木夕逻辑插件

系统教学 第一讲：编辑器基础及数据类型

(一)编辑器基础

木夕逻辑插件，大体上由两部分组成，就是 常量库 和 scene 制作器

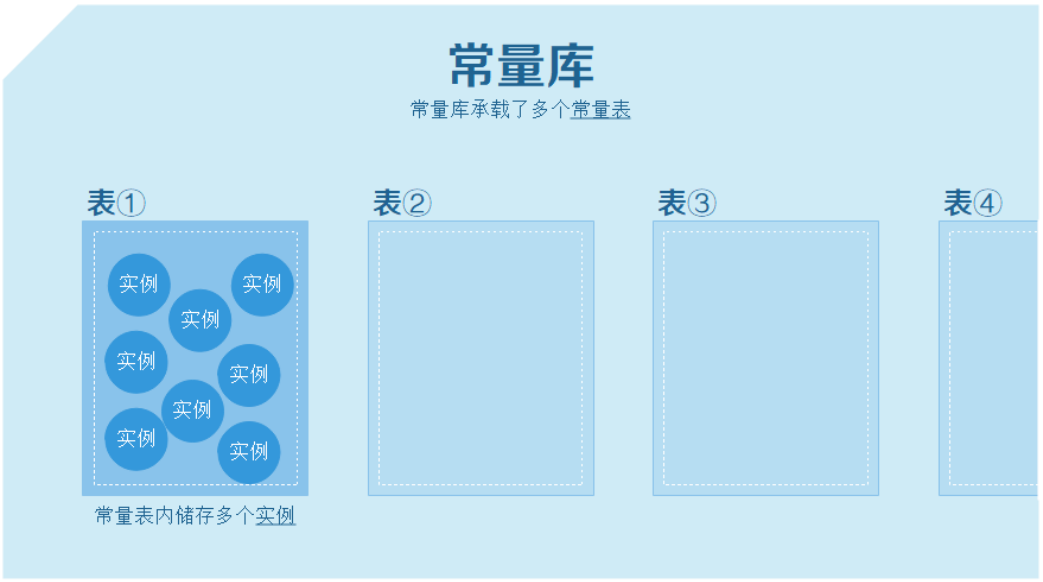
类似数据库
由多张常量表组成
每一张常量表中可储存多个实例

页面制作器
我们将制作出的页面统称为scene

1. 常量库

（图：常量库实际界面）

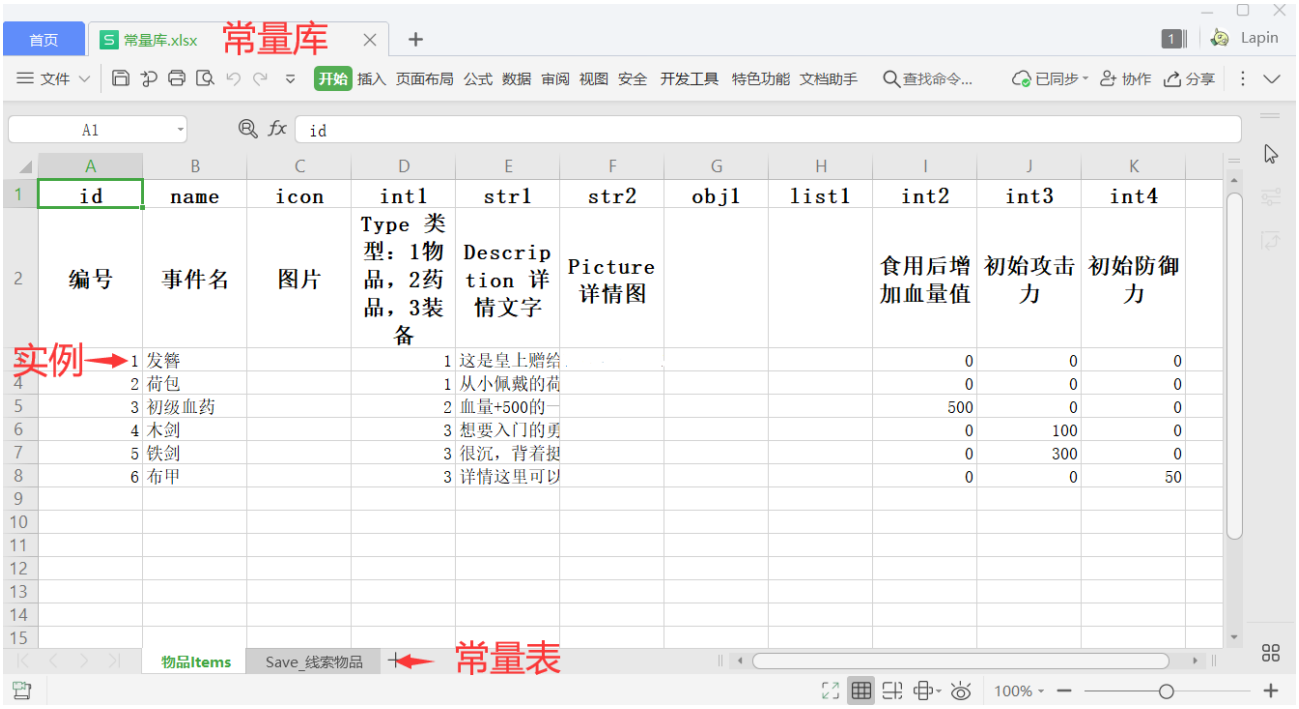
常量库 指的是用于存储 常量表 的库结构



如图所示，常量库内可存储多个常量表，表与表之间没有任何关系

常量表 则是一种用于存储 实例 的表结构

用 excel 来说明的话，大概就是这个样子



狗教官

上面我们说了好几个名词，分别是 常量库 常量表 实例

是不是已经有些懵圈了？不明白的话就仔细看上面那张示意图吧！

重要概念介绍

什么是实例？

发簪

这是皇上赠给我的发簪

类型：物品

血量+0　攻击力+0　防御力+0

以这个发簪为例，上图中，左侧是不是很像游戏作品中发簪正常的展示状态呢？

这个发簪，它其实是由多种信息组成的，这些信息包括：这个发簪的图片、类型、各项数值等。

