《dynaTrace Ajax 教程 – 基础篇》

大灰狼系列分享

BAIDU

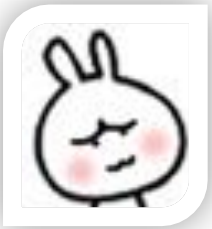
2010-4-19

作者:大灰狼

《dynaTrace Ajax 教程 – 基础篇》

大灰狼系列分享

什么是 dynaTrace Ajax



“dynaTrace Ajax 是一个详细的底层追踪工具，它不仅可以显示所有请求和文件在网络中传输的时间，还会记录浏览器 Render、CPU 消耗、JavaScript 解析和运行情况等详细的信息，而这些也只是 dynaTrace Ajax 的冰山一角。”

引用面向对象 JavaScript 的作者：Stoyan Stefanov

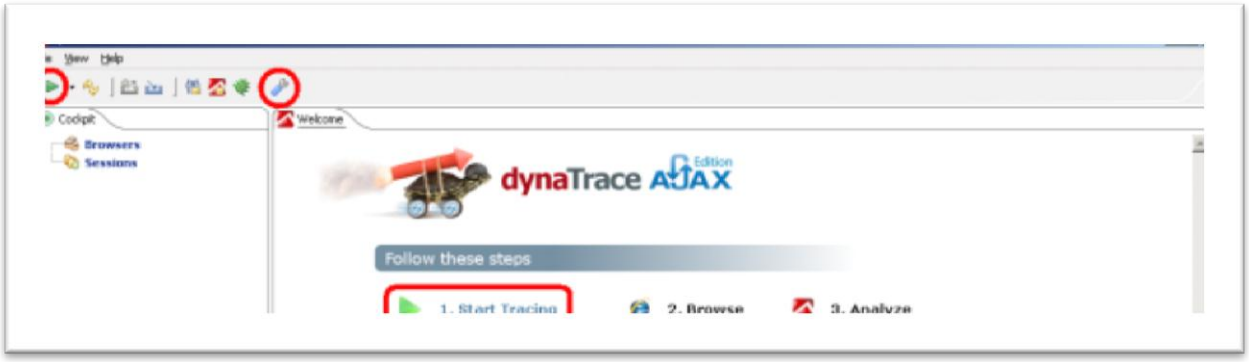
为什么需要 dynaTrace Ajax

jQuery、GWT、YUI、DoJo 等兴起的框架让构建 Web 2.0 应用变得更加容易，糟糕的是，随之而来的，定位这些应用的问题也越来越难，尤其是与性能相关的时候。

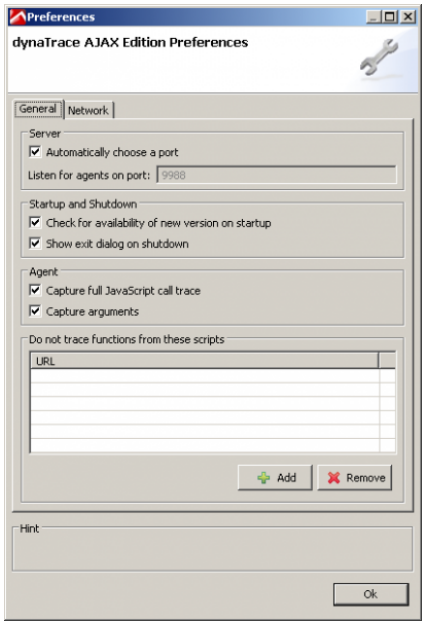
最让人头疼的浏览器依然是 Internet Explorer – 企业级环境也一样。垃圾的 IE 诊断工具让开发和测试人员都十分头痛。Web 2.0/Ajax 应用的挑战不仅是理解网络交互（有多少或何时一些资源被下载），我们也要理解这些效果的性能。问题将扩展到 JavaScript、XmlHttpRequests 对象、DOM 操纵、框架、布局和绘制。dynaTrace Ajax 的出现就是为了解决以上的所有问题，它致力于帮助人们了解到在现代的 Ajax 应用中哪些问题导致了性能瓶颈。

