简介

Kibana是一个使用 Apache 开源协议，基于浏览器的 Elasticsearch 分析和搜索仪表板。Kibana 非常容易安装和使用。整个项目都是用 HTML 和 Javascript 写的，所以 Kibana 不需要任何服务器端组件，一个纯文本发布服务器就够了。Kibana 和 Elasticsearch 一样，力争成为极易上手，但同样灵活而强大的软件。

注释

本书原始内容来源Elasticsearch 官方指南 Kibana 部分，并对 panel 部分加以截图注释。在有时间的前提下，将会添加更多关于 kibana 源码解析和第三方 panel 的介绍。

欢迎捐赠，作者支付宝账号：*rao.chenlin@gmail.com*

10 分钟入门

Kibana 对实时数据分析来说是特别适合的工具。本节内容首先让你快速入门，了解 Kibana 所能做的大部分事情。如果你还没下载 Kibana，点击右侧链接：下载 Kibana。我们建议你在开始本教程之前，先部署好一个干净的 elasticsearch 进程。

到本节结束，你就会：

- 导入一些数据
- 尝试简单的仪表板
- 搜索你的数据
- 配置 Kibana 只显示你的新索引而不是全部索引

我们假设你已经：

- 在自己电脑上安装好了 Elasticsearch
- 在自己电脑上搭建好了网站服务器，并把 Kibana 发行包解压到了发布目录里
- 对 UNIX 命令行有一点了解，使用过 `curl`

导入数据

我们将使用莎士比亚全集作为我们的示例数据。要更好的使用 Kibana，你需要为自己的新索引应用一个映射集(mapping)。我们用下面这个映射集创建"莎士比亚全集"索引。实际数据的字段比这要多，但是我们只需要指定下面这些字段的映射就可以了。注意到我们设置了对 speaker 和 play_name 不分析。原因会在稍后讲明。

在终端运行下面命令：

```
curl -XPUT http://localhost:9200/shakespeare -d '
{
 "mappings" : {
  "_default_" : {
   "properties" : {
    "speaker" : {"type": "string", "index" : "not_analyzed" },
    "play_name" : {"type": "string", "index" : "not_analyzed" },
    "line_id" : { "type" : "integer" },
    "speech_number" : { "type" : "integer" }
   }
  }
 }
}
';
```

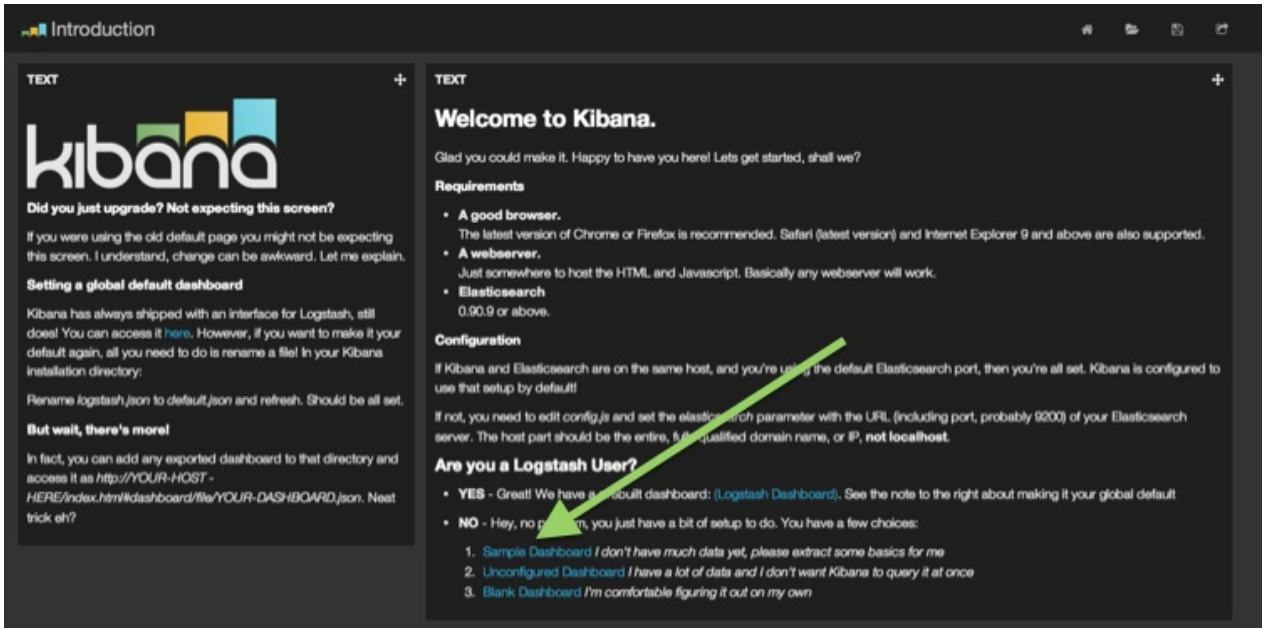
很棒，我们这就创建好了索引。现在需要做的时导入数据。莎士比亚全集的内容我们已经整理成了 elasticsearch 批量导入所需要的格式，你可以通过shakeseare.json下载。

用如下命令导入数据到你本地的 elasticsearch 进程中。这可能需要一点时间，莎士比亚可是著作等身的大文豪！

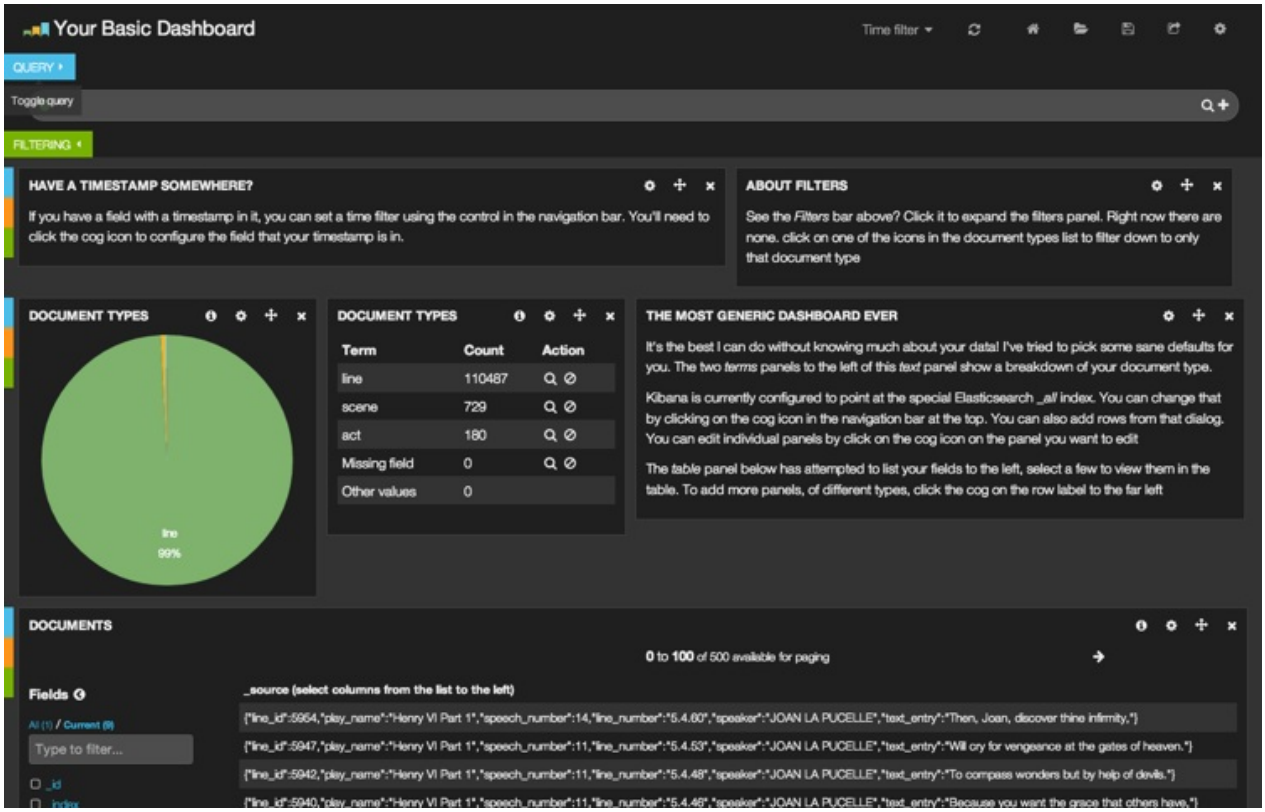
```
curl -XPUT localhost:9200/_bulk --data-binary @shakespeare.json
```

访问 Kibana 界面

现在你数据在手，可以干点什么了。打开浏览器，访问已经发布了 Kibana 的本地服务器。



如果你解压路径无误(译者注：使用 github 源码的读者记住发布目录应该是 `kibana/src/` 里面)，你已经就可以看到上面这个可爱的欢迎页面。点击 Sample Dashboard 链接

