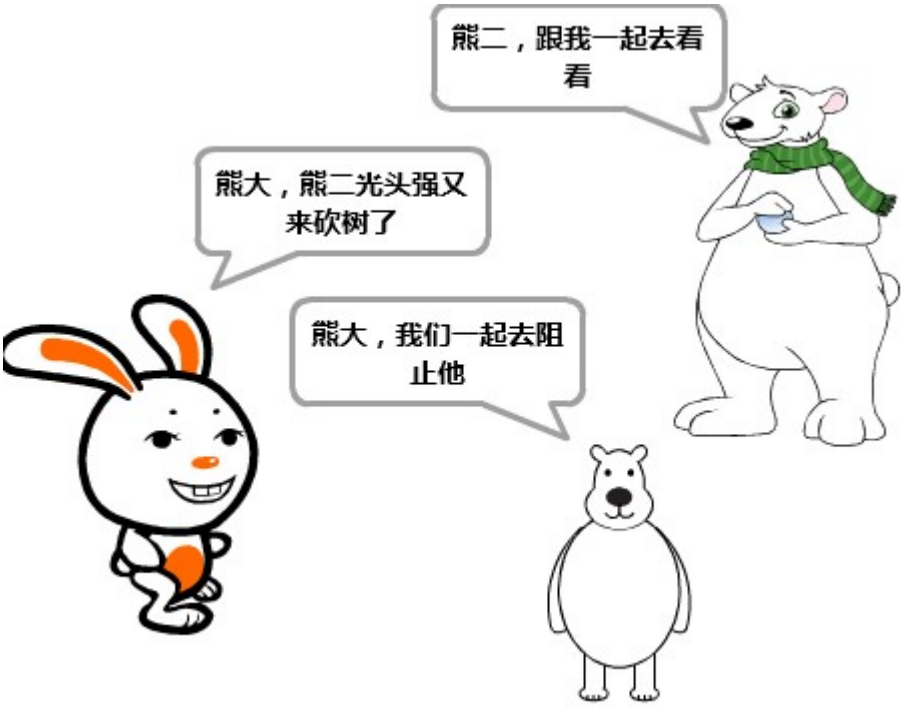
上图中的这些积木都是外部事件，因此他们都是脚本的第一个起始积木，由他们作为触发执行脚本，比如，当绿旗被击中时，它的外部事件就是舞台上的小绿旗，一旦点击就触发这块积木，“当按下”积木，它的外部条件就是用户按下了键盘上的某个键，来使脚本运行，鉴于事件的含义比较好理解和掌握，这里就介绍两三个积木，其他的积木性质都是一样的，使用方法也差不多。

广播

广播我相信大家都知道，就是一个角色把一个消息传递给每一个角色，让别人都知道，当别人接收到这条消息时，就会做出相应的反应。因此，广播就包含了两个部分，发送消息和接收消息。下面通过角色对话的例子来给大家介绍下这两个阶段：