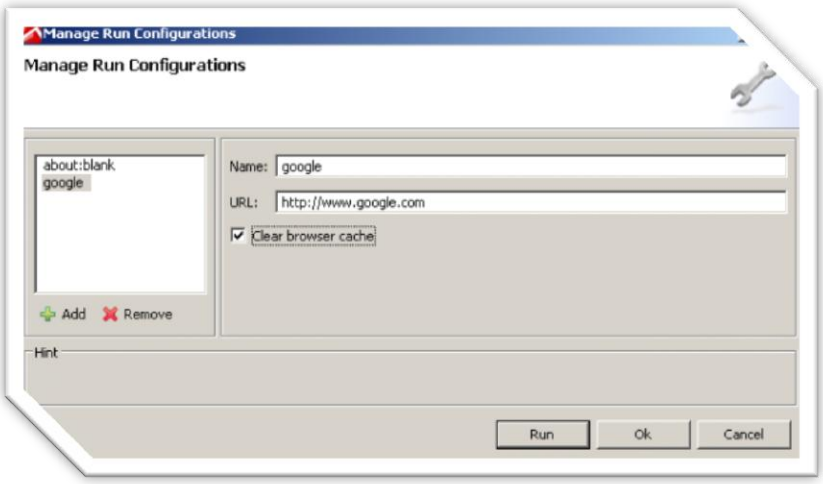
开始追踪

安装好 dynaTrace 后，我们会进入到下面的主界面。



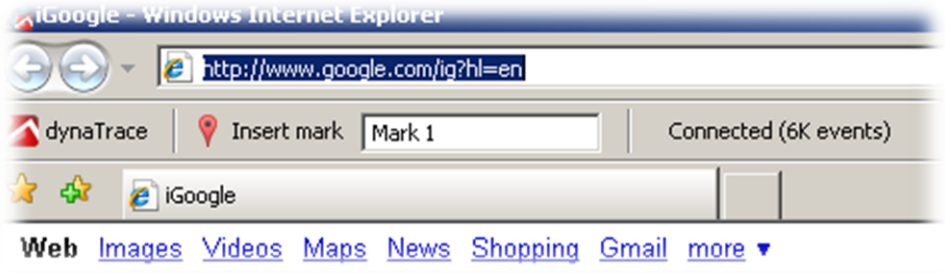
dynaTrace 使用所谓的运行设置（Run Configurations）来直接浏览你要追踪的网站，避免每次重复的输入网址。你可以通过左上角（图中所示）的按钮随时管理运行设置（添加、修改或删除）。在下拉列表中选择 Manage Run Configurations 将进入下图所示的页面。





接下来我们将以追踪 Google 为例，在 Name 中填入本次追踪的名称，在 URL 中输入要追踪的站点的 URL，Clear broswer cache 复选框用来设置是否清空缓存，当你想在追踪时忽略缓存的影响时可以将其选中，设置好后点击 Run 运行。

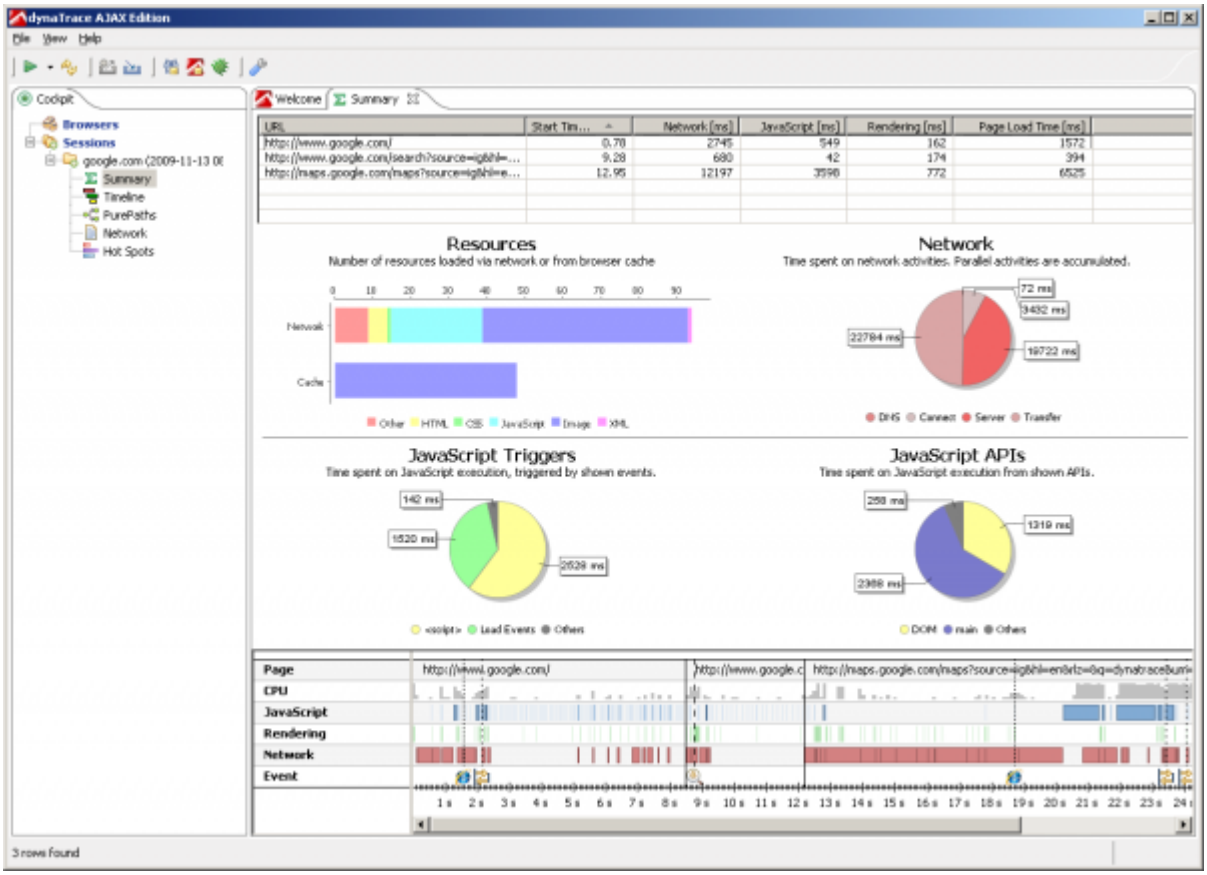
在 dynaAjax 工具栏中最右边的按钮可以用来打开首选项（Preferences）用以进行更多的设置。