这个发簪就是一个实例！

简单来说，“实例”就是一个拥有多种属性的东西，它可以是一件装备，可以是一个角色，可以是一本武功秘籍等等……如果还不明白，我们可以以人物举一个例子。

实例名	他的介绍	搞笑	恋爱	热血	科幻	恐怖
岳云鹏	DYS 男团门面担当	9	2	4	4	1
宋小宝	DB-boys 男团门面担当	9	1	3	3	2
吴亦凡	DWKM 男团门面担当	4	7	7	6	5

上面以表的形式，展示了三个人物，他们每一个人物都拥有多种属性（介绍、能力值），这每一个人物就是一个实例。

注意：同一张 常量表 内的实例，其属性类型（即表头）是相同的

还是以这个发簪为例

发簪

这是皇上赠给我的发簪

类型：物品

血量+0　攻击力+0　防御力+0

右图中就是以填表的形式展示出了这个发簪的各项属性

我们将这些属性分成了 数值、字符串、实例、数组 4 种类型。

如它的血量攻击力这些就是数值，而文字介绍（这是皇上赠给我的发簪）就是字符串类型。

高阶知识：为什么实例的属性里还有实例呢？

实例是可以嵌套实例的，举一个最常见的例子：一个角色（李逍遥），他本身就是一个实例，他拥有自己的等级、攻击力等，但他身上还可以装备一把武器，这个武器就是一个实例，因为这个武器也有武器自己的属性

