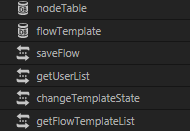
### 编辑端

**后端结构**

编辑端主要控制流程模板和节点模板。一个流程模板有多个节点模板，通过流程模板id进行对应关联，定义节点模板的意义在于客户端用户提交时可以根据节点模板生成节点实例。节点实例可以做用户行为跟踪/生成提交历史/版本查看。



**前端**

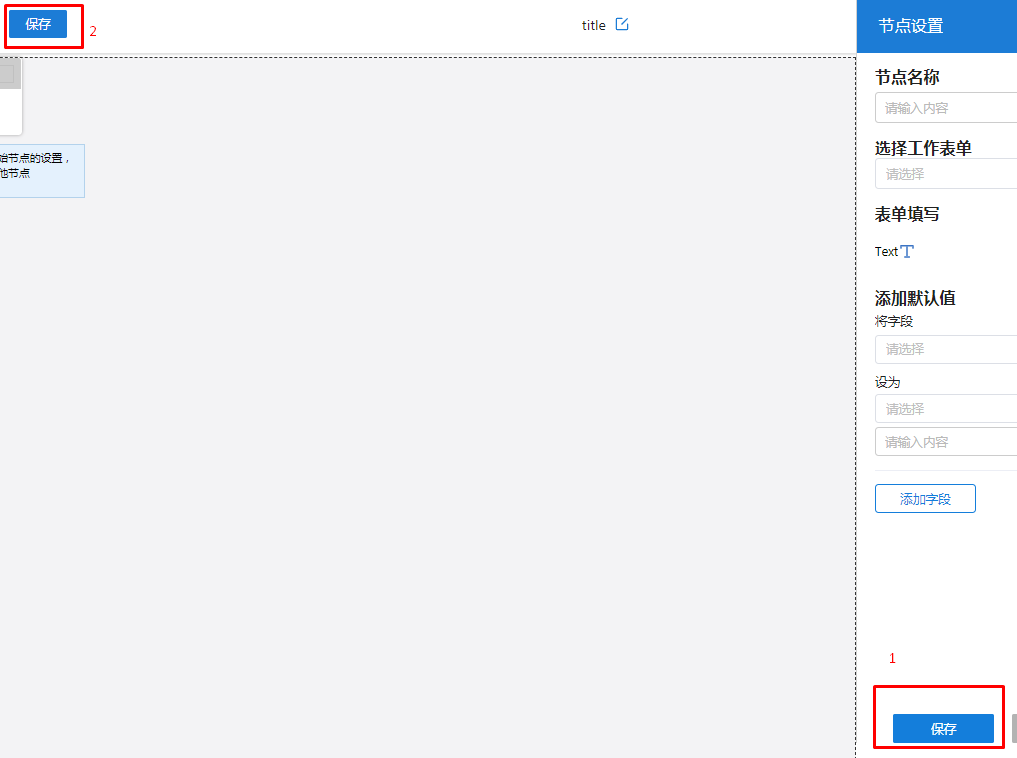
编辑页主要的交互部分是一个流程容器，用来显示和编辑流程结构。流程容器可以自定义节点的显示样式，和显示逻辑，用if组件判断显示相应样式。



流程容器有特殊的点击节点事件，可以获取当前在容器里所点击节点的相关信息，包括节点id、节点文本、节点类型、附带信息等。通过点击事件获取信息之后便可以进行针对节点的编辑工作。演示案例中编辑栏位于页面右侧，当点击节点时判断是否为业务节点（发起/审批/填写/盖章/if）。案例中的编辑权限操作是通过控制position（该节点所能操作的用户身份）和feildPermissions（控制对应表单的读写权限）变量来进行的，可以参考案例中的点击节点事件。

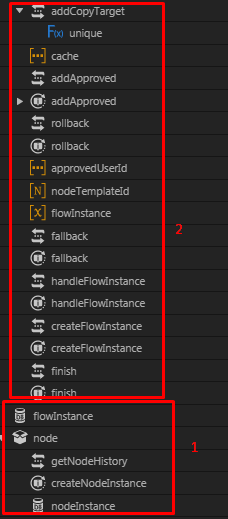


编辑好节点后点击保存，将刚才的配置存在当前的节点配置列表中，最后点击页面上方的保存便可以提交数据进数据库。通过共享服务，可以把模板列表分享给客户端拉取模板。



### 客户端

客户端数据表是对标模板的流程实例和节点实例，当创建新的流程/用户提交时会生成，之后的操作都基于其中的字段进行，下图1是数据表部分，2是基于其实现的其他功能，比如退回，用户提交，手动结束，拿回等。这些功能涉及到多表操作，因此采用可回滚的服务，用一般服务调用可回滚服务，前端再调用一般服务，这样可以避免判断条件过多造成的代码冗长。



这里需要注意flowInstance中有很多用于判断流程状态的字段：

depracated：用于存储分支选择时未选择的节点，比如当前可选的下一个执行节点有a和b，如果当前提交的用户选择了a那么将[b]加入该字段，在其他节点提交时可用于屏蔽b节点（即如果没有选择b，那么在别的节点提交时便不会再显示b节点）。这里便需要了解流程引擎的获取pending节点的逻辑，可以参考文档https://www.ivx.cn/docview?lang=zh&page=data-flow&type=doc&showall=0#；

pendingUser：存储当前执行人的userId；

approved：已经执行过的节点；

approvedUserId：已经执行过的节点对应执行人的userId；

copyTarget：该实例抄送人的userId；

根据业务不同可以选择不同的解决方案，比如设计其他的通用字段存储冗余，在选择节点/选择下一节点执行人时的操作和数据结构有关，这里自己在使用时要考虑自身需求以及性能。