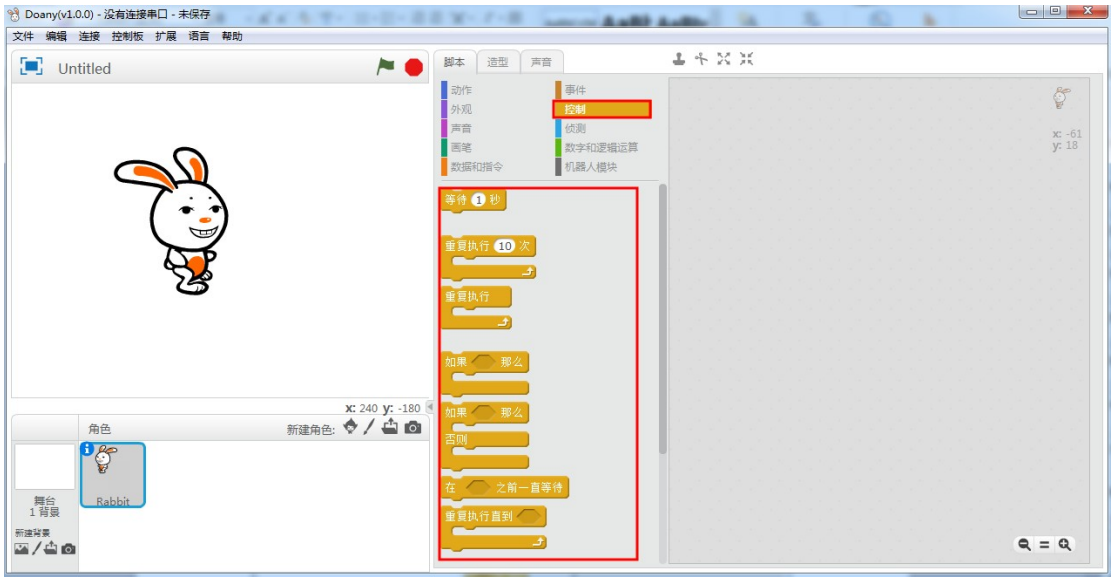


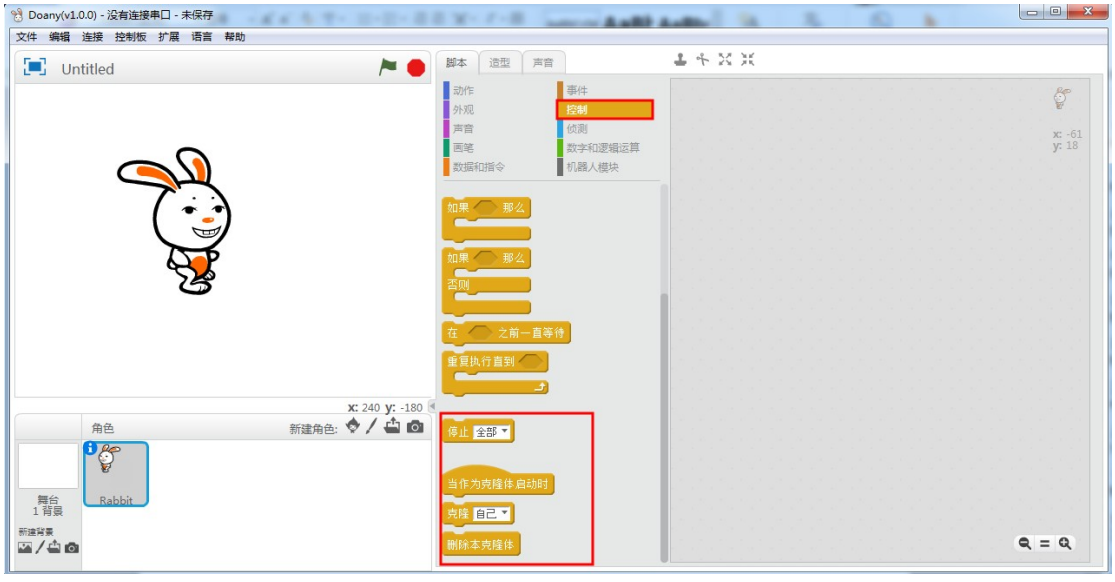
这两个积木的区别在于，前者广播玩之后就会执行下面积木的程序，而后者在广播之后会等待所有人接收到消息后在执行下面积木的程序。如果你写的程序需要严格的协调，同步角色间的行为，则必须使用后者，如果没严格要求，使用前者积木就好。

控制

控制指令区

下图中的红框区域就是下面要介绍的控制指令区：





等待

积木就是无条件的等待，通常用于延时操作； 

这块积木是有条件的等待，适用范围很广， 

这个模块就是在空格键按完后才能执行下面积木模块。

分之结构

分支结构涉及到的积木如下所示:

如果结构是我们编程过程中必不可少的一部分，非常重要，因为它是一种用来判断的结构，如果这样做，否则那样做。第一块积木为单分支结构，第二块积木为双分支结构，两者之间还可以进行组合，形成多分支结构，如下图所示：

循环的使用

循环的积木有三个： 

下面分别介绍下，第一块积木是无限重复，专业的叫法叫做死循环，通常用在持续的检测某些事件，注意这块积木的下方是平的，说明它的下面不能连接任何积木。第二块积木是次数重复，适用于知道次数的情况下进行重复执行，第三块积木是直到型循环，或者说是有条件的重复，适用于知道某个条件去不知道次数的情况下使用。下面介绍一些例子来说明这三种循环模块

的使用：

一直向前走

绘制五角星

停止

有的时候，程序运行过程中需要停止，就要用到停止模块：

停止模块它有三个选项，第一个选项就是停止整个项目的全部程序，第二个选项就是停止当前积木的脚本，第三个选项就是停止本段脚本之外的所有脚本。

克隆

有的时候，在程序运行时如要复制角色，可以在角色列表或者舞台中右键进行复制，但是这种做法是在程序运行之前就确定数量的，如果在这之前，复制的数量无法确定时，那么克隆就实现了复制角色的功能。主要的积木有

