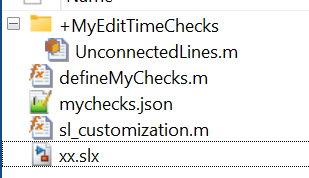
# 自定义编辑时检查

## 官网示例

<https://ww2.mathworks.cn/help/slcheck/ug/define-custom-edit-time-checks-that-fix-issues-in-architecture-models.html>

## 需要的文件



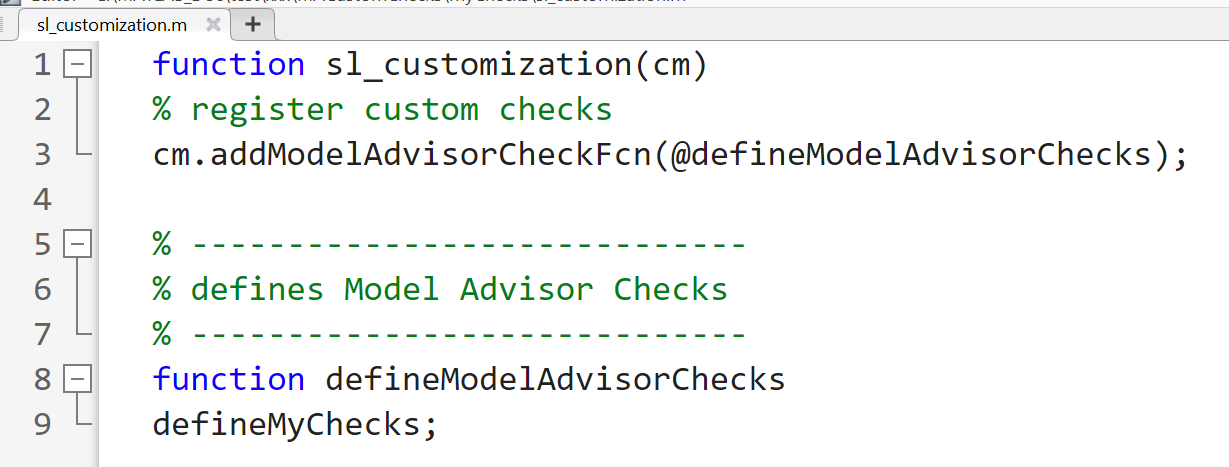
## sl\_customization.m

体现 Simulink 自定义更新的文件，必须放在搜索路径或者当前路径下。搜索路径上可以有多个同名的 sl\_customization.m 文件，执行对应的更新指令，例如 >> sl\_refresh\_customizations 时，MATLAB 会依次检索执行这些文件。此处我们需要更新 Model Advisor，所以稍后要在 MATLAB 中执行以下指令

>> Advisor.Manager.refresh\_customizations

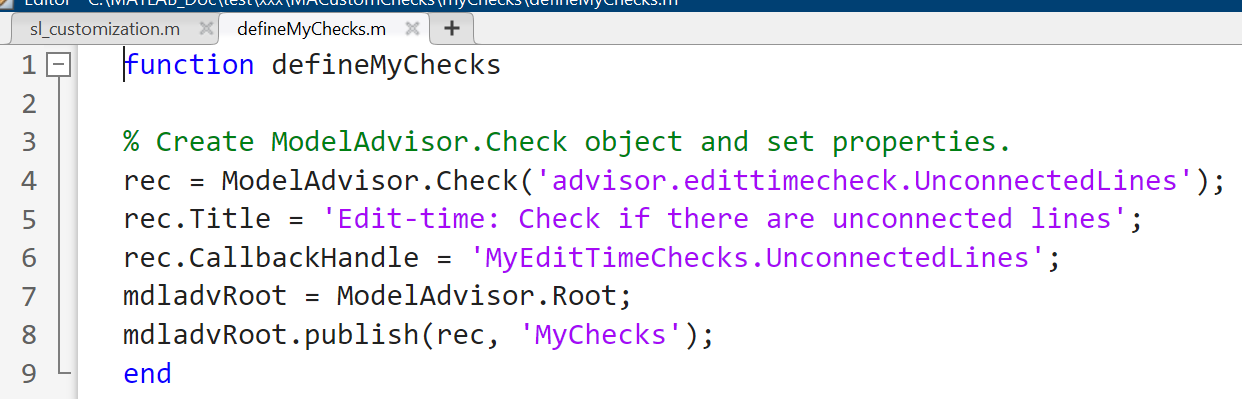
具体操作见下一节【加载自定义检查的操作步骤】。

这个文件的内容非常简单：



cm.addModelAdvisorCheckFcn 调用了 defineModelAdvisorChecks，又进一步调用了 defineMyChecks，这里的所有调用是为了将自定义检查加载到 Model Advisor。