在制作中，擅用实例和常量库，可以提升你的数据管理能力，使你的制作效率加倍

我在实例操作的界面发现了“实例库”和“常量库”，它们俩有什么区别？

好问题！我来解答一下，这里要仔细听讲。

实例库 与 常量库 的区别

狗教官

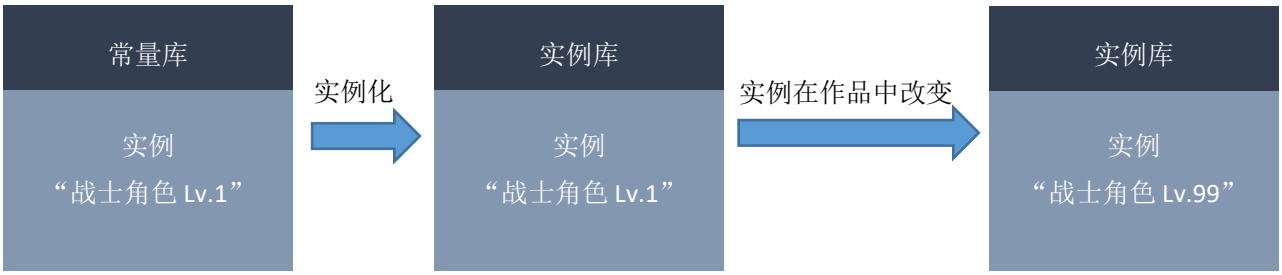
常量库：对我们来说是初始数据库，它不是用于存储当前所有数据的，而只是用于存储初始数据。

实例库：用于存储游戏中所用到的实例。

常量库相较于作品的存储数据，其概念其实更接近于作品的基础设置，是一种“设置好了就不会在作品运行过程中发生改变”的东西。

而实例库是单纯的作品存储数据，可以在作品运行过程中发生改变。

★ 所以在实际使用时，需要先将常量库里存储的实例进行“实例化”，也就是将之放进实例库中，然后使用 数值操作、字符串操作 等数据操作功能对实例库中的这个实例进行改变。



// 木夕逻辑插件 1.0 测试版本须知

当前版本仅在 scene 中，可通过 数值操作、字符串操作 等行为，对实例中的数值、字符串变量进行调整

暂时还不支持木夕工具中的以上操作

2. SCENE 制作器

Scene 的制作器大概分为三个区域：scene 列表、传入传入参数、事件区

- **Scene 列表**：显示当前作品中存在的所有 scene
- **传入传出参数区**：展示当前选中 scene 的传入参数和传出参数
- **事件区**：展示当前选中 scene 的编辑事件

传入传出参数

传入参数

是指 呼叫该 scene 时，从木夕数据向 scene 传入了一系列参数变量，这些变量赋值给本 scene 私有的临时变量

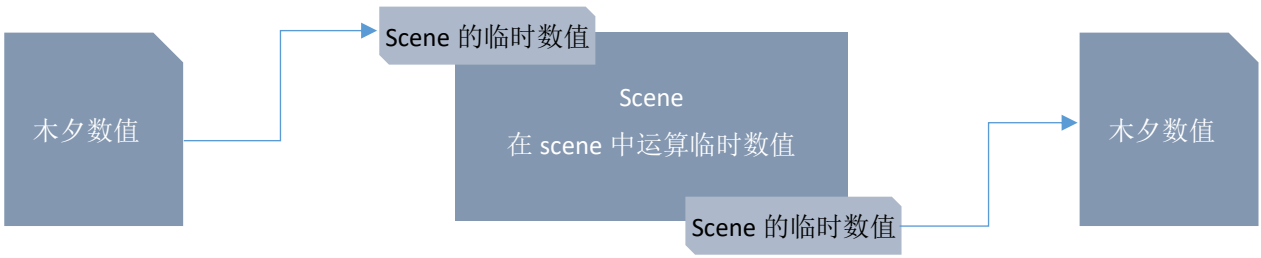
传出参数

指 当关闭该 scene 时，将该 scene 中的私有临时变量赋值给木夕数据

传入传出参数，是木夕逻辑插件的特色设计。其设计理念是希望：一般作者级用户在使用 scene 的时候，他们作品本身的数值，与 scene 里的数值，不会互相冲突。

所以每一个 scene 都会拥有一套自己私有的临时变量。

整个传出传出参数的过程图示如下：



整个过程不会对木夕数值进行运算，只会对 scene 的临时数值进行运算

事件区

事件区是编辑当前 scene 的核心区域

事件编辑大致分为 5 个步骤

1) 初始化事件 – 绘制前搞数值

在绘制 scene 的界面前，有时候我们需要先对一些数值进行处理，也就是初始化事件。

举个例子：界面上有一个显示当前钱数的元件，我们想要让钱数<10000 时显示 XXXX 元，而≥10000 时显示 XX 万元。这时候我们就先在初始化事件里判定钱数是否≥10000，然后将需要显示的结果通过一个临时变量处理出来，在绘制 UI 时显示处理后的结果。

