子弹是一个比较特殊的元素，要考虑的事情比较多。我们要解决的问题有：  
1.如何绘制子弹

2.如何让子弹动起来

3.如何做子弹的碰撞检测和销毁子弹

4.我方tank如何发射子弹

接下来逐一介绍上面的问题如何实现或解决。

绘制子弹

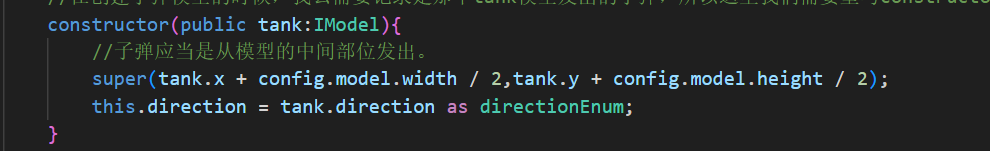
同样我们需要先创建子弹的画布类和模型类，先来说子弹的画布类应该如何创建好了：  
我们要在子弹的画布类中写一个创建子弹模型的方法，但创建子弹模型需要记录具体是那个tank模型发出的子弹，同时我们不能让子弹一直发射，应当是子弹在碰撞到画布的边界或某些特殊的模型（比如墙）时子弹消失后，才能在画布上再次创建子弹（也就是发射子弹）。

要记录是那个tank发射的子弹，我们就需要导入tank的画布类，我们利用画布类中的IModel数组记录的tank模型来创建子弹模型，在创建子弹模型的时候，我们做一个简单的判断，如果子弹的画布类的IModel数组中的子弹实例中的tank属性（一开始我们创建子弹模型的时候传到子弹模型类的构造函数中的）等于tank画布类上的IModel数组中保存的tank模型实例，说明这个tank模型已经创建过或者说已经发射过子弹了，我们就不允许再创建子弹模型。



创建子弹模型的方法

上面提到过子弹的模型类应该接收一个tank模型类的实例，所以模型的抽象类的构造函数我们就不能用了，子弹的模型类必须要有自己的构造函数。



子弹的模型类的构造函数



在模型类中我们需要重写draw方法

但这样会出现构造器签名不匹配的问题（因为我们通过子弹的模型类创建的实例上的属性包括子弹模型了类的构造函数接收的参数都受vite-env.d.ts中我们先前定义好的IModel接口的约束）所以我们要修改vite-env.d.ts文件。这里我先说文件是怎么改的，最后我再介绍什么是ts中的构造器签名，以及什么是构造器签名错误。



修改vite-env.d.ts文件

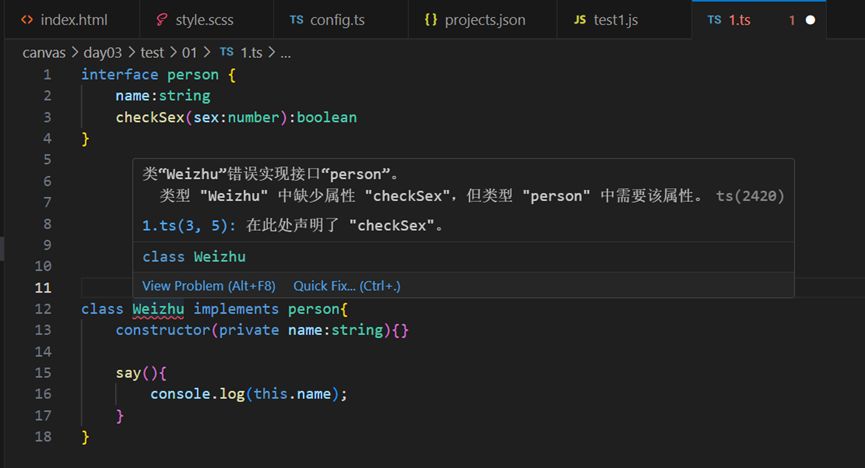
但是只是修改vite-env.d.ts文件是不够的，因为我们在画布的抽象类中将model方法定义为抽象方法，这个方法的返回值我们先前定的是只返回ModelConstructor构造函数产生的实例，现在也要将bulletModelConstructor也添加上去。