### 加载自定义检查（defineMyChecks.m）



上一步最后调用的 defineMyChecks，其函数名可以自定义。这个函数注册了一个 ID 叫做 advisor.edittimecheck.UnconnectedLines 的 Model Advisor 检查并且将它发布到了 Model Advisor 的检查子集 MyChecks 中——稍后在Model Advisor 界面上可以看到这个新增的检查集。

其对应的回调函数 MyEditTimeChecks.UnconnectedLines 是一个描述了检查逻辑的类函数。

### 自定义编辑时检查算法（+MyEditTimeChecks/UnconnectedLines.m）

我们需要一个继承了 ModelAdvisor.EdittimeCheck 抽象类的子类，用以体现检查的算法逻辑。

这里 +MyEditTimeChecks/UnconnectedLines.m 即是体现检查算法的类文件。文件夹 +MyEditTimeChecks 名称中带有一个 + ，标志着它是一个 package 文件夹，可以包含一组函数/类文件——这类文件夹不需要添加到搜索路径，只要保证它的上一级在搜索路径上，或者当前工作路径是它的上一级路径，它就可以被 MATLAB 发现。使用包含在 package 中的函数，例如我希望调用 +myPackage/myFunction.m ，则在 MATLAB 中直接写 myPackage.myFunction() 即可，不需要带上加号。

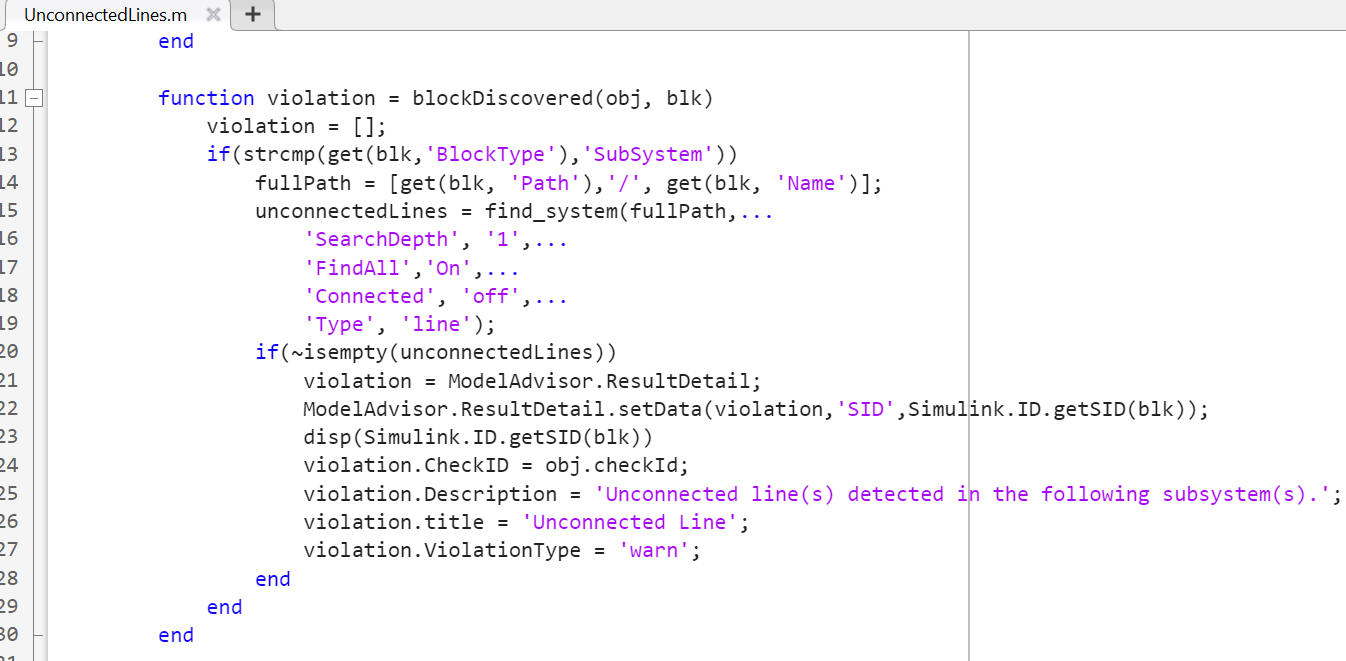
前一节中 rec.CallbackHandle = 'MyEditTimeChecks.UnconnectedLines' 指向的正是这个类函数。