和删除积木

，下面来简单的制作一个克隆案例：

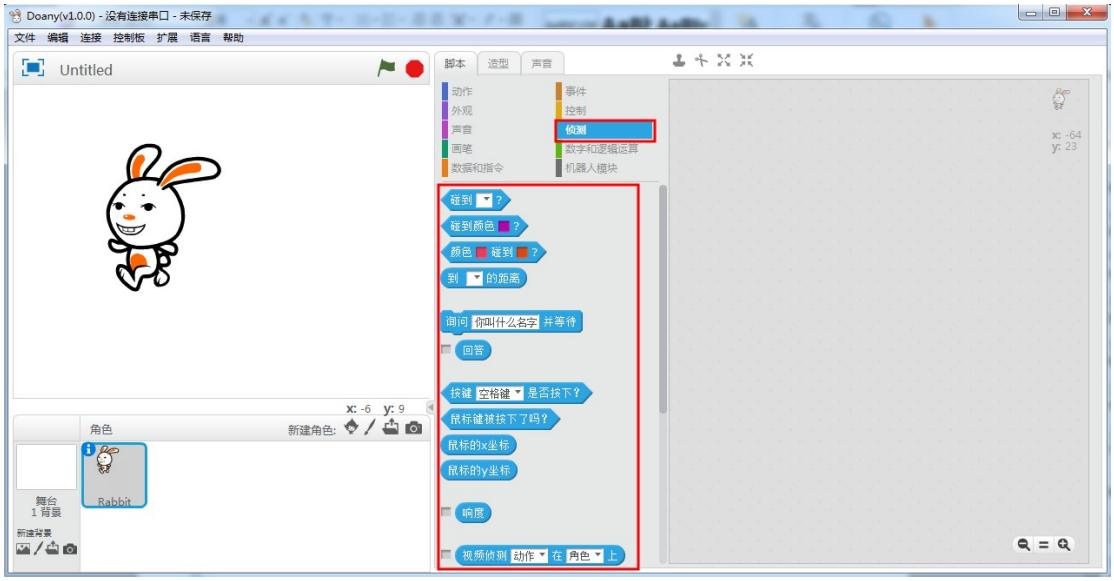
每当击中绿旗后，先隐藏自己，然后重复的移动到某个随机的位置，克隆自己。

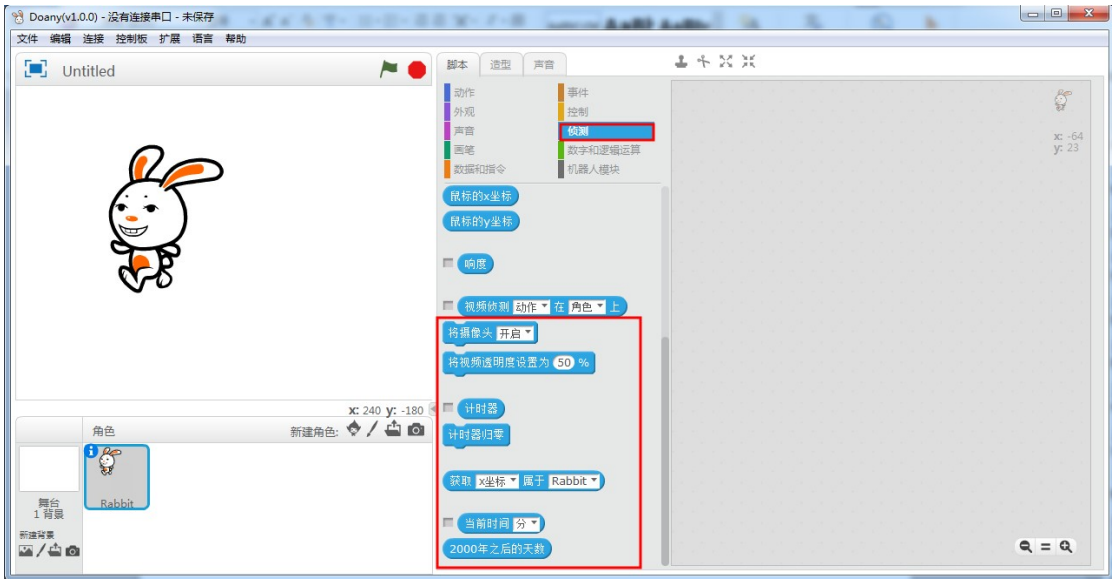
当每个克隆体启动时，首先将自身显示出来，然后不断地减少 y 轴下降的过程，同时判断是否碰到边缘，碰到边缘就删除克隆体。

侦测

控制指令区

下图中的红框区域就是下面要介绍的侦测指令区：



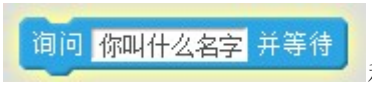


碰撞模块

碰撞设计是一些游戏中必不可少的功能，在软件中对应的积木有以下几个：

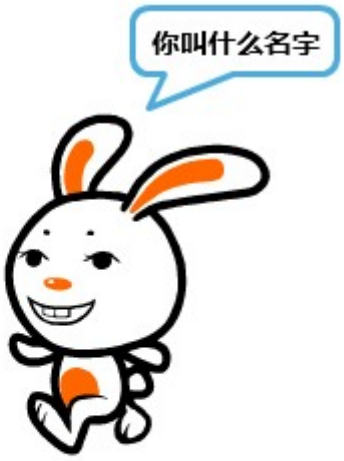
第一块积木可以侦测是否碰到鼠标，边缘，其他角色，第二块积木侦测是否碰到什么颜色，第三块积木侦测本角色的颜色是不是碰到其他的颜色，第四块积木可以算出到鼠标指针或其他角色的距离。

用户输入

所谓用户输入就是根据程序要求，输入一些信息，例如人机交互时，我们可以用下面的积木获得用户输入的信息： 

和 

输入的内容会被保存在“回答”的积木中，下面就是程序的显示结果：



按键鼠标

在一些程序的运行时，可以通过键盘输入，用鼠标进行交互，在 Doany 软件中的积木有以下几个：

第一块积木是侦测用户是否按下某个按键，第二块积木侦测到鼠标的左键和右键，最后两块积木显示鼠标的当前所在位置。

音频和视频

Doany 的交互还可以使用音频和视频（需要硬件支持）：

第一块积木会获得麦克风的响度，第二块是视频交互的关键，它有四种排列方式：

动作是视频在舞台/角色中移动幅度的表现，移动幅度越大，动作值越大，方向是视频在舞台/角色中移动方向的表现，右方向和上方向是正数，左方向和下方向是负数。
第三块积木可以将摄像头进行开启和关闭，第四块积木设置舞台中摄像头的透明度，0 为完全不透明，100 完全透明。

计时器

计时器就是任何编程都必须提供的功能了，Doany 的计时器功能有点偏弱，不仅只有一个计时器，而且无法停止，当然它作为入门级的编程语言，仅使用以下这两块积木是可以满足大部分程序的要求： 