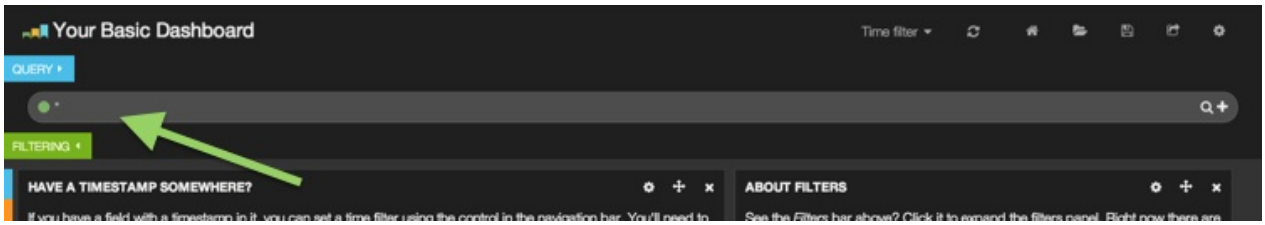


好了，现在显示的就是你的 sample dashboard！如果你是用新的 elasticsearch 进程开始本教程的，你会看到一个百分比占比很重的饼图。这里显示的是你的索引中，文档类型的情况。如你所见，99% 都是 lines，只有少量的 acts 和scenes。

再下面，你会看到一长段 JSON 格式的莎士比亚诗文。

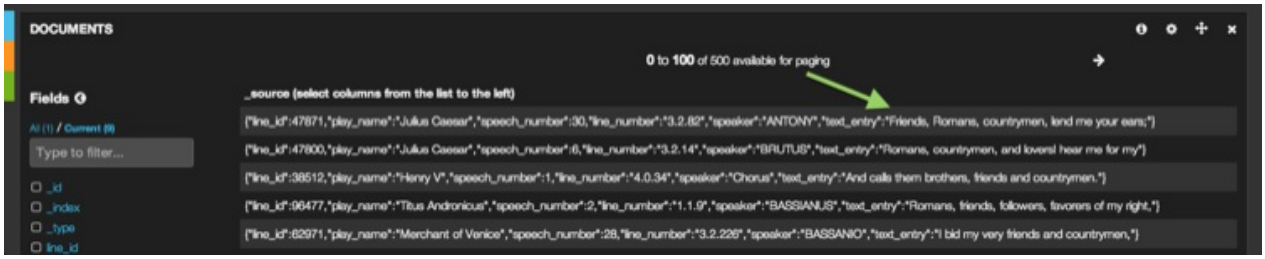
第一次搜索

Kibana 允许使用者采用 Lucene Query String 语法搜索 Elasticsearch 中的数据。请求可以在页面顶部的请求輸入框中书写。



在请求框中輸入如下内容。然后查看表格中的前几行内容。

```
friends, romans, countrymen
```

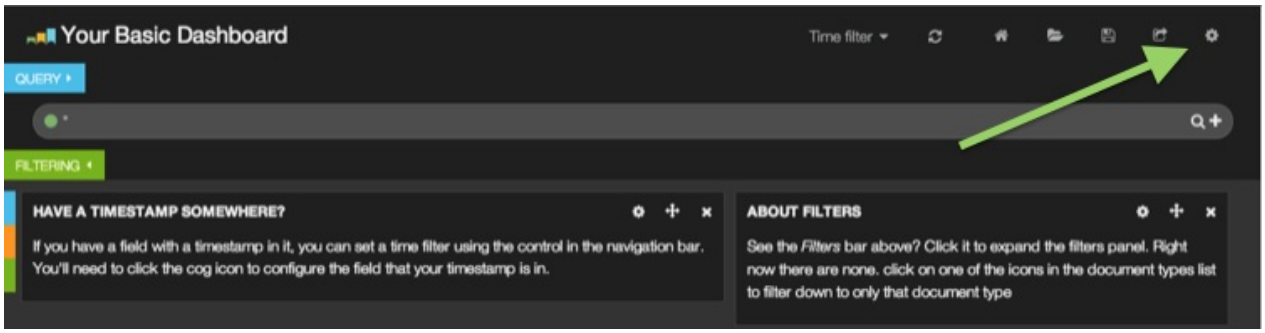


关于搜索请求的语法，请阅读 Queries and Filters。

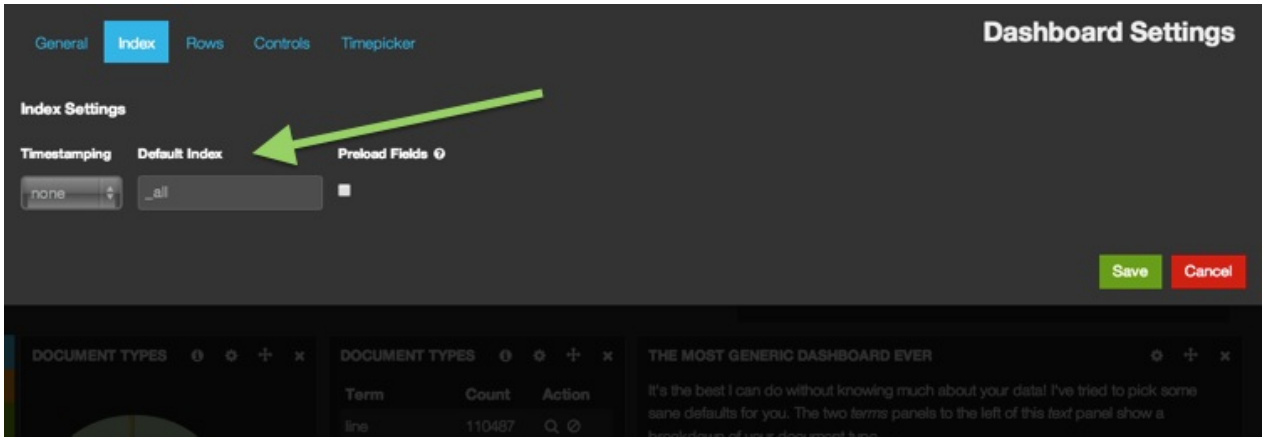
配置另一个索引

目前 Kibana 指向的是 Elasticsearch 一个特殊的索引叫 `_all`。 `_all` 可以理解为全部索引的大集合。目前你只有一个索引，`shakespeare`，但未来你会有更多其他方面的索引，你肯定不希望 Kibana 在你只想搜《麦克白》里心爱的句子的时候还要搜索全部内容。

配置索引，点击右上角的配置按钮：



在这里，你可以设置你的索引为 `shakespeare`，这样 Kibana 就只会搜索 `shakespeare` 索引的内容了。



下一步

恭喜你，你已经学会了安装和配置 Kibana，算是正式下水了！下一步，打开我们的视频和其他教程学习更高级的技能吧。现在，你可以尝试在一个空白仪表板上添加自己的面板。这方面的内容，请阅读 Rows and Panels。

请求和过滤

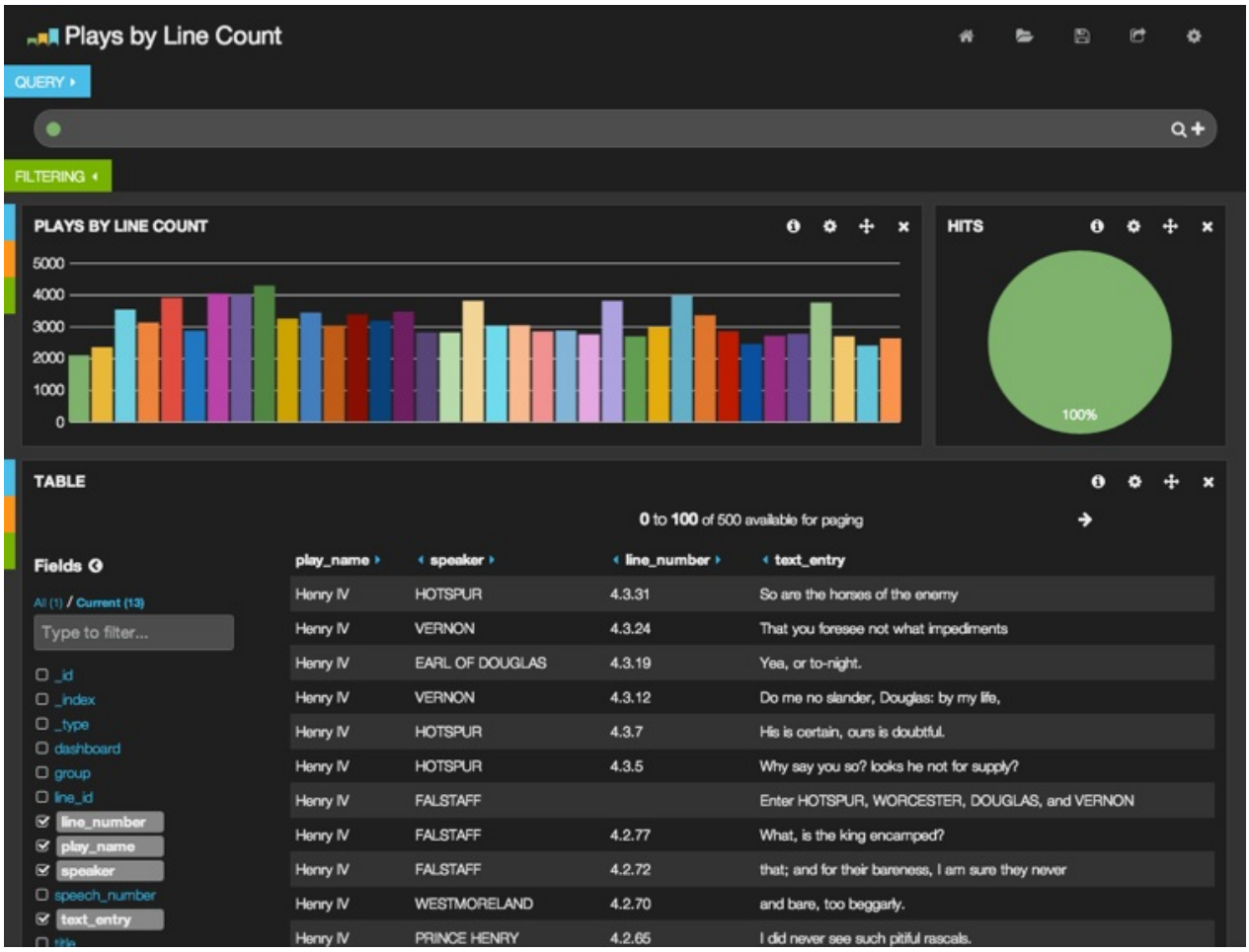
图啊，表啊，地图啊，Kibana 有好多种图表，我们怎么控制显示在这些图表上的数据呢？这就是请求和过滤起作用的地方。Kibana 是基于 Elasticsearch 的，所以支持强大的 Lucene Query String 语法，同样还能用上 Elasticsearch 的过滤器能力。

