17 文件与程序的中转站——Resource 资源类

一、Resource 概述

①Resource 的作用

- Resource 是游戏文件与游戏程序的中转站。游戏项目内的大部分文件都要先转为 Resource 对象，然后才可以被游戏程序中的节点等对象使用。

②Resource 的继承关系

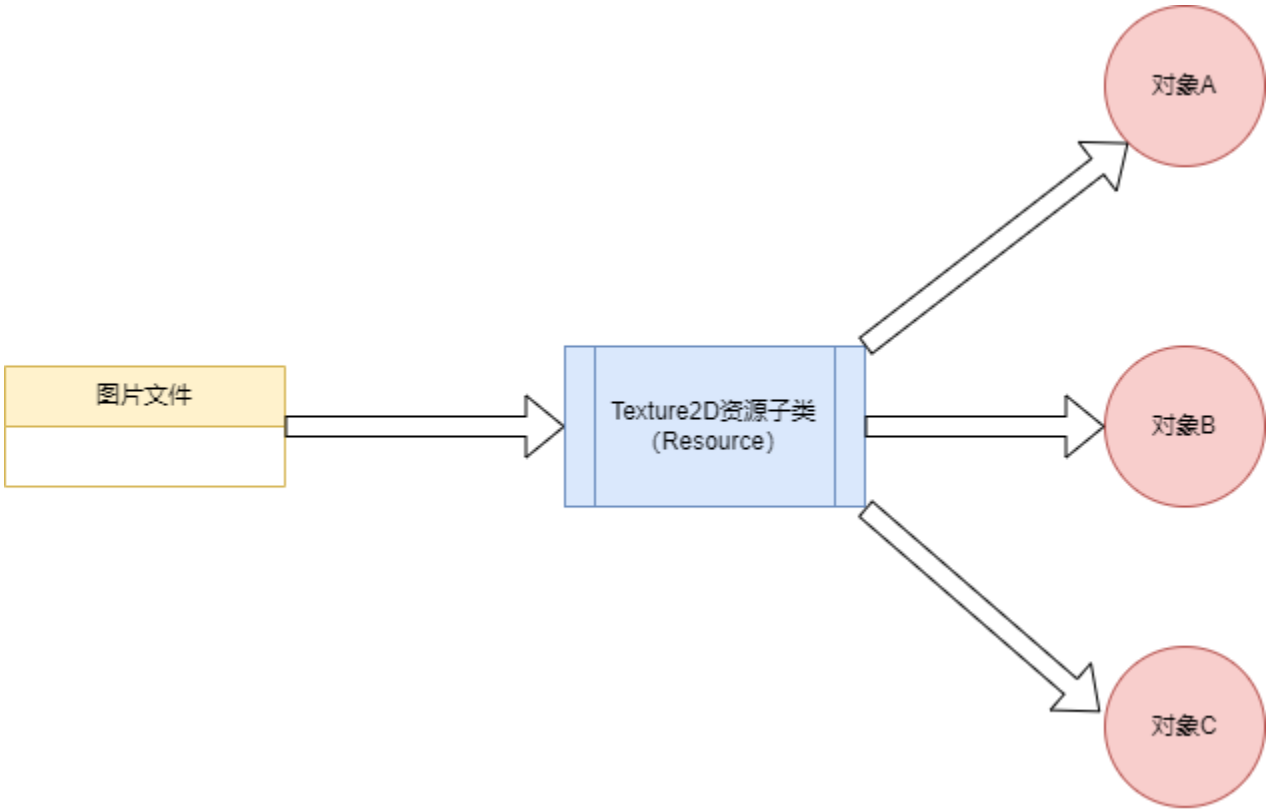
- Resource 对象继承自 RefCounted，RefCounted 继承自 Object，Resource 拥有二者全部的功能。
- Resource 可以进行信号绑定、脚本绑定等一系列 Object 可以进行的操作。
- Resource 也有内置引用计数器，计数器归 0 时，资源自动被删除。

