**概述**

致敬WordPress Hooks，开发了JS版的钩子

用途：1、在预埋的钩子位置执行额外动作；2、在预埋的钩子位置执行过滤器

用法：脚本的 add\_action 和 add\_filter 与 WP 的类似，接收三个参数，分别是：钩子名称、钩子方法、优先级（默认10）

例如：

function render\_complete\_alert() {

    alert("Ajax不刷新加载完成！");

}

add\_action("ajax\_render\_complete",render\_complete\_alert,10);

**目前，提供以下脚本过滤器**

过滤器名：replace\_load\_blog\_options\_func

用途：替换页面加载时从后端请求基础配置的方法**【注：使用此过滤器强制设置示例中的配置后，主题后台的对应设置将会失效】**

参数：原始请求方法（并没什么用）

过滤器需返回：新的请求方法（接收1个参数：将数据resolve进行后续处理）

示例：

add\_filter("replace\_load\_blog\_options\_func",function(oldFunc) {

    return function(resolve) {

        // 首先Ajax请求获取配置，假设配置都获取好了，将配置resolve进行后续处理

        resolve({

            // 主题设置-是否启用不刷新加载页面

            enable\_pageLoader:true,

            // 主题设置-是否启用Ajax强制缓存

            ajax\_forceCache:true,

            // 当前访问页面的用户是否登录

            is\_user\_loggedin:true,

            // WP设置-评论是否强制要求输入昵称和邮箱

            cmt\_req\_name\_email:true,

            // 主题设置-需要强制输入昵称和邮箱时，弹出框的标题

            cmt\_req\_name\_email\_title:"\* 昵称与邮箱为必填项",

            // WP评论提交的PHP地址（不直接输出是为了防止被刷广告）

            cmt\_action\_url:"http://your\_wordpress\_url.com/wp-comments-post.php",

            // 主题设置-FAQ找不到答案时预设文本

            chat\_nodata:"找不到答案",

            // 主题设置-是否启用代码高亮

            enable\_highlightjs:true

        })

    }

});

过滤器名：get\_blog\_options

用途：得到主题配置后，对得到的配置进行过滤

参数：主题的配置对象

过滤器需返回：修改后的对象

示例：

add\_filter("get\_blog\_options",function(data) {

    // 无论后台怎么配置，都不允许启用Ajax加载页面

    data.enable\_pageLoader = false;

    return data;

})

过滤器名：replace\_query\_func

用途：替换搜索按钮的检索方法

参数：原始请求方法（并没什么用）

过滤器需返回：新的请求方法（接收2个参数：1将数据resolve进行后续处理、2待检索的query信息）

示例：

add\_filter("replace\_query\_func", function(oldFunc) {

    return function(resolve,query) {

        //打印查看query内容：s(关键字)、post\_type(查询的文章类型)、only\_title(仅包含标题)

        console.log(query);

        //根据query查询数据。假设已得到了搜索结果，将结果resolve进行后续处理

        resolve([{

            "thumbnail": "文章特色图url地址",

            "title": "文章标题",

            "href": "文章url地址",

            "date": "文章发布时间",

            "tags": [

            {

                "color": 1,

                "tag": "标签名称"

            }

            ],

            "like": "点赞数量",

            "comment": "评论数量"

        }]);

    }

})

过滤器名：get\_query\_data

用途：点击顶部搜索按钮，使用主题默认的Ajax成功请求到搜索结果后，对得到的数据进行过滤

参数：搜索结果JSON数组

过滤器需返回：修改后的搜索结果JSON数组

示例：

add\_filter("get\_query\_data", function(data) {

    var dataStr = JSON.stringify(data);

    // 把http:换成https:

    dataStr = dataStr.replace(/http:/ig,"https:");

    return JSON.parse(dataStr);

});

过滤器名：replace\_like\_func

用途：替换文章点赞方法

参数：原始请求方法（并没什么用）

过滤器需返回：新的请求方法（接收2个参数：1将数据resolve进行后续处理、2点赞成功后的点赞数量）

示例：

add\_filter("replace\_like\_func", function(oldFunc) {

    return function(resolve,postID) {

        // 根据 postID 向后端发送点赞文章的ID，将返回的点赞数量resove进行后续处理

        resolve('9999');

    }

})

动作名称：ajax\_render\_complete

用途：页面不刷新加载完成后执行

参数：无

示例：

add\_action("ajax\_render\_complete",function() {

    console.log('加载完成');

})