我们假设你已经：

- 在自己电脑上安装好了 Elasticsearch
- 在自己电脑上搭建好了网站服务器，并把 Kibana 发行包解压到了发布目录里
- 读过 Using Kibana for the first time 并且按照文章内容准备好了存有莎士比亚文集的索引

我们的仪表板

我们的仪表板像下面这样，可以搜索莎士比亚文集的内容。如果你喜欢本章截图的这种仪表板样式，你可以下载导出的仪表板纲要(dashboard schema)



请求

在搜索栏输入下面这个非常简单的请求

```
to be or not to be
```

你会注意到，表格里第一条就是你期望的《哈姆雷特》。不过下一行却是《第十二夜》的安德鲁爵士，这里可没有"to be"，也没有"not to be"。事实上，这里匹配上的是 `to OR be OR or OR not OR to OR be` 。

我们需要这么搜索(译者注：即加双引号)来匹配整个短语：

```
"to be or not to be"
```

或者指明在某个特定的字段里搜索：

```
line_id:86169
```

我们可以用 AND/OR 来组合复杂的搜索，注意这两个单词必须大写：

```
food AND love
```

还有括号：

```
("played upon" OR "every man") AND stage
```

数值类型的数据可以直接搜索范围：

```
line_id:[30000 TO 80000] AND havoc
```

最后，当然是搜索所有：

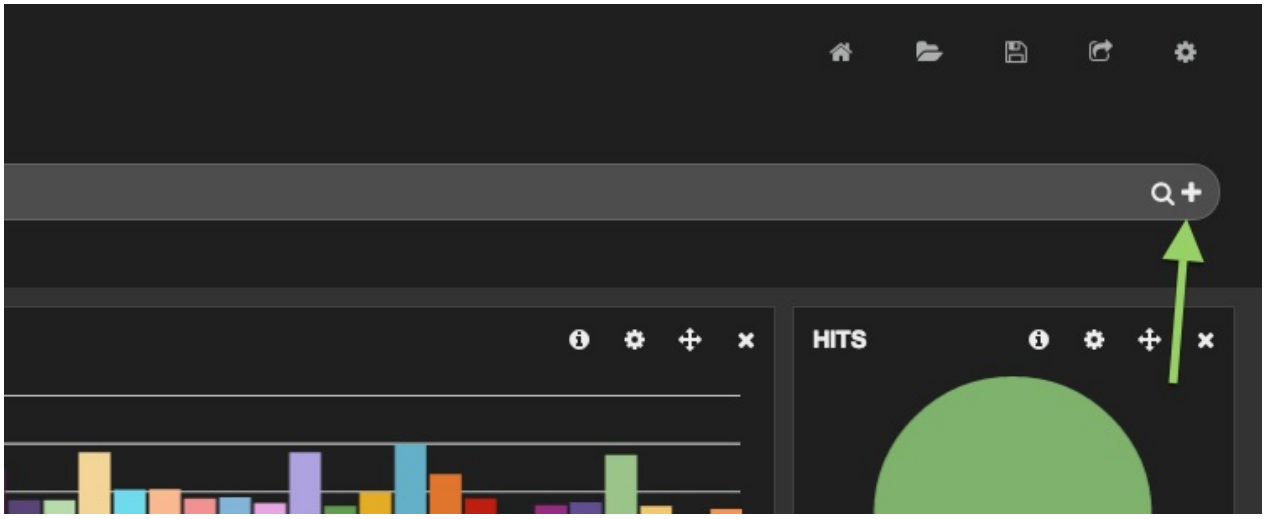
```
*
```

多个请求

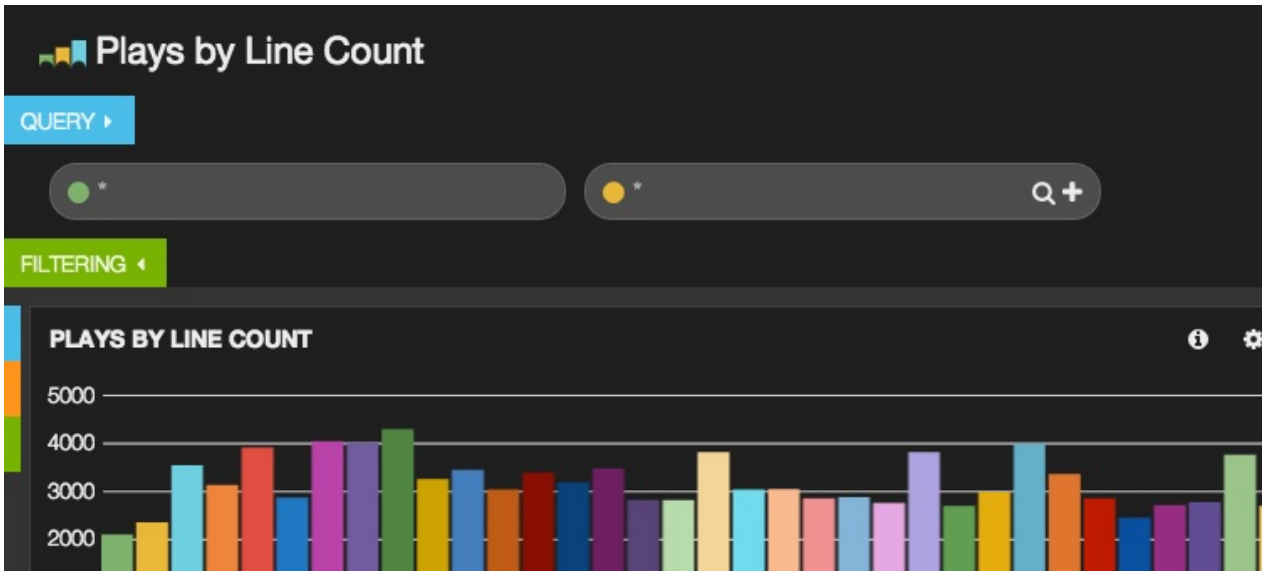
有些场景，你可能想要比对两个不同请求的结果。Kibana 可以通过 OR 的方式把多个请求连接起来，然后分别进行可视化处理。

添加请求

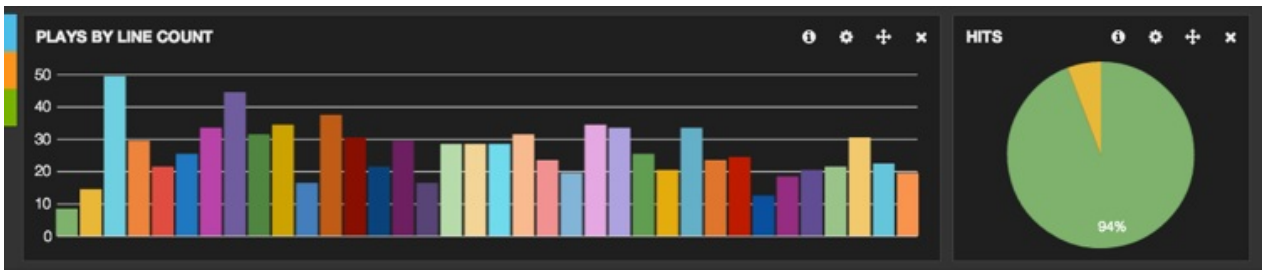
点击请求输入框右侧的 + 号，即可添加一个新的请求框。



点击完成后你应该看到的是这样子

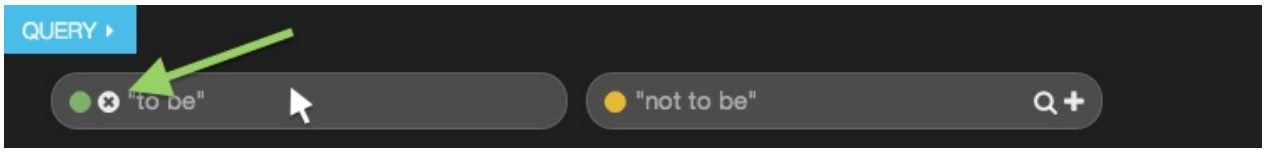


在左边，绿色输入框，输入 `"to be"` 然后右边，黄色输入框，输入 `"not to be"`。这就会搜索每个包含有 `"to be"` 或者 `"not to be"` 内容的文档，然后显示在我们的 hits 饼图上。我们可以看到原先一个大大的绿色圆形变成下面这样：