描述该类的文档如下：

<https://ww2.mathworks.cn/help/slcheck/ref/modeladvisor.edittimecheck-class.html>

这个类具有 blockDiscovered、 finishedTraversal、 fix 三个方法，其中前两个方法中均可以承载检查的逻辑：

* blockDiscovered 方法会在检查进行过程中被循环调用，用以遍历模型中的每个模块，发现问题时可以返回一个包含警告信息的 violation （规则违反）对象（在编辑时检查激活时，会依据 violation 的 SID 字段将对应的模块高亮）
* finishedTraversal 方法将在遍历结束后被调用，用来做一些统计性质的工作（可以在 blockDiscovered 中先将模块筛选出来再统一检查），发现问题时返回 violation 对象（本例中没有涉及，可以参考官网示例）
* fix 方法中提供自动修复选项（本例中没有涉及，可以参考官网示例）



可以看到这里的逻辑是对每个模块进行遍历，发现子系统，即在此层级执中行 find\_system，若能找到未连接的线，则为此子系统生成一个 violation。

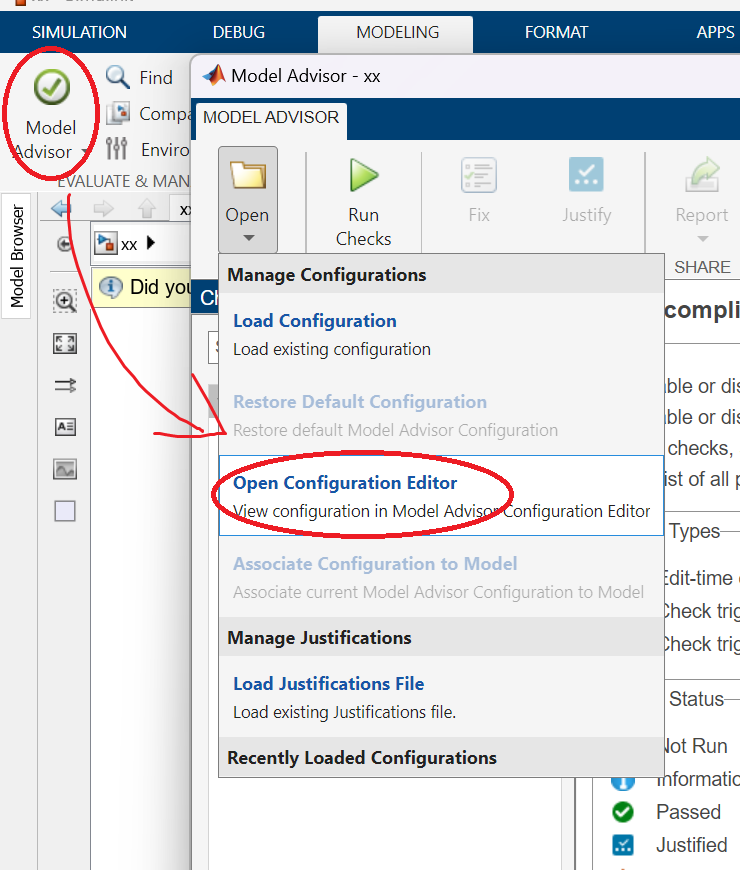
## 加载自定义检查的操作步骤

【1】刷新 Model Advisor缓存：

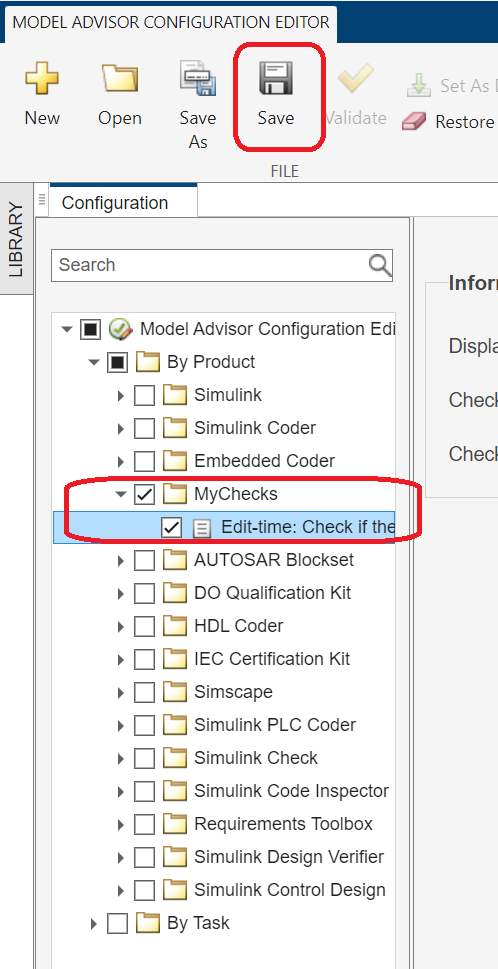
在MATLAB 中运行

>> Advisor.Manager.refresh\_customizations

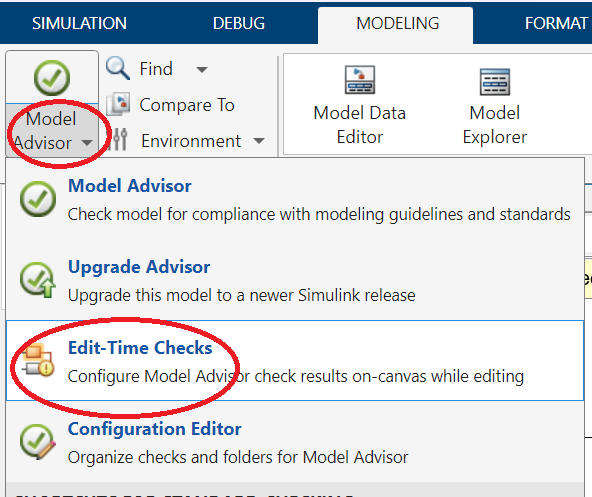
【2】打开 Model Advisor 配置：



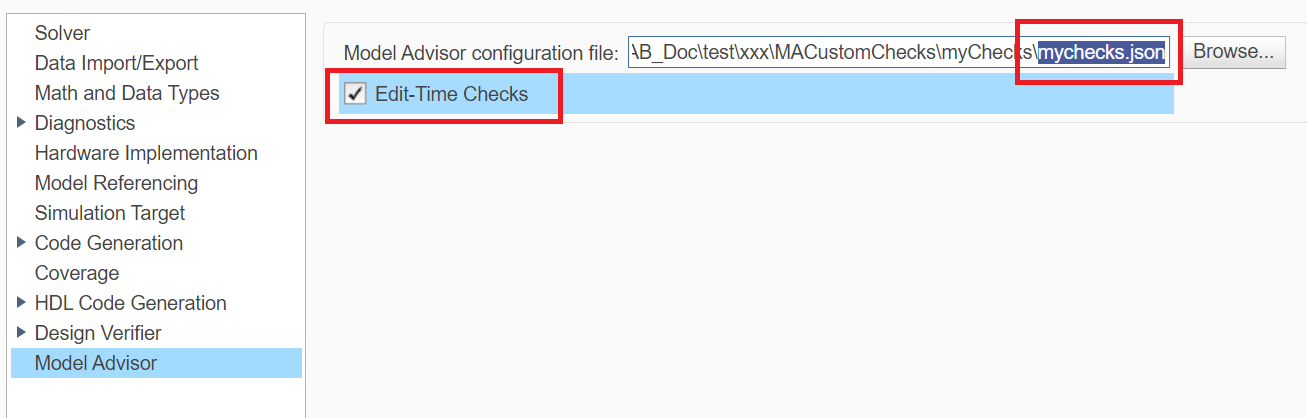
【3】勾选需要的检查项目，保存为一个 .json 文件



【4】启用编辑时检查



【5】配置中选择刚才保存的 .json 文件



【6】可以看到有问题的子系统被高亮了