，使用计时器通常要将它归零，你会发现每次将其归零，都会从 0 开始计时，因为无法停止，所以需要使用变量记录时间或者使

用积木展示某一刻时间： 

获取属性和时间

每一个角色都有许多属性，如坐标位置，方向大小，当前造型等。如果需要从一个角色去访问另一个角色中的属性，就可以使用下面的积木来完成：

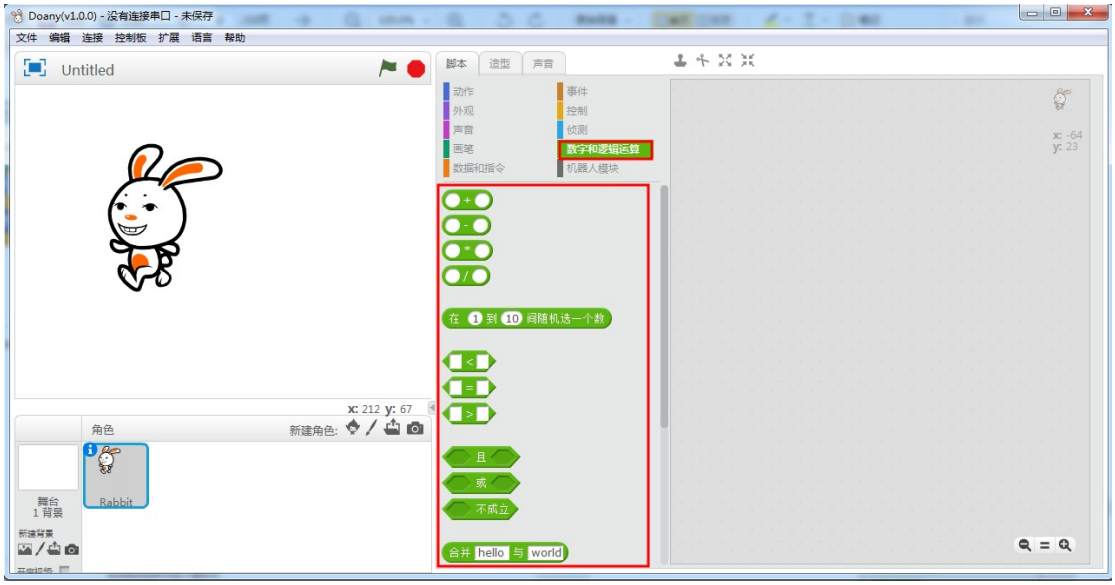
一些时间相关的积木如下所示：

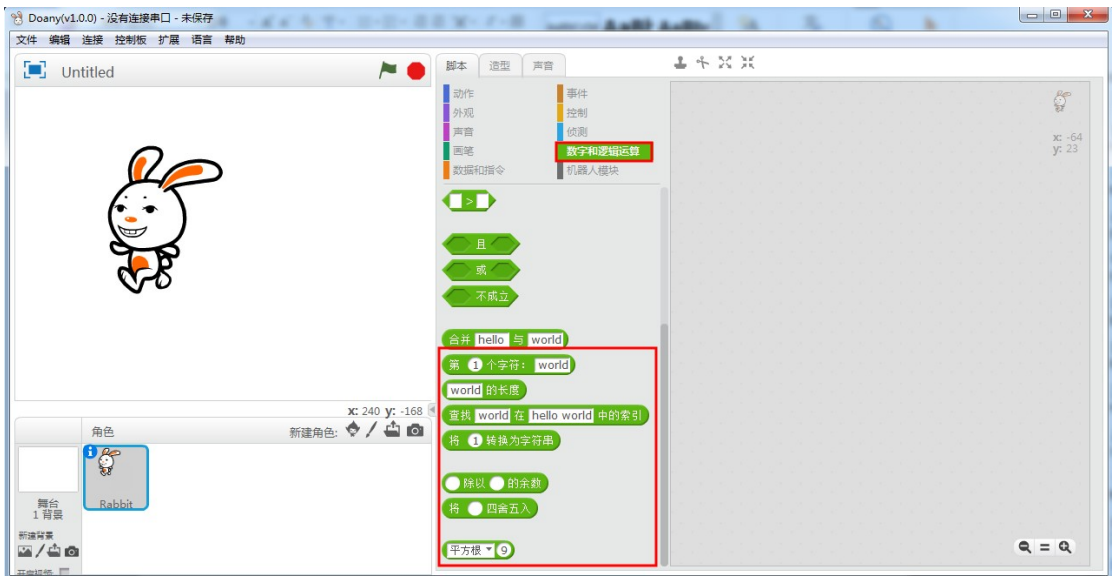
积木本身很简单，基本看一眼就知道怎么去用，可以做一些跟时间相关的程序，如日历和钟表等。但是需要注意的一点就是当前时间信息需要减一才符合中国的习惯。

数字和逻辑运算

数字和逻辑运算指令区

下图红色区域就是数字和逻辑运算指令：





算数运算

算数运算估计就是加减乘除和取余数：

比较简单就不在叙述了。

逻辑运算符

我们的生活中，无时无刻都充满了逻辑运算，本软件有三个逻辑：

且运算就是要两个都成立才能进行下面积木的程序，或运算就是只要一个成立就可以进行下面的积木运算，不成立运算就是如果需要按下，而你没按下就成立，会进行下面积木的运算。可以通过下面程序进行了解。

