# 论文第四版

之后调用setImageView判断结果图是否存在，如果存在则显示，如果不存在则显示没有图片，当点击分享时，首先会调用ResultContriller中的mergeBitmap（）方法，将所有的结果重新写入一张图片中，如果写入时发现结果不存在，则写入数据缺失，之后在以Intent的方式携带合并后的结果图进入到其他app的Activity

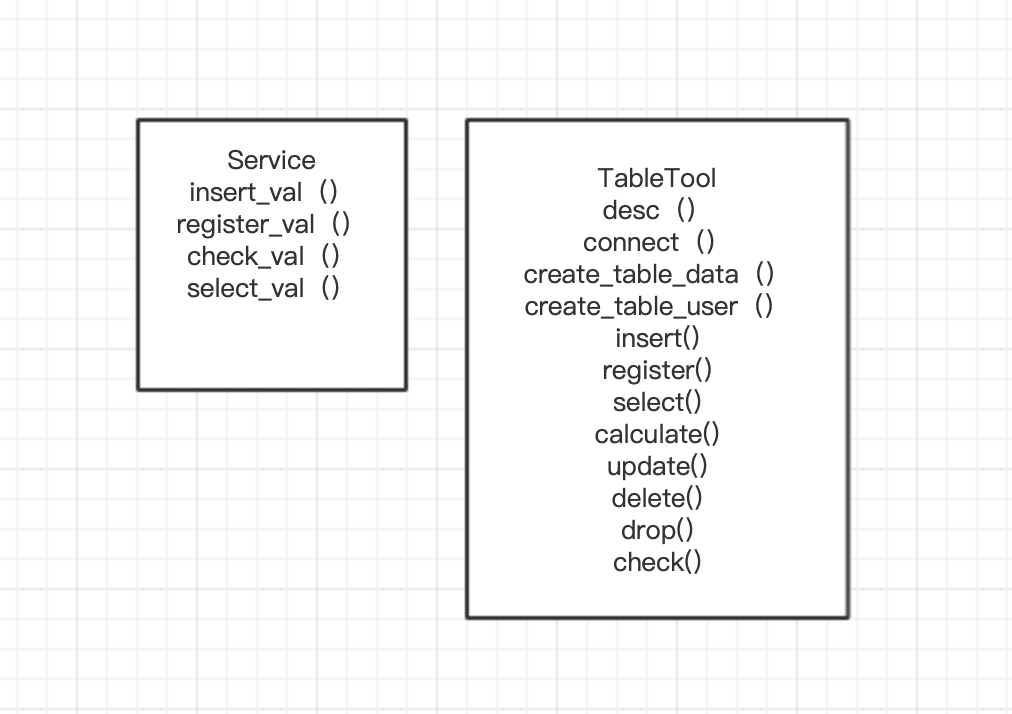
当进入到UnityPlayerActivity开始测试之前，会先调用ifStart（）方法，调用传感器来监听检测着是否确认开始检测，如果确认检测开始，则调用UnityPlayer.UnitySendMessage()方法，向unity发送消息调用其中的checking()开始检测，期间视标会在规定的范围内随机出现，当同一个位置被检测过两次后，则此位置视标不会再次出现，视标的亮度会随着视野阈值的降低而增加，从视标在某一位置出现开始到在另一位置上出现记为一次检测，在这次检测过程中，如果之前进行了对视标的响应，而本次没有响应，则假阴性次数加一，如果在视标消失的过程中进行了对视标的响应，则假阳性次数加一，如果对非有效位置上显示的视标进行响应，则固视丢失次数加一，如果对有效位置的视标进行了响应，且视标是处于显示状态，则此位置上的检测次数加一，如果没有对有效位置的视标进行响应，且视标处于显示状态，则不仅此位置上的视野阈值会降低，与其相邻的周围九个位置（如果存在）的视野阈值也会降低，每次操作控制器或者每隔两秒的时间都会调用checking（），为了保证不会发生冲突，只有当没有checking（）正在进行时，才会执行checking（），否则调用的checking（）都将视为无效，当总检测次数大于108次后，检测结束，调用ResultDisplay下的OnPostRender()方法使用opengl将结果转变成图像，为了使图像更标准，将每个需要绘制的位置作为中心点，以每两个位点之间距离的一半作为偏移量，绘制成正方形，如果记两个点分别为（x1,y1）,(x2,y2)，那么以（x1,y1）为中心所绘制的图形的坐标为（x1-(x2-x1)/2,y1+(y2-y1)/2）,（x1-(x2-x1)/2,y1-(y2-y1)/2）,（x1+(x2-x1)/2,y1+(y2-y1)/2）,（x1+(x2-x1)/2,y1-(y2-y1)/2）,所绘制的图形的颜色依据这个位置上视野阈值的高低而选择，绘制完成后将结果保存到指定位置，并将检测结果返回给Android，重新跳转回MainAvtivity

## 数据模块

### 功能描述

该模块主要用于处理客户端请求，并返回相应结果，以及操作数据库

### 类图设计



### 关键流程设计

当接收到客户端发送来的insert请求时，会进入到insert\_val()方法，首先调用TableTool中的connect()与数据库建立连接，之后调用TableTool中的insert（）向data表中插入数据，如果插入失败则进行更新操作调用tableTool中的update（），插入成功则返回True否则返回false

当接收到客户端发送来的check请求时，会进入到check\_val()方法，首先调用TableTool中的connect()与数据库建立连接，之后调用TableTool中的check（），向user表中查询请求过来的值与表中的值是否一致，如果id不存在，则返回id不存在，如果id存在，password不一致，则返回password错误，如果一致，则返回success

当接受到客户端发送来的register请求时，会进入到register\_val()方法，首先调用TableTool中的connect（）与数据库建立连接，之后调用TableTool中的register（）向user表中插入数据，插入失败则会返回用户已存在

当接收到客户端发送过来的select请求时，会进入到select（）方法，首先调用TableTool中的connect（）与数据库建立连接，之后调用TableTool中的select()向data表中查询数据，然后会以查询的结果中的视野阈值为依据，调用TableTool中的calculate（）方法，再次向data表中查询每个值占总体的比例，（////////////////////////////我记得网上有个聚合算法，可以加快全表查询的速度///////////////////////////），之后将结果返回