移除请求

要移除一个请求，移动鼠标到这个请求输入框上，然后会出现一个 x 小图标，点击小图标即可：

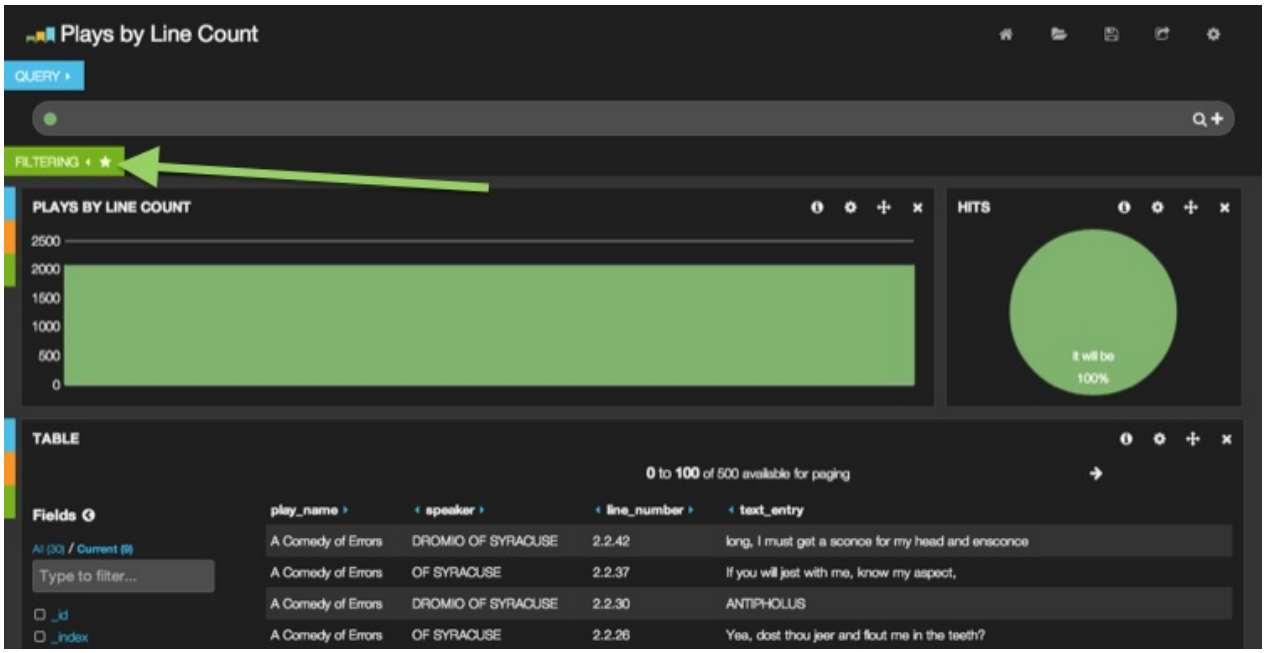

QUERY
"to be"
"not to be"

颜色和图例

Kibana 会自动给你的请求分配一个可用的颜色，不过你也可以手动设置颜色。点击请求框左侧的彩色圆点，就可以弹出请求设置下拉框。这里面可以修改请求的颜色，或者设置为这个请求设置一个新的图例文字：

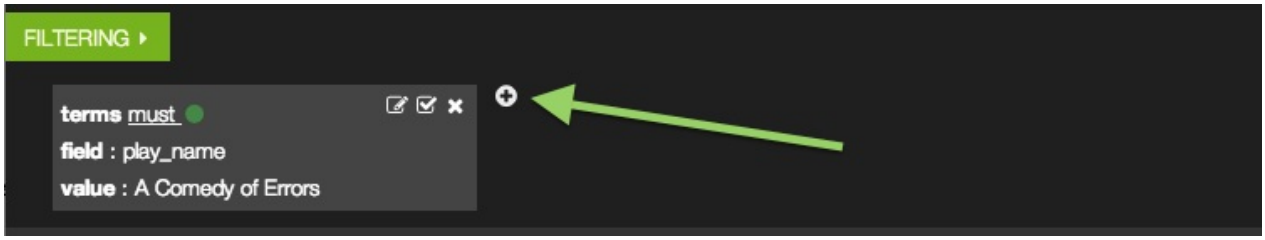
过滤

很多 Kibana 图表都是交互式的，可以用来过滤你的数据视图。比如，点击你图表上的第一个条带，你会看到一些变动。整个图变成了一个大大的绿色条带。这是因为点击的时候，就添加了一个过滤规则，要求匹配 `play_name` 字段里的单词。



你要问了“在哪里过滤了”？

答案就藏在过滤(FILTERING)标签上出现的白色小星星里。点击这个标签，你会发现在 *filtering* 面板里已经添加了一个过滤规则。在 *filtering* 面板里，可以添加，编辑，固定，删除任意过滤规则。很多面板都支持添加过滤规则，包括表格(table)，直方图(histogram)，地图(map)等等。



过滤规则也可以自己点击 + 号手动添加。

更多阅读

你现在已经可以处理过滤和请求了，你可能很好奇在 Kibana schema 里，他们是怎么存在的。如果你还想知道如何通过 URL 参数来添加请求和过滤，欢迎阅读 Templated and Scripted Dashboards

