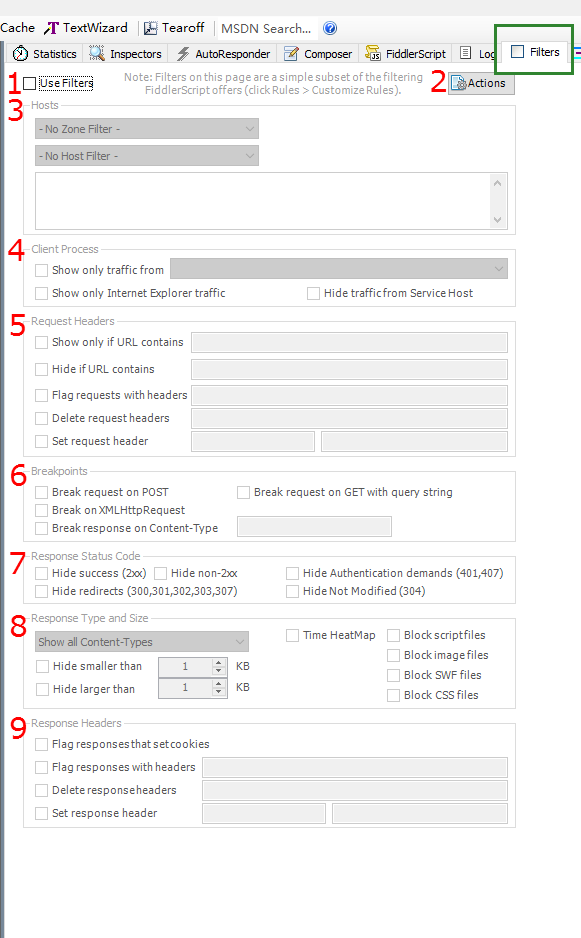
fiddler filters 使用（fiddler只显示指定请求，fiddler不显示指定请求，即filter请求过滤）

Fiddler 有一个filters可以很好的帮助我们只显示我们关系的请求或者隐藏掉我们不关心的请求。这在开发中是非常有用的，尤其是你在这边想看请求，那边有程序或者网站在自动频繁发送求情，真的很烦。回想一下买票插件，是不是会在不停的刷，你在fiddler看东西就会很烦。

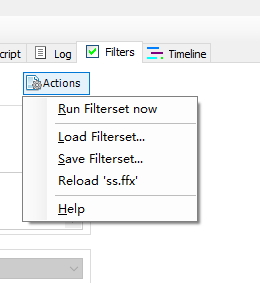
另外就是使用fiddler的时候我们几乎不会关心js和css请求，现在js越来越多的模块化了，在开发模式下往往会加载很多js文件（可能几百个），这往往都是我们不关系的请求。

如下图，先整体看看。

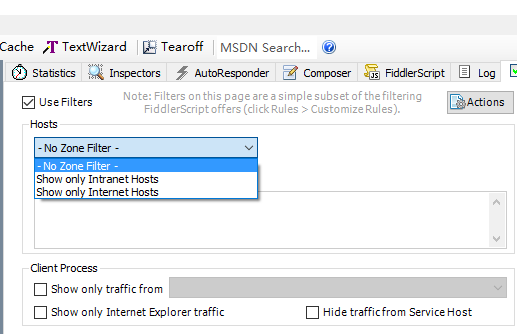


接下来是分步解释（按照途中标记的号码）：

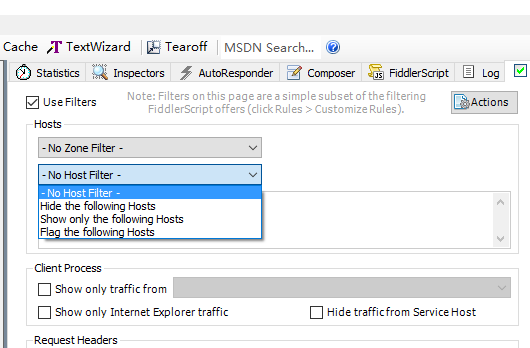
1. 是否启用filters。
2. 我们可以维护多套filters，在不同的情形下加载不同的filter，在Actions中可以把当前的filter保存起来，也可以加载以前保存的filter，如下图。



