//关于Firefox的技巧的文档：侧边栏视图、缩略图视图，by江3如此多娇；本文更新地址，

//相关须知
//外观
//自动完成-AutoComplete：
//自动完成 - 操作符：
//网络关键字 - Internet Keywords：
//域名猜测 - Domain Guessing：

//扩展：Better URL Bar、Enter Selects、Location Bar Enhancer、
//Browser.urlbar.default.behavior，
//Browser.urlbar.restrict.typed，
//wikipedia.org/wiki/Favicon，
//wikipedia.org/wiki/Address_bar，

//牛B的火狐智能索引，
//FF搜索引擎管理 - 江3，你一般需要同时了解地址栏和搜索栏
//Firefox 参数 - DNS - 江3，地址栏解析时也涉及到DNS解析步骤

//火狐地址栏解析网址的3步骤：AutoComplete->语法更正-> Internet Keywords->Domain Guessing；
//第三、四步太浪费时间，建议都禁用，把地址栏还原为最原始的本地URL请求工具，
//关闭后2步，智能地址栏的工作就不再多次访问网络了，一次Enter/Ctrl+Enter只访问一次网络；别担心，智能地址栏的前2步依然很智能，
//比如，输入www。baidu。com，会自动更正语法为http://www.baidu.com

//外观：

```
//user_pref("browser.urlbar.formatting.enabled",true);//高亮域名？如www.baidu.com；默认true；参考；
user_pref("browser.urlbar.trimURLs",false);//精简地址栏ULR？默认true，取消顶级域名后的(/)号，若URL的协议为http://，且子域名不是ftp，且不包含@时，隐藏http://；参考；
//如https://www.example.org/变为https://www.example.org；http://www.example.org/foobar 变为www.example.org/foobar，但http://ftp.example.org/foobar不变.
~~//user_pref("browser.urlbar.hideGoButton",false);//隐藏地址栏的->按钮？该参数在FF3+中已不需要；参考；~~
```

```
//user_pref("browser.urlbar.clickSelectsAll",true);//单击全选地址栏文字？默认true；参考；
//user_pref("browser.urlbar.doubleClickSelectsAll",true);//双击全选地址栏文字？默认false；参考；
```

//自动完成-AutoComplete：
//何谓Awesomebar？
//如何自定义自动完成功能？
//就是指地址栏智能补全网站，并列出下拉网址列表，自动完成优先级最高，不需要使用远程服务器处理，仅靠浏览器搞定

```
user_pref("browser.urlbar.autocomplete.enabled",true);//开启智能地址栏/Awesomebar？优先级最高，false时连同Inline autocomplete一并禁用；参考，
user_pref("browser.urlbar.autoFill",true);//开启Inline autocomplete/地址栏自动补全？FF14+默认/建议true，参考，
//user_pref("browser.urlbar.default.behavior",2);//智能地址栏筛选策略？如果像我这样禁用历史记录，可以改为2；参考，
//FF3.5+默认0，该十进制参数值可视为6位BitMap算法，下面基数可随意相加，比如3表示访问过的书签，17表示访问过的匹配的URL，49=1+16+32表示访问过的匹配的输入过的URL
```

0	1	2	4	8	16	32	64	128
默认/书签+历史	历史	书签	标签Tags	标题	URL	Typed	JavaScript	打开的标签Tabs

```
//user_pref("browser.urlbar.filter.javascript",true);//地址栏筛选时，排除书签脚本-即javascript:URIs？默认true；参考，
user_pref("browser.urlbar.maxRichResults", 27);//地址栏建议数？默认值12，建议27（在使用AwesomeBar Popup样式时 ）
//user_pref("browser.urlbar.delay", 0);//"自动补全"的延迟时间0，参考，
```

//自动完成 - 操作符：

```
user_pref("browser.urlbar.match.title","!");//标题，标题当然用感叹号表示重要性了！！！
user_pref("browser.urlbar.match.url","@");//URL，@表示路径，好记！
user_pref("browser.urlbar.restrict.tag","#");//标签Tag ，
user_pref("browser.urlbar.restrict.typed","$");//输入，linux输入符$，相当于光标
user_pref("browser.urlbar.restrict.openpage","%");//打开的标签页，O|O|O形似并列的标签
user_pref("browser.urlbar.restrict.bookmark","0");//书签 ，0 和 00 不用区分中英，应分配给使用频率最高的书签和历史
user_pref("browser.urlbar.restrict.history","00");// 历史，同上
```