画布的抽象类也要修改。

什么是构造器签名

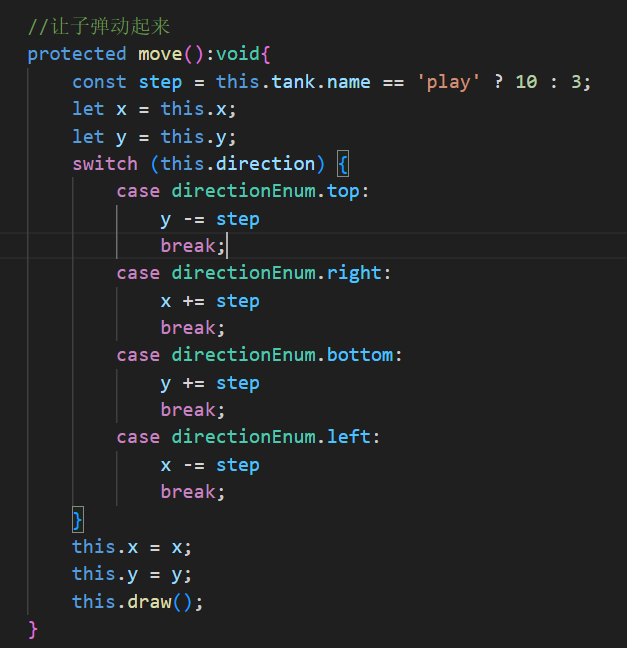
在完成了上面的工作后，其实子弹模型已经创建出来了，只不过它现在处于tank模型的正中心，我知道我们的主要任务还未完成。但这里我还是想先介绍什么是ts中的构造器签名防止你对后续的一些操作产生误解。我们知道在ts中interface其实是用于限制对象“{}”中属性及方法的参数类型和返回值类型的，构造器签名用于限制类中构造函数的参数类型，以及这个类产生的实例上要有那些方法。



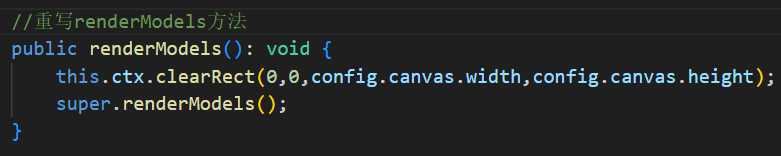
上图演示了当接口作用于类上起了什么作用，图中的提示就是告诉你你要讲checkSex方法加上。

让子弹飞起来

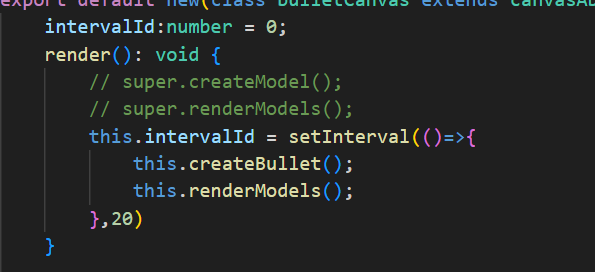
要让子弹动起来的话我们还是像以前的老套路，在子弹的模型类中写一个move方法让其坐标根据tank的炮口方向移动即可。我们可以在这个方法中创建一个step变量用于设置画布每刷新一次时子弹在画布上移动的距离，同时也可控制玩家子弹和敌方tank子弹的移动速度。