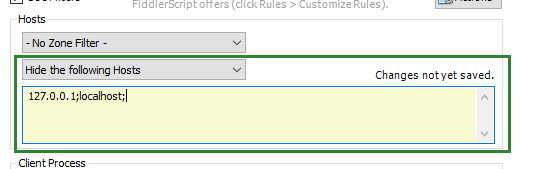
1. 根据主机名信息的filter。其中第一个下拉框是只显示内网或者外网选项，个人觉得用处很有限，如下图



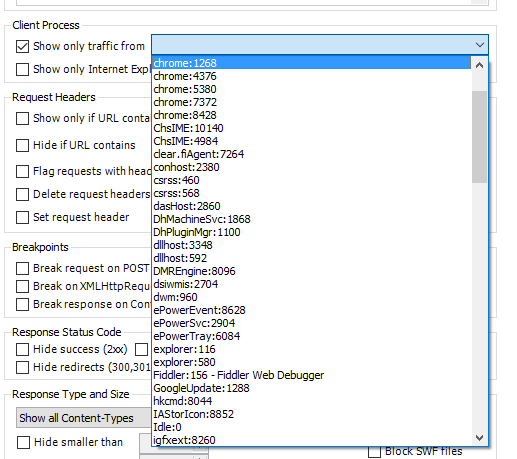
另一个是下拉框结合它下面的输入框来完成根据主机名信息显示或者隐藏或者标记指定请求，如下图。



下图是一个例子，可以简单理解为隐藏对本机的请求。

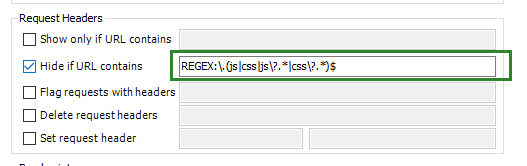


1. 根据进程信息的filter，第一个checkbox选上就可以选择进程了，那么fiddler将只显示该进程发出的请求，如下图。另外两个一个是指关心IE，另一个是隐藏来自service host的请求。

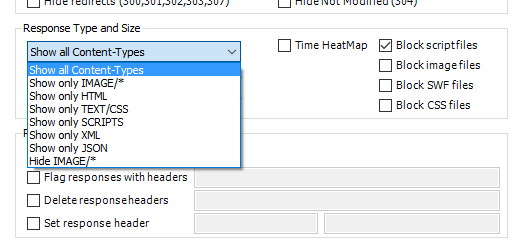


1. 根据request header的filter。这个可能是我觉得比较有用的一块儿了。

Show only if URL contains 和 Hide if URL contains 使用方法类似 功能相反。都是url中包含某些字符的filter，多种情况空格分隔。支持正则表达式和完全匹配，如下是个例子，可以简单理解为隐藏所有js和css请求。



1. 这个部分是各种各样的断点，大家应该知道fiddler是有断点功能的，可以在请求发送时断住，之后修改请求，再发送。当然也可以在响应的时候断住，做修改。而且fiddler还可以模拟响应，根本不发送给server端，在项目开发阶段，server端没有准备好API这个就特别有用了，或者用了复现一些问题。
2. 如果server端的响应代码属于或者不属于哪一类就隐藏，比较简单。
3. 这部分是根据响应类型和大小的filter，前一半是显示隐藏的filter，后block的filter。block的filter也挺有意思，如果你选中了block css那么浏览器就加载不到css了。



1. 是对response header的flag和修改。比较有用的是如果你想知道哪些请求修改了cookie那么就可以选中第一个checkbox。