行和面板

Kibana 的仪表板是由行和面板组成的。这些都可以随意的添加，删除和重组。

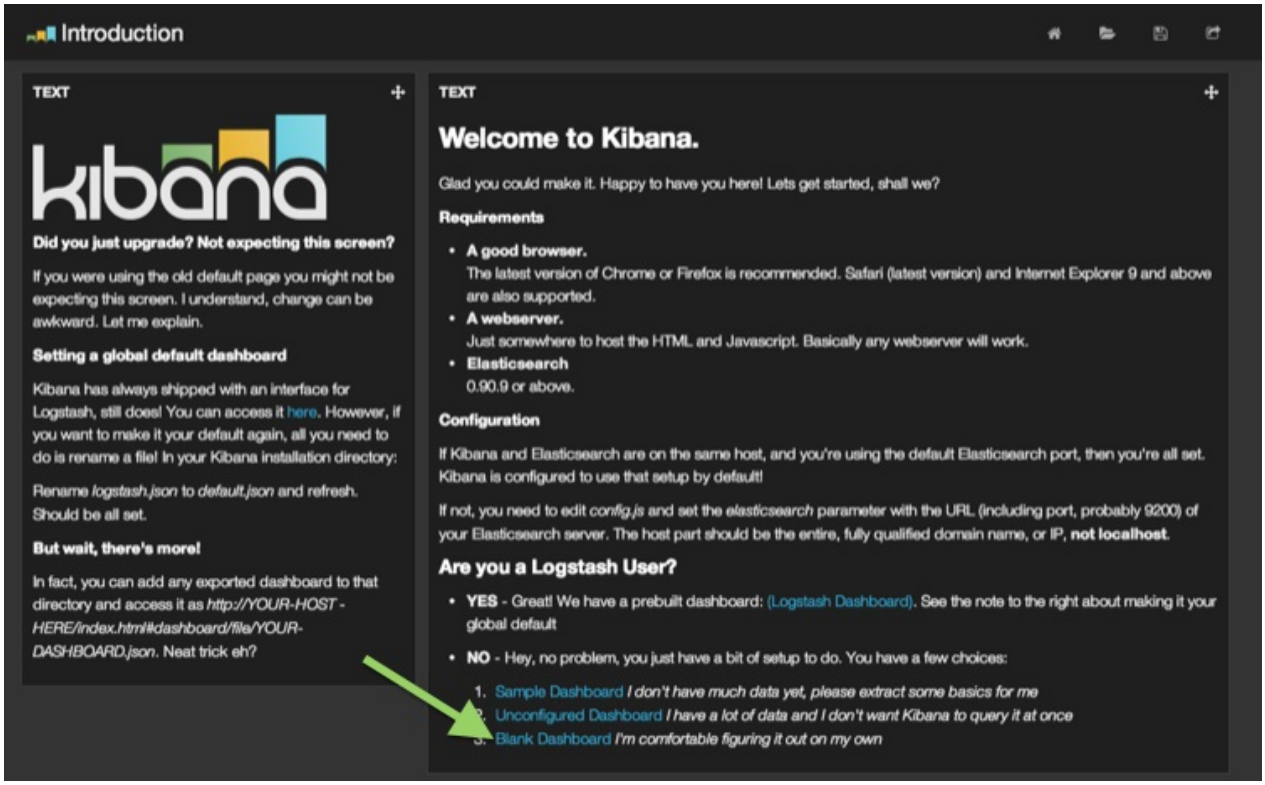
这节我们会介绍：

- 加载一个空白仪表板
- 添加，隐藏行，以及修改行高
- 添加面板和修改面板宽度
- 删除面板和行

我们假设你已经：

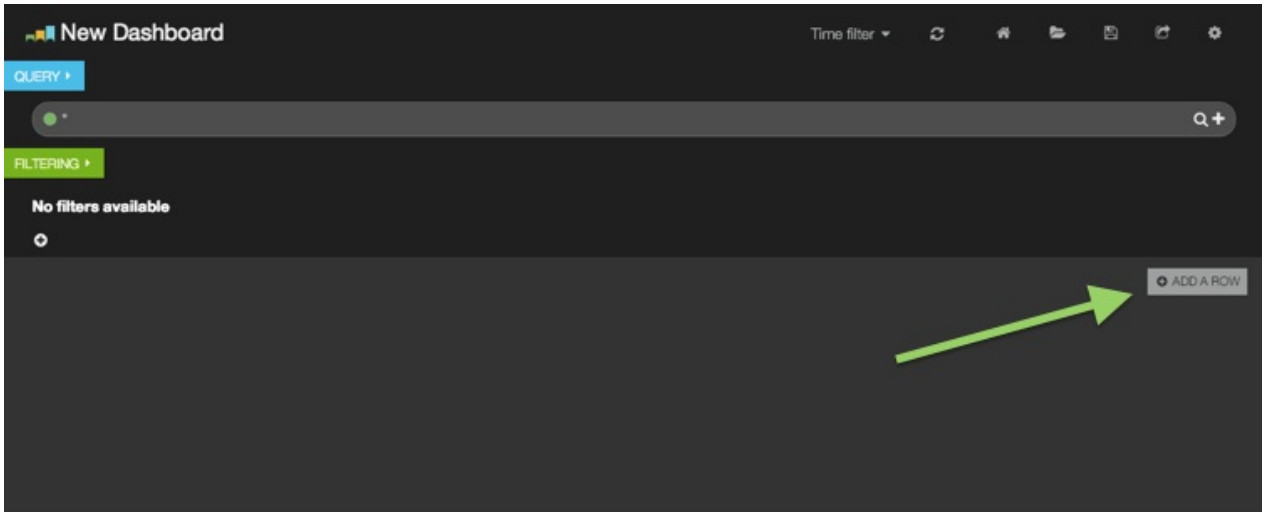
- 在自己电脑上安装好了 Elasticsearch
- 在自己电脑上搭建好了网站服务器，并把 Kibana 发行包解压到了发布目录里
- 读过 Using Kibana for the first time 并且按照文章内容准备好了存有莎士比亚文集的索引

加载一个空白仪表板

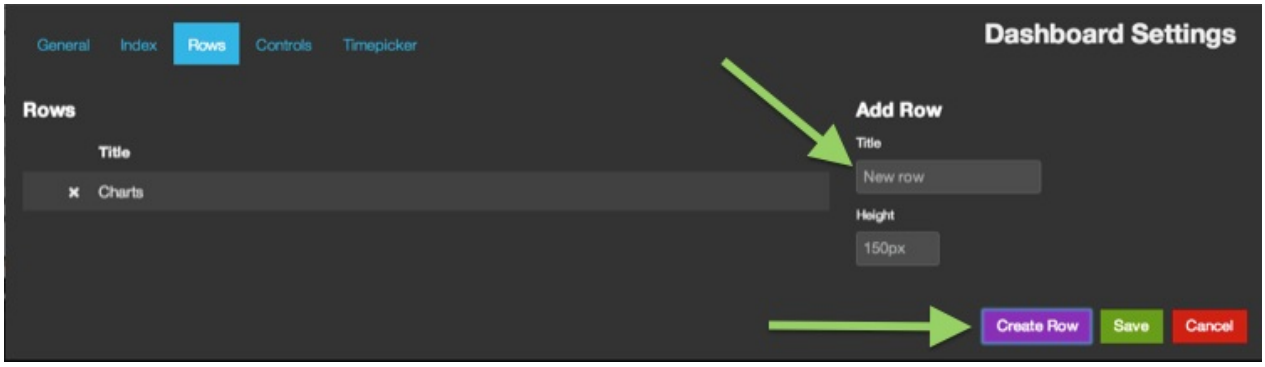


从主屏里选择第三项，就会加载一个空白仪表板(Blank Dashboard)。默认情况下，空白仪表板会搜索 Elasticsearch 的 `_all` 索引，也就是你的全部索引。要指定搜索某个索引的，阅读 Using Kibana for the first time。

添加一行



你的新空白仪表板上只有展开的请求和过滤区域，页面顶栏上有个时间过滤选择器，除此以外什么都没有。在右下方，点击添加行(ADD A ROW)按钮，添加你的第一行。



给你的行取个名字，然后点击创建(Create Row)按钮。你会看到你的新行出现在左侧的行列表里。点击保存(Save)

行的控制

现在你有了一行，你会注意到仪表板上多了点新元素。主要是左侧多出来的三个小小的不同颜色的长方形。移动鼠标到它们上面

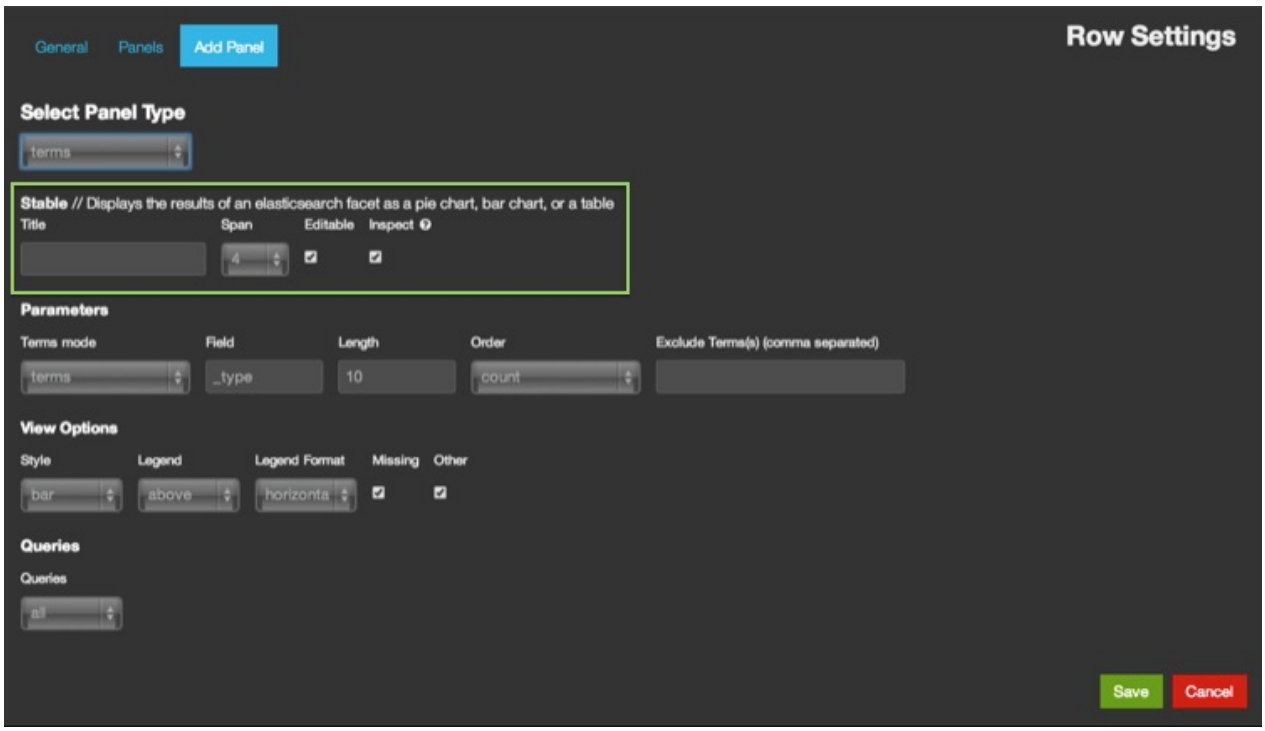
哈哈！看到了吧，这三个按钮是让你做这三件事情的：

- 折叠行(蓝色)
- 配置行(橘色)
- 添加面板(绿色)