在子弹的模型类中构建实现子弹移动的方法（此时还未做碰撞检测）。



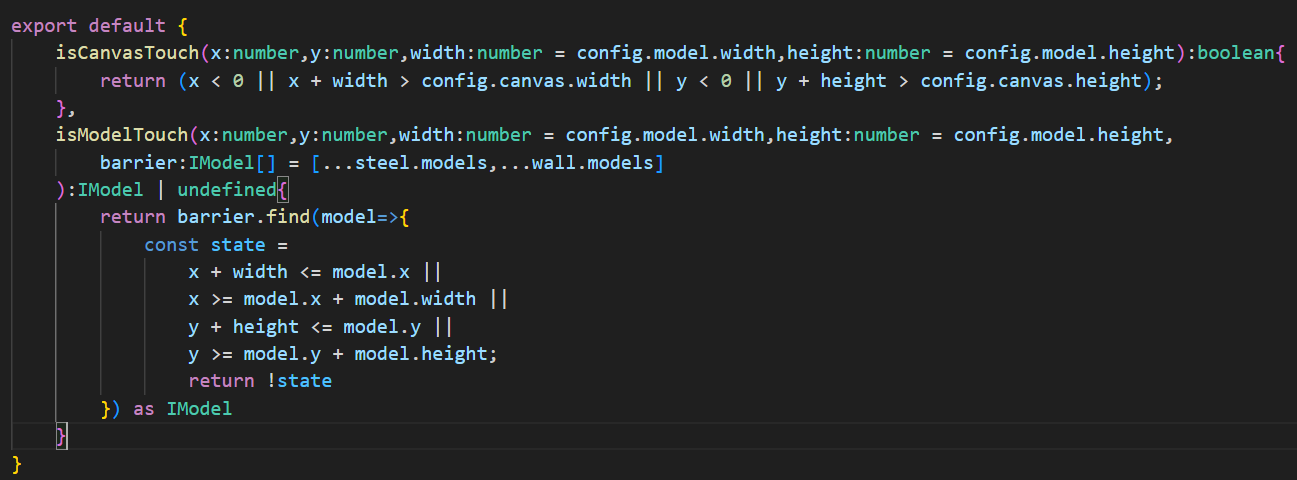
在子弹的画布类中重写renderModels方法



在子弹的画布类中创建定时器函数实现画布的不断重绘以达到子弹不断移动的效果。

子弹的碰撞检测

与之前的tank碰撞逻辑一致，我们需要在子弹模型类的move方法中实现子弹的碰撞检测。

我们先前写过一个用于做碰撞检测的方法，而且我们在之前的部分详细介绍过碰撞检测的逻辑。这里为了查看方便我还是重新贴出来一次：  


用于碰撞检测的方法，它返回的值是一个碰撞到的模型的实例。

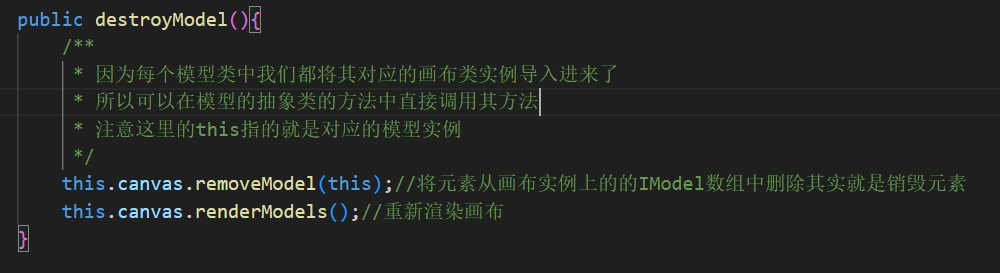
所以子弹的碰撞检测的具体逻辑就是：我们需要将所有子弹可以击中的的元素模型传入到isModelTouch方法中用于做碰撞检测。同时我们需要将子弹的坐标传入到用于做画布检测的方法中。首先我们来判断子弹是否击中了画布，如果击中了画布，就让其销毁。如果没有击中画布，那么就要判断是否击中了可击中的元素（注意tank模型自己发射的子弹不能击毁自己）如果击中了可击中的元素，先让子弹销毁，然后再进一步判断是否是击中了白色的墙，只要不是击中了白色的墙，让子弹所击中的元素销毁。所以要在模型的抽象类中写一个用于销毁元素(模型实例自己销毁自己)的destroyModel方法，供模型实例调用。