2) 绘制 UI

通过添加元件的方式，绘制出当前 scene 的界面 UI

元件包括：图片、文字、按钮、橱窗、单选泳道、进度条

● 橱窗

橱窗是一种自定义元件

通过“橱窗制作器”，你可以设计出自己所需要的元件样式，并在 scene 的绘制 UI 步骤中添加自己设计的橱窗。

橱窗的特性：

① 橱窗是一种展示用的元件

橱窗是将“scene 向橱窗传入的参数”展示出来的一个元件，橱窗不可以对 scene 的数值进行运算和改变（因为橱窗是次于 scene 的级别，仅是 scene 的一个元件，所以不可以改变比它高的级别的数值）。
所以橱窗只有“传出参数”没有传出参数。
橱窗的传入参数，是从 scene 传入的，不是从木夕传入的。如果需要从木夕传入，就得从木夕传入到 scene，再从 scene 传入到橱窗。

② 橱窗内可以添加按钮等交互元件

橱窗内可以添加按钮，但是按钮的效果并不在橱窗内生效。橱窗内的按钮需要在 scene 中绑定 scene 的事件。

● 单选泳道

泳道，简单来说用途是：将一个数组中的每一个元素都变成一个橱窗，并将这一堆橱窗展示出来。

所以泳道所必不可少的有两个要素：一个数组、一个橱窗。
数组是这个泳道的数据来源。
橱窗是这个泳道的展示模型。

泳道设置的逻辑部分一共分三块：创建橱窗的数组、传出索引值、交互时使用的事件

- ✧ **创建橱窗的数组**：就是组成这个泳道的数组
- ✧ **传出索引值**：这个地方我们需要选择一个数值型的临时变量，来承接“如果玩家点击了泳道里的某一个橱窗，那么他点的是第几个”，也就是玩家点击了数组中的第？位
- ✧ **交互时使用的事件**：这里是根据橱窗内的按钮，可存在多个事件。在按钮绑定事件里可以根据传出索引值来进行判定和运算。

3) 绘制后的一次性处理

有时候我们需要在绘制出 UI 后，再处理一些特殊的数值或画面效果。

举例：

画面中存在 10 个图片，每次只显示 5 个，如果我们先将 10 个全显示出来再消除 5 个，画面可能就会闪现一下那 5 个本来不想让他们出现的图片，所以可以在绘制 UI 的时候，先让 10 个图片都不显示，再在绘制后的一次性处理的地方，显示出那 5 个我们想要的图片，这样就不会有闪现的效果了。

4) 循环更新

在 scene 的运行过程中，有时候我们需要实时判断一些值是否变化，就可以写在“循环更新事件”里。

举例：

我们可以在循环更新事件中，写一个条件分歧“当主角血量≤0 时，播放战斗失败的动画”。这样系统就会自动在主角血量发生变化的时候进行判定，并决定是否播放这个动画。

注意：

不建议在此处写过多的事件，以免自动判断的数值太多，导致卡顿。

5) 按钮绑定事件

按钮绑定事件与其他四项不同，它不是在“scene 的表现绘制”过程中。

它相当于公共事件。

在这里，我们可以写出本 scene 会触发的所有事件，然后在绘制 UI 的步骤中，将可交互的元件（如按钮）绑定上这里写好的事件。

按钮绑定事件的特性：

① 同一个事件可以被多个交互元件绑定

② 可以绑定橱窗内的交互元件（在 scene 的橱窗元件/泳道元件里绑定）