添加面板

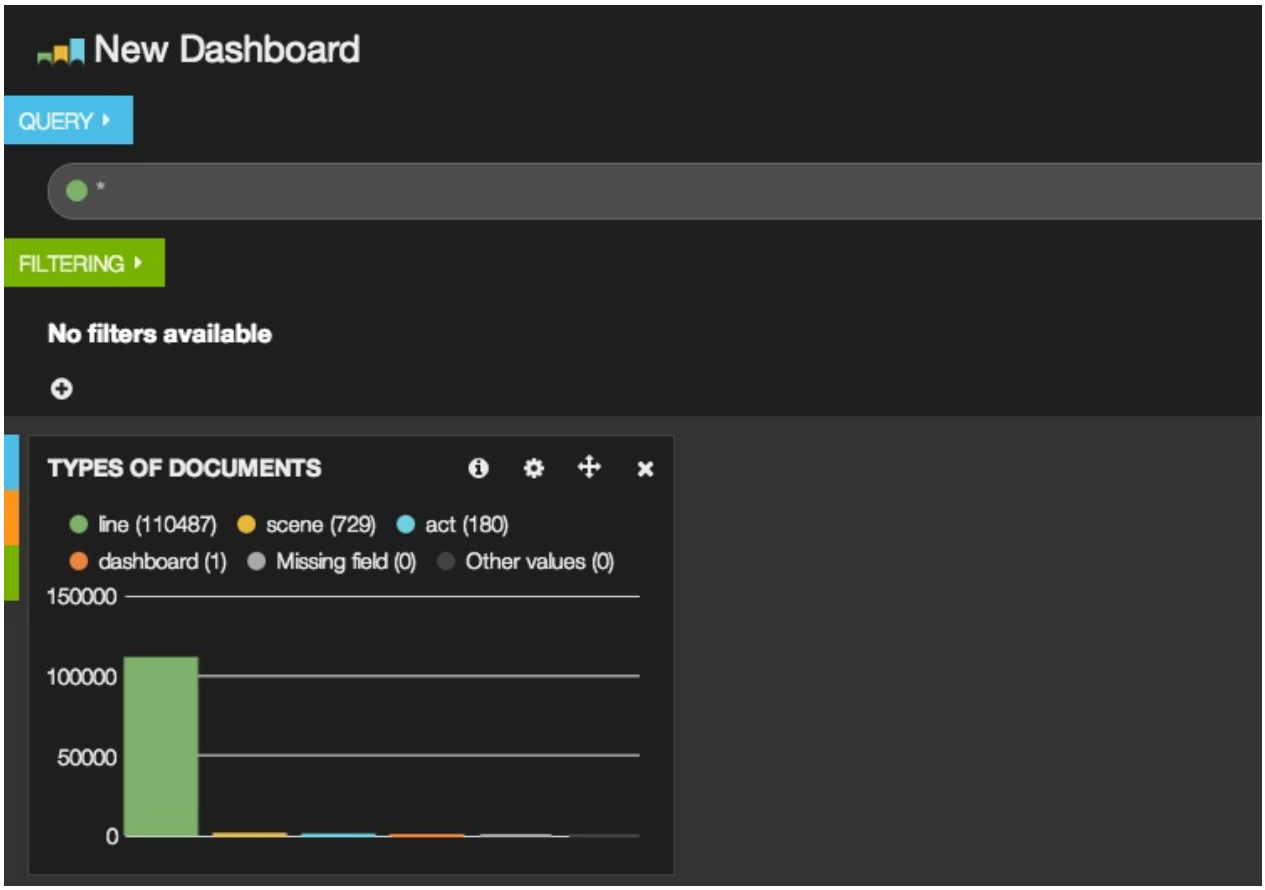
现在我们专注在行控制力的绿色按钮上，试试点击它。你也可以点击空白行内的灰色按钮(Add panel to empty row)，不过它是灰色的啊，有啥意思……

让我们来添加一个 terms 面板。terms 面板可以让我们用上 Elasticsearch 的 terms facet 功能，查找一个字段内最经常出现的几个值。



你可以看到，terms 面板有一系列可配置选选，不过我们现在先只管第一段里德通用配置好了：

1. Title: 面板的名称
2. Span: 面板的宽度。Kibana 仪表板等分成 12 个 *spans* 面板最大就是到 12 个 spans 宽。但是行可以容纳超过 12 个 spans 的总宽度，因为它会自动把新的面板放到下面显示。现在我们先设置为 4。
3. Editable: 面板是否在之后可以继续被编辑。现在先略过。
4. Inspectable: 面板是否允许用户查看所用的请求内容。现在先略过。
5. 点击 **Save** 添加你的新 **terms** 面板到你的仪表板

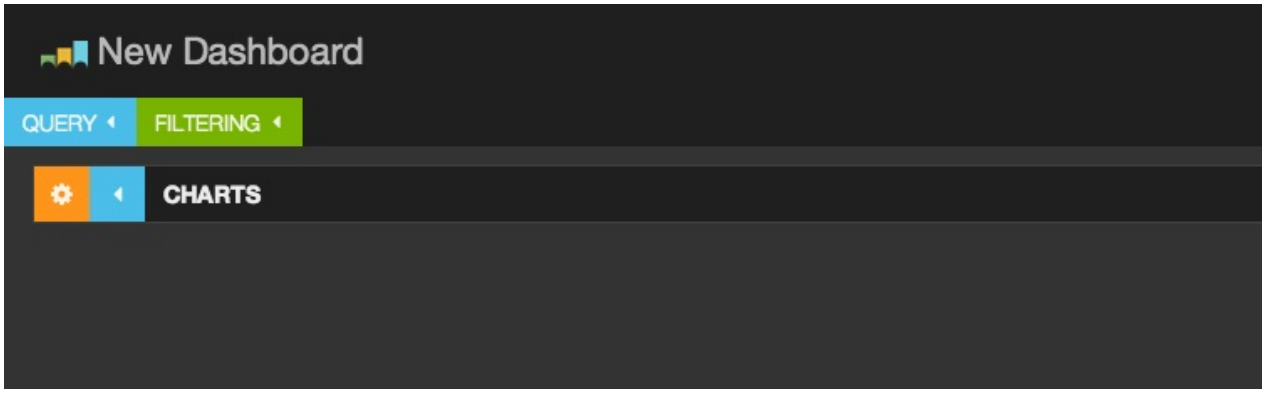


太棒了！你现在有一个面板了！你可能意识到这个数据跟 Using Kibana for the first time 中的饼图数据一样。 `shakespeare` 数据集集中在 lines，还有少量的 acts 和 scenes。

折叠和展开行

蓝色按钮可以折叠你的行。被折叠行里的面板不会刷新数据，也就不要求 Elasticsearch 资源。所以折叠行可以用于那些你不需要经常看的数据。有需要的时候点击蓝色按钮展开就可以了。