//网络关键字 - Internet Keywords：

//即类似Chrome的Omnibox的地址栏搜索，应该禁用
user_pref("keyword.enabled",false);//开启地址栏关键字搜索 参考: Keyword.enabled，
~~//user_pref("keyword.URL","http://www.google.com/search?q=%s");//地址栏默认搜索引擎地址？，参考，~~
~~//空值表示使用搜索栏默认搜索引擎，用关键词搜索，前提keyword.enabled为true，~~
//FF23+该参数失效，Mozilla取消自定义网络关键字的目的是为了让用户更好的了解搜索引擎关键字，Bug738818，可以通过扩展 keyword.URL Hack!恢复该参数效果。

//YY走：http://duckduckgo.com/?kp=-1&kn=1&kf=1&ks=s&kw=w&kt=n&ke=1&k1=-1&kv=n&q=
//Google：https://203.208.46.200/search?nfpr=1&filter=0&pws=0&hl=en&safe=off&q=
//大部分Mozilla产品默认值：http://www.google.com/search?ie=UTF-8&oe=utf-8&q=
//FF和SB是：http://www.google.com/search?btnI=I%27m+Feeling+Lucky&ie=UTF-8&oe=UTF-8&q=
//FF2默认值改为了Google的Browse by Name服务：http://www.google.com/search?ie=UTF-8&oe=UTF-8&sourceid=navclient&gfns=1&q=

//网络关键字与搜索引擎关键字不是一个概念，虽然都是用远程搜索引擎处理的，但是网络关键字特制地址栏的默认搜索引擎，
//地址栏的处理过程是十分复杂的：如果用户输入的网站URL，则直接打开，但这是很少的情况，大部分时候用户输入的是ip、域名、文件路径等，
//这时候FF地址栏会使用一系列解析器把这几种不完整的URL转为完成的URL，再打开；
//但上面的的方法打不该网站（解析不到/链接不到），就把地址栏输入的网址发送给网络关键字代表的搜索引擎服务器，

//如果输入的是多个空格隔开的部分，则直接由网络关键字接管并搜索
//但事实上，绝大部分的域名请求失败都会被关键字截获并猜测，即使只输入一个英文单词的域名，比如baidu，也不会进行下面的域名猜测
//所以，除非我们输入形如www.baidu.com或者输入baidu后Ctrl回车，否则，基本所有的地址栏输入都会被网络关键字解析，
//所以，为了下面的域名猜测更好的借款形如www.*.com类型的单单词域名网址的打开，我们应该关闭网络关键字，

//那你肯定会问"这样的话用地址栏搜索不是要多输入一个搜索引擎关键字么？"
//的确，不过这并不麻烦，而且所有搜索引擎都用关键字，不是很公平呢

//域名猜测 - Domain Guessing:
//FF1.3之后域名猜测的UI被取消，用下面3个参数控制，bug，
user_pref("browser.fixup.alternate.enabled",**false**);//自动对找不到的网址添加www和com？参考，
//user_pref("browser.fixup.alternate.prefix","www.");
//user_pref("browser.fixup.alternate.suffix",".com");

//为什么叫"域名猜测"-Domain Guessing，而不是其他的("autocomplete","hostname completion","domain searching")？
//因为autocomplete是根据用户之前的输入或保存的书签主动显示的一些列最有可能符合用户意向的网址，而域名猜测是在网站请求失败时才触发的，
//hostname completion - 域名补充，往往是和DNS解析相关的，与域名猜测的也原理不同，故不能用

//如果开启了关键字(keyword.enabled为true)，它会截获绝大部分域名猜测，
//理论上说，一个www.*.com 式的网站在输入中间部分后应该由域名猜测正确进入，但并非如此，
//事实上，基本上只有形如 www.baidu 和 baidu.com 的输入才会被域名猜测并打开，
//如果有上面的网络关键字存在，域名猜测就形同虚设了，所以应该把网络关键字关闭，

//而且，域名猜测仍然是发生在DNS解析失败的输入进行补全，所以还是会浪费时间，建议直接禁用，养成自己Ctrl+Enter补全前后缀的习惯，
//自己Ctrl+Enter的好处是打开速度更快，且我们如果一个网址输入错误，我们不会等很长时间才知道

域名猜测的不足共有如下**6**点：

- 慢：域名解析需要重新补全前后缀再进行DNS解析
- 命中率低：域名解析不仅受关键字影响，还手输入的域名的URL影响，总之到底什么时候会命中完全不能预测
- 错误提示问题：如搜索hostname，若解析出错正常的错误提示应该是hostname无法解析/链接失败，
- 但是用了域名猜测后，变成了www.hostname.com有问题，这很可能让你一塌糊涂，以为是不是输入时拼写有问题啊
  DNS解析流问题，每次猜测都等于一次DNS解析的尝试，增加了额外的开销，还会给DNS服务器的缓存中加入一些原本就不存在的域名，具体比较复杂
- 安全隐患：如果猜测错误，可能把一些敏感信息通过URL发送出去，某些网站收到你的请求可以随意使用你的信息，你也没辙
- 不能通过代理服务器进行猜测