③需要转为 Resource 的常见文件

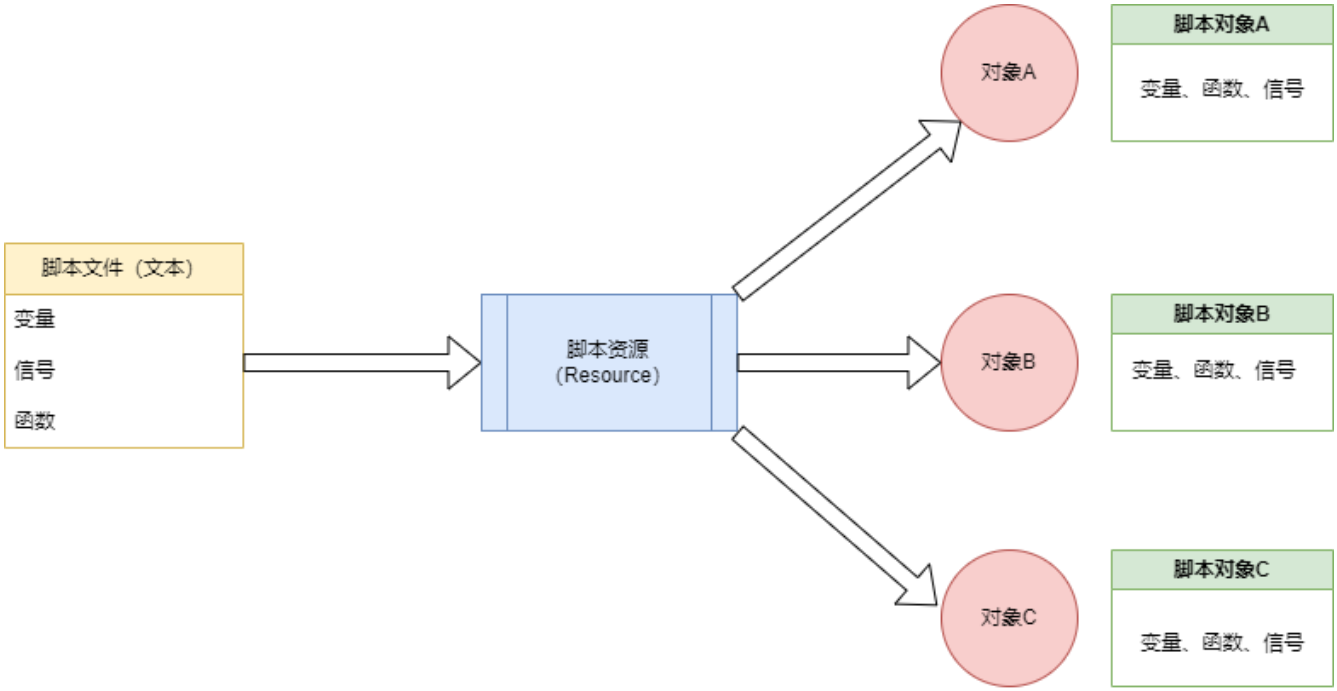
- 图片、3D 模型、音频、视频文件
- 场景文件
- 脚本文件

④资源供对象的使用的方式

- 多对象使用同一资源本身、适用于图片、3D 模型、音频、视频文件等。



- 资源经过再处理后，生成新对象，供不同对象使用。
- 脚本资源再处理生成的对象在 GDS 中不可见，这个新对象被编写在内置代码中，当我们使用 set_script 或绑定脚本的对象生成时它也将自动生成。



二、程序中 Resource 对象的创建

通过路径加载文件生成资源。

①资源随场景启动而自动创建

- 将资源文件（可以转化为资源的文件）拖入游戏项目中。
- 编辑器对资源文件进行导入，生成特殊文件以及资源文件对应的 import 文件。import 文件会指向特殊文件。
- 游戏启动后借由 import 文件加载特殊文件生成资源对象。

②load 与 preload

如果加载的文件已经被转化为资源，且此资源引用计数器不为 0，则在此加载此文件不会产生新的资源对象，而是返回一个原先资源对象的引用。

- load("文件路径")或 preload("文件路径")

