1、基于组件

无数据

多表数据

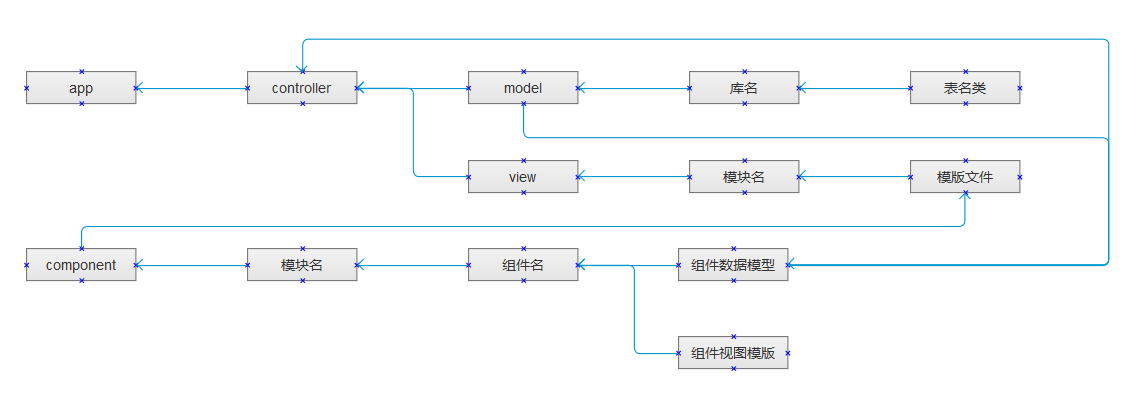
单表数据

组件

有数据

2、组件视图和数据接口兼容

e入口、m模块、c控制器



自动加载

1. 应用入口抽离到核心代码外部，只需要加载框架引导文件。而应用的基础常量则由框架引导文件定义
2. 组件部分属于通用层代码，但由于多入口、多库等特征，可能要求同一组件视图使用不同库的数据（保留变量，以根据变量加载对应的组件数据模型文件），并且视图是可能不需要数据的
3. 缓存类，分离动态缓存，并入模版类中；静态缓存和redis结合组成新的缓存类
4. 静态资源分离（缓存类不对静态资源类作动态解析及静态缓存）
5. 自动加载类文件

app->controller->模块名->控制类

resource->controller->index->静态资源方法类

model->read->数据查询类

model->write->数据变更类

component->模块名->组件名->组件数据模型类

z->core->核心类

z->core->model->核心模型类

z->lib->库类

app->public->公共类

app->extend->扩展类

appConMethodNameCls

resConMethodLongNameCls

modReadMethodNameCls

modWriteMethodNameCls

com组件名