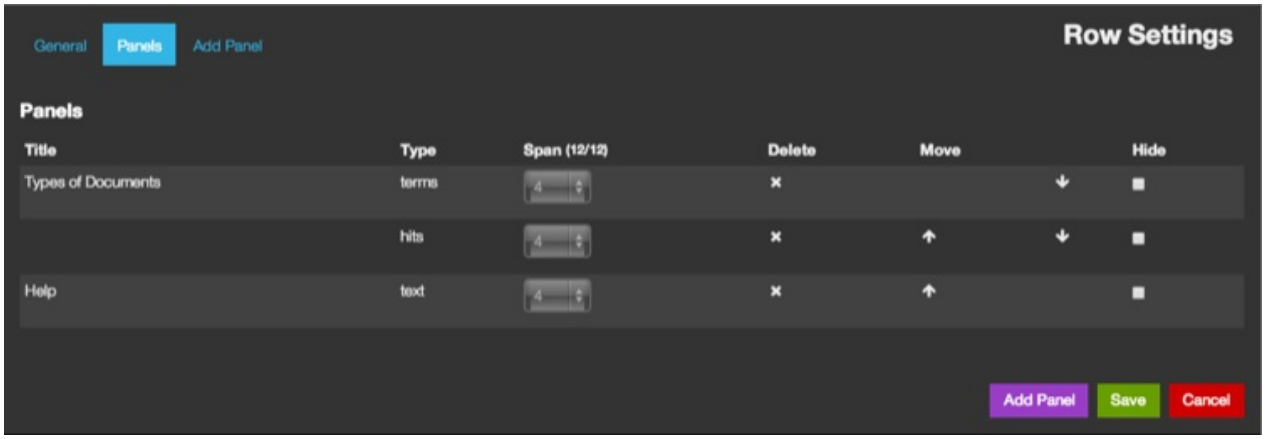
顶部的请求和过滤区域也可以被折叠。点击彩色标签就可以折叠和展开。



编辑行

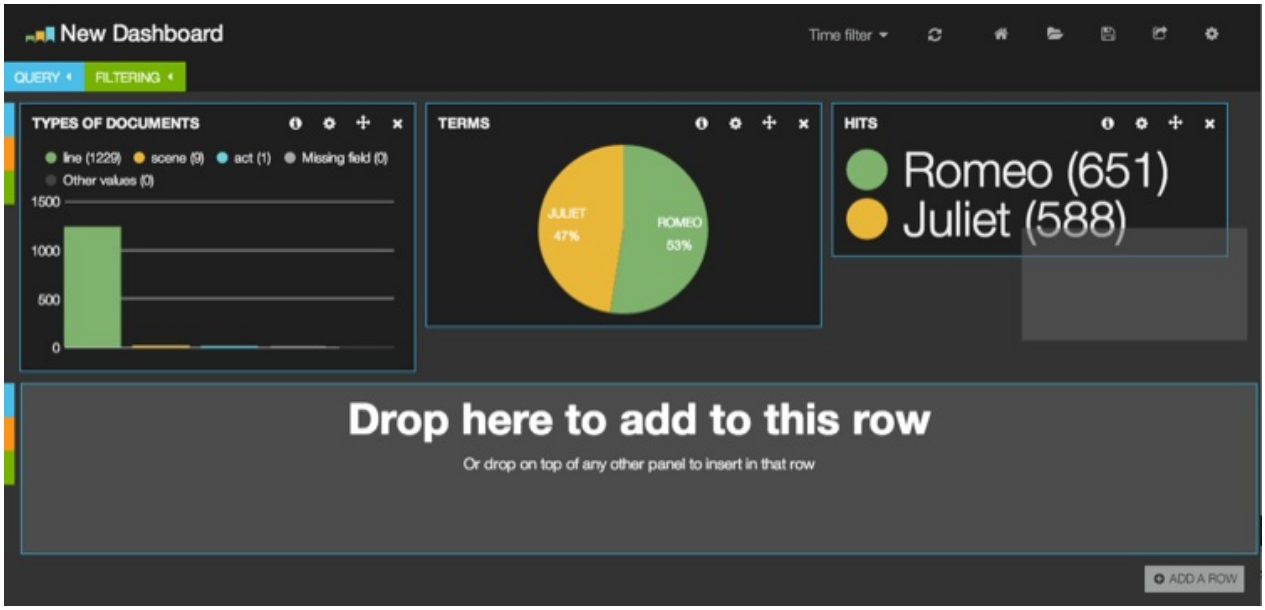
通过行编辑器，可以给行重命名，改行高等其他配置。点击橙色按钮打开行编辑器。

这个对话框还允许你修改面板的排序和大小，以及删除面板。

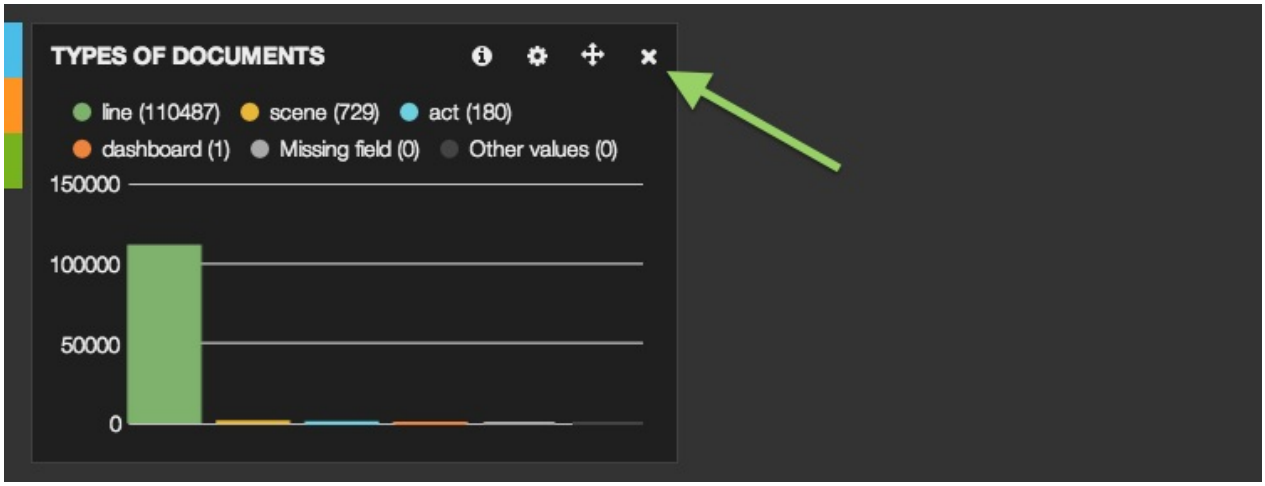


移动和删除面板

面板可以在本行，甚至其他行之间任意拖拽。按住面板右上角的十字架形状小图标然后拖动即可。



点击面板右上角的 *remove* 小图标就可以从仪表板上移除它。前面说到从行编辑器上也可以做到统一效果。



移动和删除行

行可以在仪表板配置页中重新排序和删。点击屏幕右上角的配置按钮，选择行(Rows)标签切换到行配置层。看到这里你一定会记起来我们在添加第一个行时候的屏幕。

左侧的箭头用来修改仪表板上行的次序。X 用来删除行。

下一步

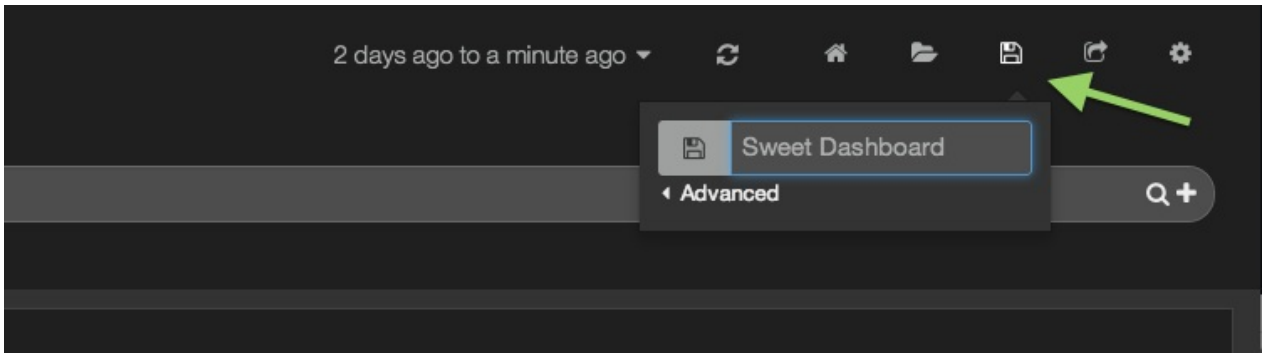
在你关闭浏览器之前，你可能打算保存这个新仪表板。请阅读 Saving and Loading dashboards。

保存和加载

你已经构建了一个漂亮的仪表板！现在你打算分享给团队，或者开启自动刷新后挂在一个大屏幕上？Kibana 可以把仪表板设计持久化到 Elasticsearch 里，然后在需要的时候通过加载菜单或者 URL 地址调用出来。

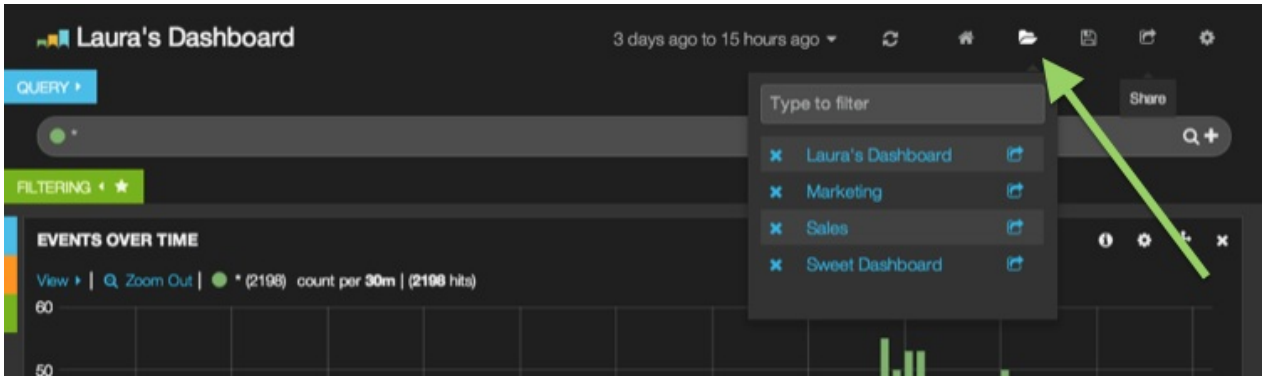
保存你漂亮的仪表板

保存你的界面非常简单，打开保存下拉菜单，取个名字，然后点击保存图表即可。现在你的仪表板就保存在一个叫做 `kibana-int` 的 Elasticsearch 索引里了。