说明

因环境差异每次追踪文件都会不同，为了保证本教程内所有内容一致行，请将本教程附带的 step_by_step_google.dtas 文件通过 Import Session 导入，如下图：

运行追踪后，我们会发现浏览器上出现 dynaTrace 工具栏，上面的 Connected 用以表示当前正出于追踪状态，此时你可以浏览所有要检测的网页，所有数据都会被记录，浏览好后关闭浏览器，会自动生成追踪文件。双击左边的 google.com 将打开 Summary（总结），这里显示了大部分我们需要的信息，如下图所示。



在 Summary 中记录了所有访问的网页的详细信息，点击上面不同的 URL 将会看到相关的中间的图表和下面的时间线。在图中所示的 URL 中，我们可以知道以下一些事情：

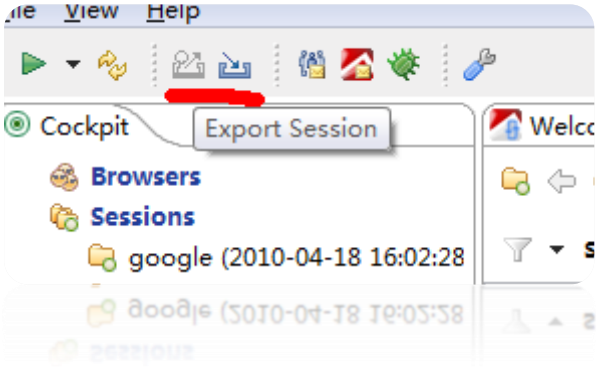
- **加载使用了多久**：Page Load Time[ms]列在 onLoad 时间被捕获前的总时间。
- **网络请求消耗了多长时间**：Network[ms]列显示了网络请求的时间，在下面的图表中我们可以清晰的看到所有网络活动分别的时间（DNS、连接、服务器处理、数据发送）。
- 有**多少**或是哪种类型的**资源**被下载，并且哪些资源是从缓存中直接获得的。
- JavaScript 一共**运行了多久**，每个事件分别在合适被触发，JS 调用不同 API 的占比等。这在确定问题的性能瓶颈出自哪里时非常有用。
- **呈现(Rendering)**使用的时间。浏览器需要计算布局并将页面呈现到屏幕。呈现的快慢依赖于 HTML、样式，而动态的 DOM 操纵可能消耗很多时间来重绘布局。Rendering[ms]列将会显示页面在这些活动中消耗了多长时间。
- **精确的页面生命周期**：在底部的时间线中的图表告诉我们：在一个页面的生命周期内，什么时间文件被下载、JavaScript 被执行、页面在呈现、CPU 资源消耗了多少，事件发生。

在我们的作为例子的追踪文件和 Summary 中，我们可以发现下面的事情：

- maps.google.com 的 Page Load 时间是 6.5s，在这个时间内浏览器下载了所有相关的文件，并初始化 HTML 和所有引用的对象，之后才会触发 OnLoad 时间。

- 在网络中消耗了 12s，我们可以从饼状图中看到 50%以上的时间用在了网络传输（这能证明我现在的网速很慢），42%的时间停留在服务器端（就是到服务器刚开始响应的时间），而 8%的时间用来创建和 Web 服务器间的物理连接。
- JavaScript 也是那个页面的主角，总共消耗了 3.6s。在 JavaSscript 饼形图中我们可以知道：2.1s 用在了脚本加载上，1.3s 用在 OnLoad 事件和鼠标点击句柄的处理。
- 在时间线（Timeline）我们还可以看到页面发送了两个 XmlHttpRequest 请求（通过时间线上的小标志）。

导入和导出



我们可以将追踪内容导出成追踪文件，当然，也可以将追踪文件导入到 dynaTrace 进行解析。

Bosn 的总结

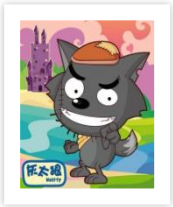
本文对 dynaTrace Ajax 的功能进行了简单介绍，对于基础篇，我们着重讲解常用和基本的功能，这些功能对于测试人员来说可能已经足够。如果想了解更深层的使用方法，敬请关注即将推出的 dynaTrace Ajax 进阶篇。谢谢！

如果文中有误，欢迎批评指正。

Hi: bosnma

E-mail: bosnma@live.cn

感谢您阅读大灰狼系列分享文档。



参考文献：http://blog.dynatrace.com/2009/11/17/a-step-by-step-guide-to-dynatrace-ajax-edition-available-today-for-public-download/

THE END