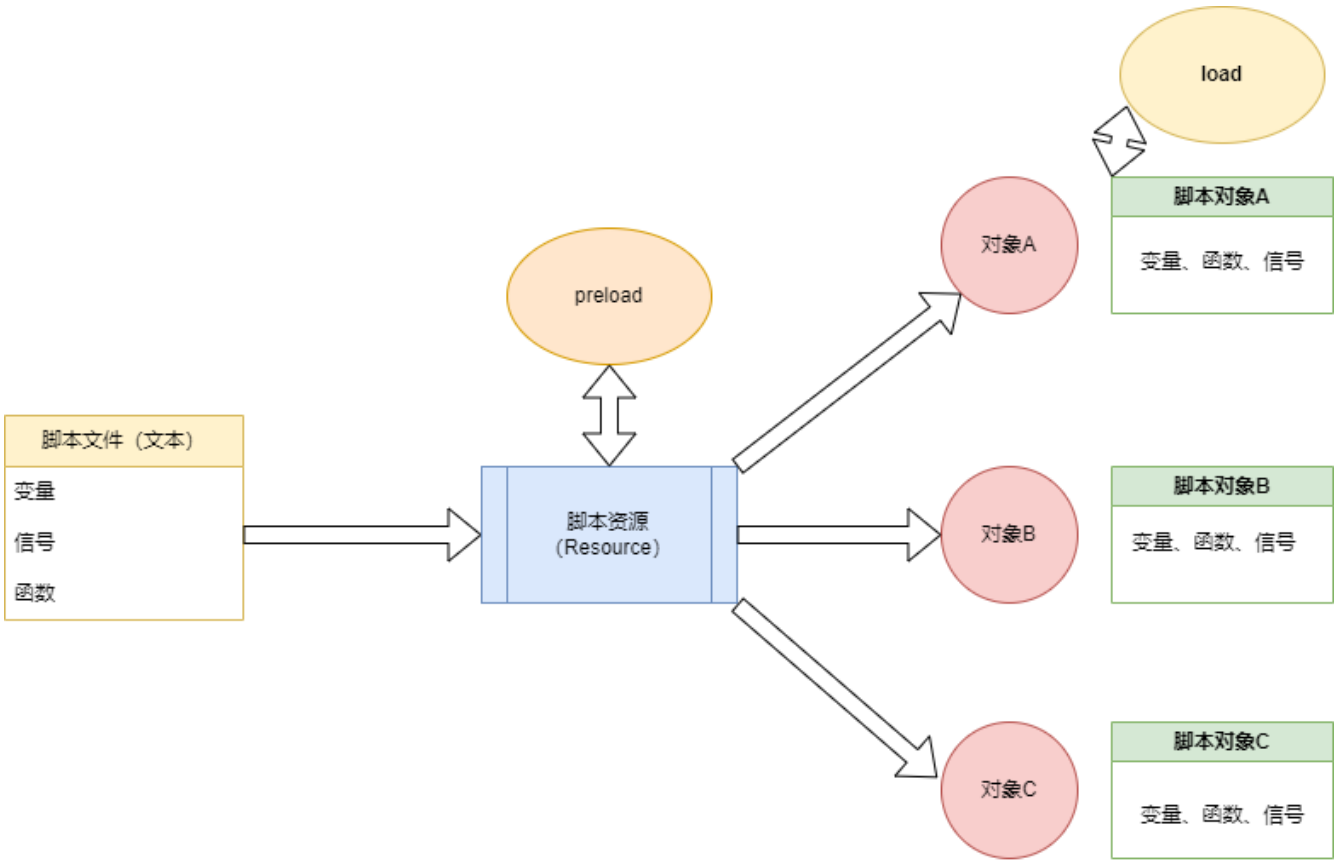
创建新对象。

③由类名.new（）创建，不会产生相同引用。

- 内置资源类：某些内置资源类也可以使用 new 以外的函数来创建资源对象。部分内置资源类内部拥有读取与保存文件的函数。
- 自定义资源类

三、load 与 preload 的区别

- load 与 preload 的参数都是文件路径，都是字符串形式，但 load 的参数可以是字符串变量。而 preload 则无法使用变量，这是由于二者执行的时机不一样所导致的。
- 当脚本文件转化为脚本资源时，转换部分的程序会自动翻找文件中是否出现了 preload 函数，若出现，则在脚本资源生成的同时进行 preload 资源的加载。这时候变量还没有出现，因此只能以文本字符串的形式来告知 preload 加载的内容。



- load 后可跟随字符串变量，字符串变量的值可以在程序运行时改变，如果改变变量，load 可以加载不同的文件。
- preload 后只能跟随固定的字符串，它不会在程序运行时改变。

四、资源保存的格式

资源对象是文件和程序的中转站，它不是简简单单的文件本身，一个资源对象可能是对一个或对个文件处理后产生的数据对象。这些处理可以是对若干图片的裁剪、压缩、拼接；一个资源对象，也可能是对节点某部分功能的记录与描述，比如 Pong 中区域检测时 shape 节点的形状就也是一个资源对象。要保存这些复杂的信息，Godot 必须准备单独的文件。

- tres ： 文本资源格式。资源的内容以文本的形式储存，资源文件具有可读性。
- res ： 二进制资源格式。简单来说就不是人话，但是程序加载速度更快。
- 某些资源会提供特殊格式来保存自己的数据。

五、Resource 对象的特征

- 加载路径产生资源时，如果原先的资源不消失，则不会产生这个文件的新资源对象，而只返回引用。在任意一处对此资源的修改，都将影响到所有使用此资源的对象。