关系运算符

关系运算符比较重要，在一些逻辑逻辑转换到计算机语言中，本质上都在比较数值的大小，等于关系。以下为关系运算符的积木，比较简单，就不详细阐述了。

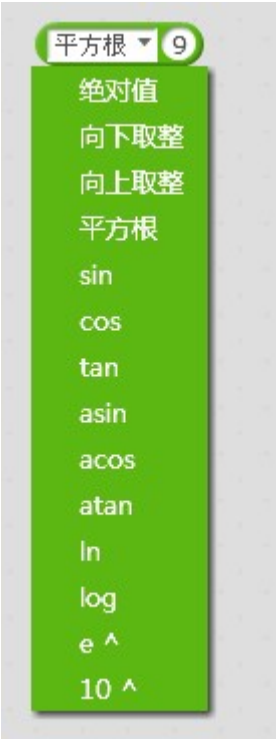
数学函数

数学函数对于程序非常重要，我们可以使用它进行很多计算：

随机模块的应用场景比较多，如设置随机时间，随机号码等，此积木可以生成连续的随机整数，小数等，比较简单。

四舍五入模块，使用非常简单，默认把小数四舍五入成整数位：

像一些数学函数用的最多的是第三块模块：



字符串处理

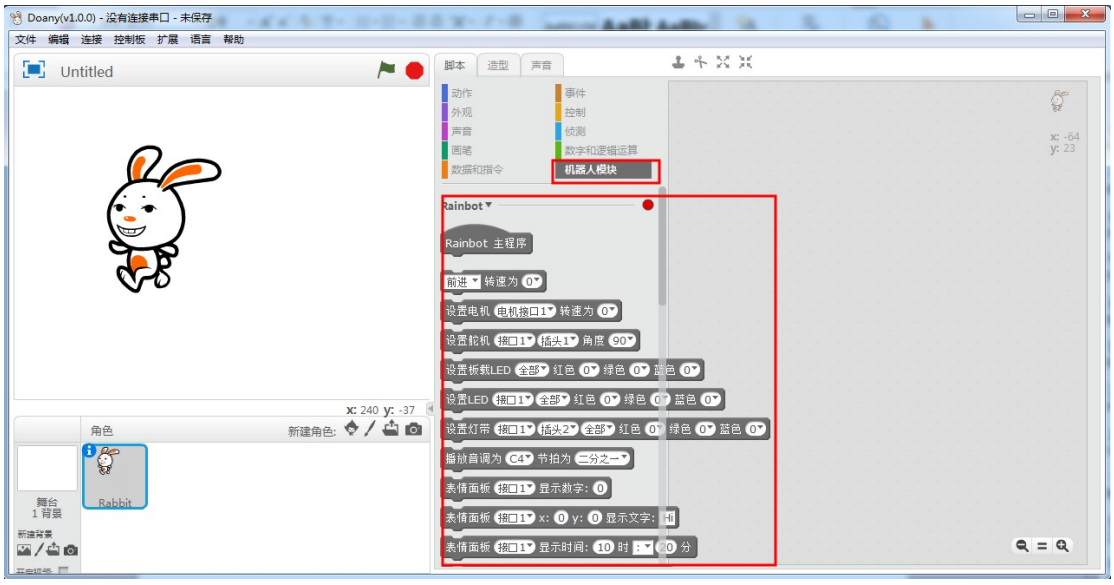
字符串即数字，字母，符号，的集合，字符串也可以进行许多操作，可以计算字符串的长度啊，字符串之间可以进行比较，下面讲一下字符串的应用。

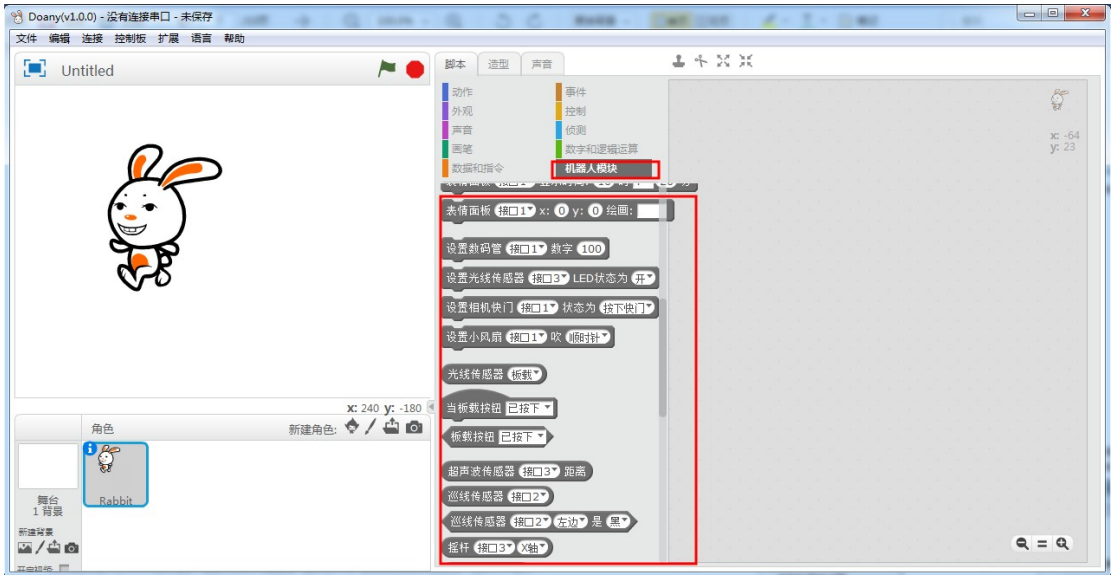
字符串的长度即字符串中包含字符的数量，当我们需要遍历字符串中的每一个字符时就需要使用第二和第三块积木了，字符串的合并也比较简单，如下所示：