在子弹模型类的move方法中我们需要做子弹的碰撞检测

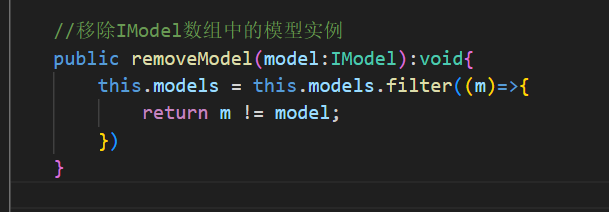
让子弹击中的元素消失

让元素消失的其实特别简单，因为我们先前在每个画布类中都设置了一个用于保存元素模型实例的数组IModel，所以我们可以在模型的抽象类中写一个destroyModel方法，在这个方法中我们调用画布实例上的removeModel方法(写在画布的抽象类中的方法)，用于移除画布类实例上的IModel数组中的模型实例，然后再重新渲染画布即可。（请注意：最终调用destroyModel方法的是模型的实例，也就是说：“那个元素需要消失，我们让那个具体的模型实例调用这个方法”）



模型的抽象类中的destroyModel方法

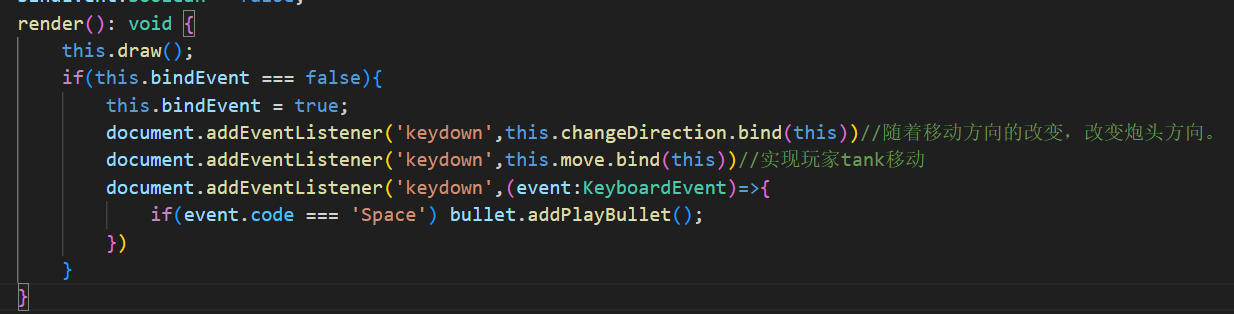
我们还要在画布的抽象类中我们要写一个用于移除模型实例的方法。直接使用数组的filter方法过滤一下就行。



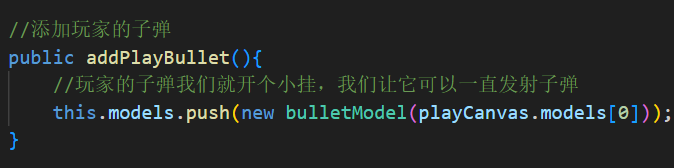
画布的抽象类上的removeModel方法

我方tank发射子弹

玩家发射子弹应该是在玩家按下空格键后。所以应该是通过给document元素绑定键盘事件，绑定键盘事件很简单。在何时给document绑定键盘事件是我们值得考虑的一件事。可以确定的是创建玩家的子弹模型应该还是在子弹的画布类中。绑定键盘事件的时机则应该是渲染玩家模型的时候。这两个方法比较简单，我们直接来看是怎么完成的好了。



在玩家tank的模型类的render方法中绑定键盘事件（其实就是说是在渲染玩家模型的时候绑定键盘事件）



在子弹的画布类中我们要写一个addPlayBullet的方法用于创建和添加玩家的子弹模型