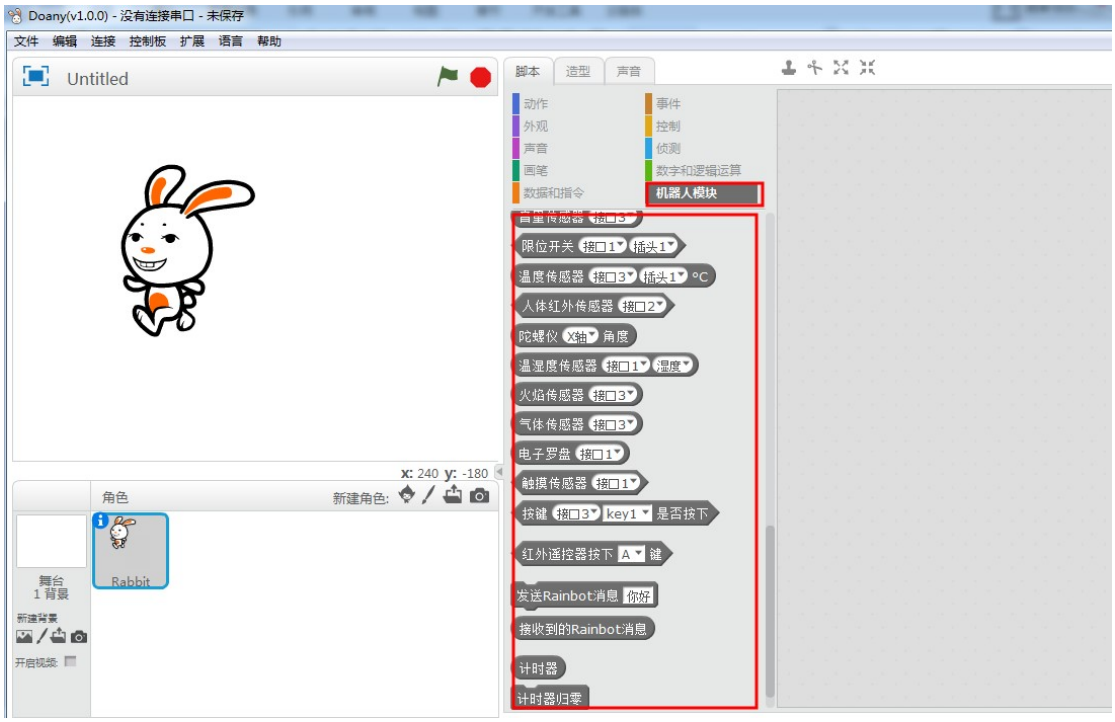
机器人模块

机器人模块指令区

下图中红色区域就是接下来要介绍本软件的最后一个机器人模块指令：

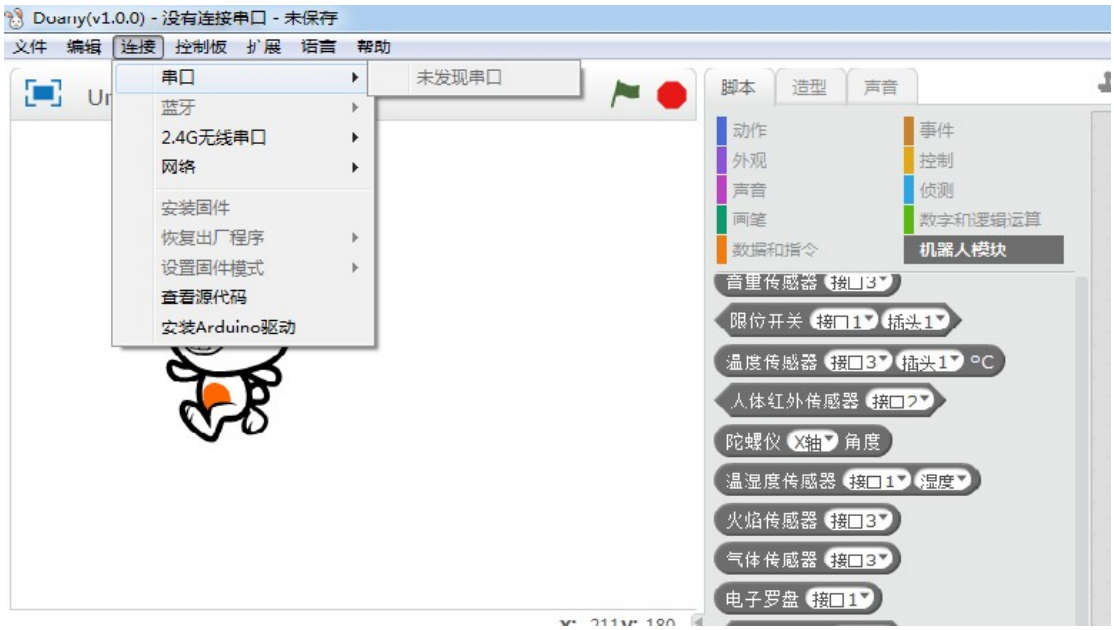




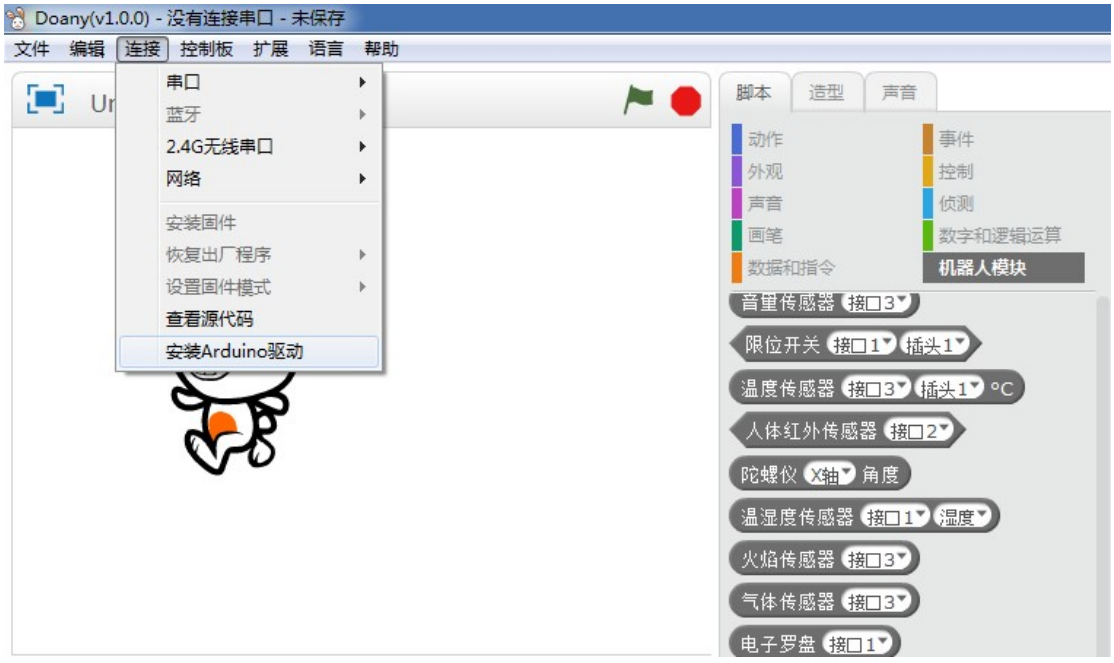


连接

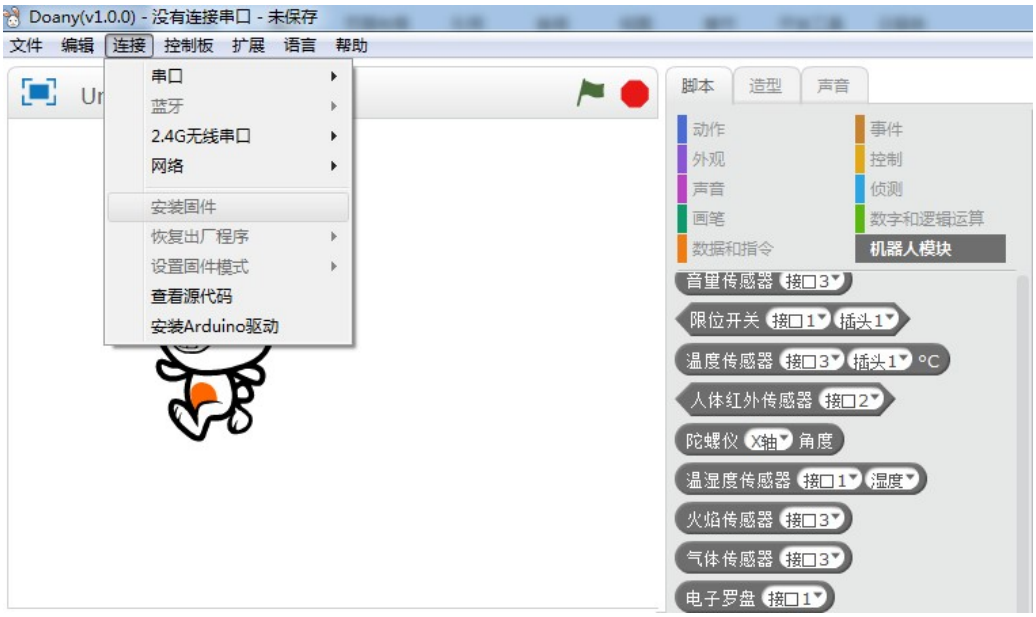
我们需要将硬件和软件进行连接，连接方式有 USB 数据线，蓝牙，和 2.4G,首先选择主控板和电脑连接，选择 COM 端口。



如果设备端口找不到端口号，先安装 Arduino 驱动后在查找并选择端口：



Doany 成功的连接了主控板，但是他们之间还不能进行通信，所以我们需要将协议安装在主板上，选择当前主控板，在选择安装固件：



下面讲解蓝牙的连接方法，打开电脑的蓝牙开关，断开之前的 COM 端口连接，在菜单栏中选择蓝牙，发现，然后出现蓝牙列表，选择对应的蓝牙，连接就好了（当蓝牙未连接时，蓝牙模块蓝灯闪烁，连接成功后，蓝牙模块蓝灯常亮）。

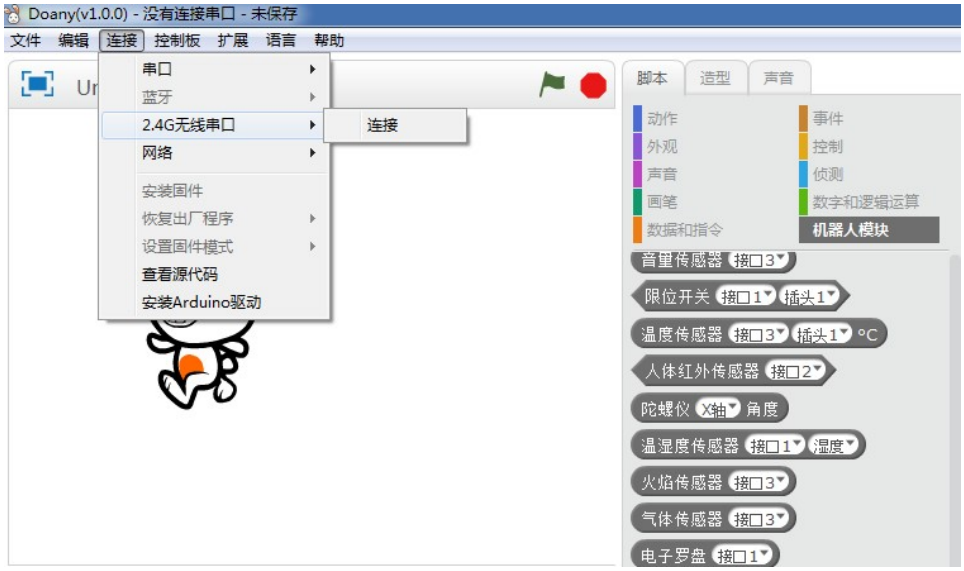
2.4G 目前仅支持 Rainbot 小车，其操作简单，将 2.4G 模块连接到主板上。

2.4G 有两种模式

第一种是慢闪模式：2.4G 连接到主控板，上电，2.4G 模块蓝灯会缓慢闪烁，说明只能跟上一次适配器配对，不能跟新的配对。

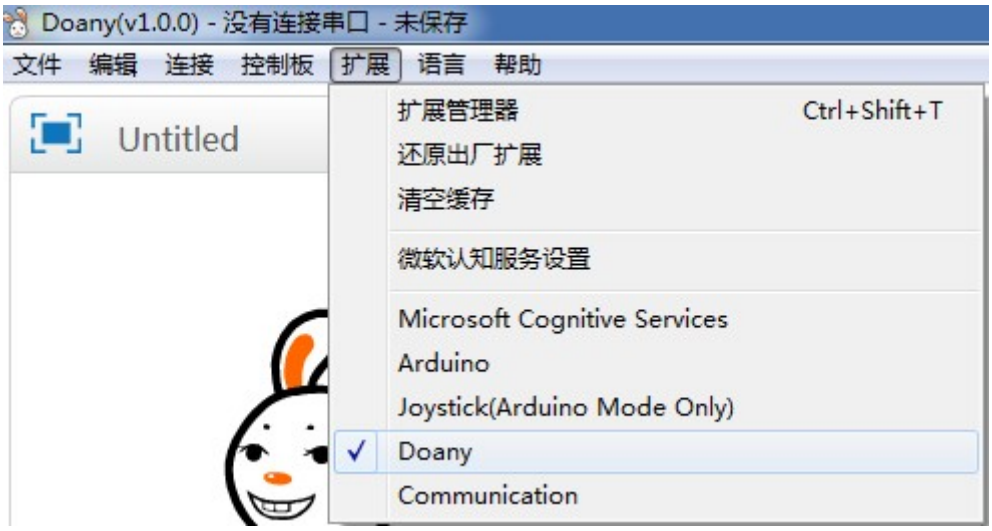
第二种是快闪模式：2.4G 连接到主控板，上电，2.4G 模块蓝灯会快速闪烁，就会立即连接，比较方便。

配对成功后，蓝灯会常亮，选择 2.4G 就可以无线通信了。



编程

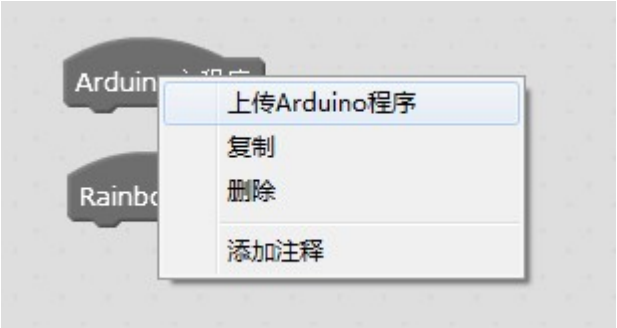
机器人模块中的个部分指令都是在菜单中选择的，下图选择了 Doany 主控板：



还可以选择一些 Arduino 系列的主控板，Joystick 系列的，根据自己的需求进行编程，达到比较好的效果。

上传到主板

每一个程序编程过后就需要将程序上传到主板，这样就能够进行脱机工作了，主要过程就是将积木模块的程序转化成 Arduino 代码，直接烧录到主板上，就可以了。



上面就是将程序上传到主板了，非常的简单，希望大家能够多多学习。