调用你的仪表板

要搜索已保存的仪表板列表，点击右上角的加载图标。在这里你可以加载，分享和删除仪表板。



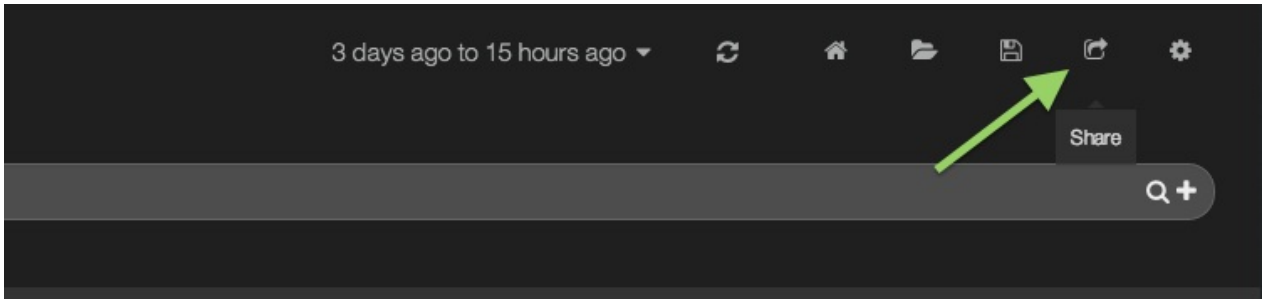
分享仪表板

已保存的仪表板可以通过你浏览器地址栏里的 URL 分享出去。每个持久化到 Elasticsearch 里的仪表板都有一个对应的 URL，像下面这样：

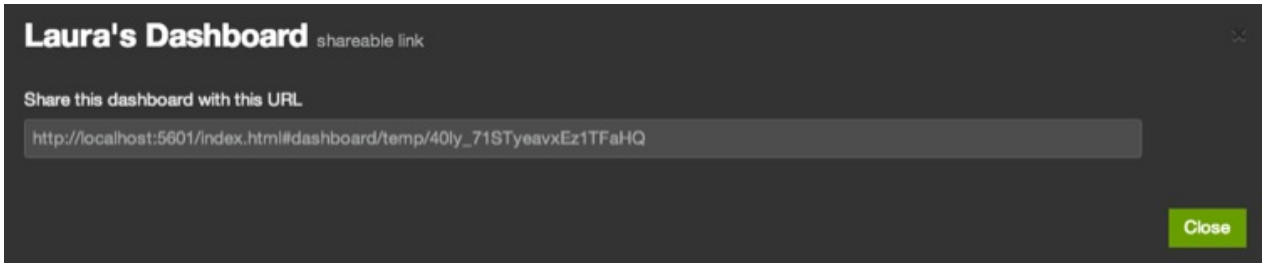
```
http://your_host/index.html#/dashboard/elasticsearch/MYDASHBOARD
```

这个示例中 `MYDASHBOARD` 就是你在保存的时候给仪表板取得名字。

你还可以分享一个即时的仪表板链接，点击 Kibana 右上角的分享图标，会生成一个临时 URL。



默认情况下，临时 URL 保存 30 天。



保存成静态仪表板

仪表板可以保存到你的服务器磁盘上成为 `.json` 文件。把文件放到 `app/dashboards` 目录，然后通过下面地址访问

```
http://your_host/index.html#/dashboard/file/MYDASHBOARD.json
```

`MYDASHBOARD.json` 就是磁盘上文件的名字。注意路径中得 `/#dashboard/file/` 看起来跟之前访问保存在 Elasticsearch 里的仪表板很类似，不过这里访问的是文件而不是 elasticsearch。导出的仪表板纲要的详细信息，阅读 The Dashboard Schema Explained

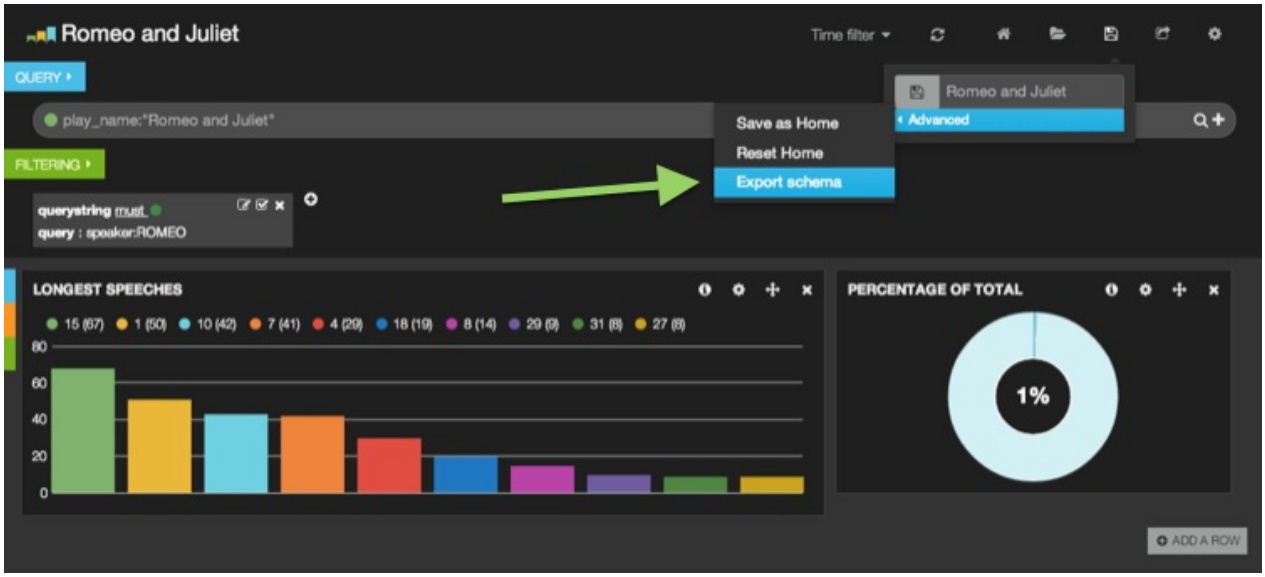
下一步

你现在知道怎么保存，加载和访问仪表板了。你可能想知道怎么通过 URL 传递参数来访问，这样可以在其他应用中直接链接过来。请阅读 Templated and Scripted Dashboards

仪表板纲要

Kibana 仪表板可以很容易的在浏览器中创建出来，而且绝大多数情况下，浏览器已经足够支持你创建一个很有用很丰富的节目了。不过，当你真的需要一点小修改的时候，Kibana 也可以让你直接编辑仪表板的纲要。

注意：本节内容只针对高级用户。JSON 语法非常严格，多一个逗号，少一个大括号，都会导致你的仪表板无法加载。



我们会用上面这个仪表板作为示例。你可以导出任意的仪表板纲要，点击右上角的保存按钮，指向高级(Advanced)菜单，然后点击导出纲要(Export Schema)。示例使用的纲要文件可以在这里下载： schema.json

因为仪表板是由特别长的 JSON 文档组成的，我们只能分成一段段的内容，分别介绍每段的作用和目的。

和所有的 JSON 文档一样，都是以一个大括号开始的。

```
{
```

服务(services)

```
"services": {
```

服务(Services)是被多个面板使用的持久化对象。目前仪表板对象附加有 2 种服务对象，不指明的话，就会自动填充成请求(query)和过滤(filter)服务了。

- Query
- Filter

query

```
"query": {
  "list": {
    "0": {
      "query": "play_name:\"Romeo and Juliet\"",
      "alias": "",
      "color": "#7EB26D",
      "id": 0,
      "pin": false,
      "type": "lucene",
      "enable": true
    }
  },
  "ids": [
    0
  ]
},
```

请求服务主要是由仪表板顶部的请求栏控制的。有两个属性：

- List: 一个以数字为键的对象。每个值描述一个请求对象。请求对象的键命名一目了然，就是描述请求输入框的外观和行为的。
- Ids: 一个由 ID 组成的数组。每个 ID 都对应前面 list 对象的键。 ids 数组用来保证显示时 list 的排序问题。

filter

```
"filter": {
  "list": {
    "0": {
      "type": "querystring",
      "query": "speaker:ROMEO",
      "mandate": "must",
      "active": true,
      "alias": "",
      "id": 0
    }
  },
  "ids": [
    0
  ]
}
},
```

过滤的行为和请求很像，不过过滤不能在面板级别选择，而是对全仪表板生效。过滤对象和请求对象一样有 list 和 ids 两个属性，各属性的行为和请求对象也一样。

垂幕(pulldown)

```
"pulldowns": [
```

垂幕是一种特殊的面板。或者说，是一个特殊的可以用来放面板的地方。在垂幕里的面板就跟在行里的一样，区别就是不能设置 span 宽度。垂幕里的面板永远都是全屏宽度。此外，垂幕里的面板也不可以吧被使用者移动或编辑。所以垂幕特别适合放置输入框。垂幕的属性是一个由面板对象构成的数组。关于特定的面板，请阅读 Kibana Panels

```
  {
   "type": "query",
   "collapse": false,
   "notice": false,
   "enable": true,
   "query": "*",
   "pinned": true,
   "history": [
    "play_name:\"Romeo and Juliet\"",
    "playname:\"Romeo and Juliet\"",
    "romeo"
   ],
   "remember": 10
  },
  {
   "type": "filtering",
   "collapse": false,
   "notice": true,
   "enable": true
  }
 ],
```

垂幕面板有 2 个普通行面板没有的选项：

- Collapse: 设置为真假值，代表着面板被折叠还是展开。
- Notice: 面板设置这个值，控制在垂幕的标签主题上出现一个小星星。用来通知使用者，这个面板里发生变动了。

导航(nav)

`nav` 属性里也有一个面板列表，只是这些面板是被用来填充在页首导航栏里德。目前唯一支持导航的面板是时间选择器(timepicker)

```
"nav": [
  {
    "type": "timepicker",
    "collapse": false,
    "notice": false,
    "enable": true,
    "status": "Stable",
    "time_options": [
      "5m",
      "15m",
      "1h",
      "6h",
      "12h",
      "24h",
      "2d",
      "7d",
      "30d"
    ],
    "refresh_intervals": [
      "5s",
      "10s",
      "30s",
      "1m",
      "5m",
      "15m",
      "30m",
      "1h",
      "2h",
      "1d"
    ],
    "timefield": "@timestamp"
  }
],
```

loader

`loader` 属性描述了仪表板顶部的保存和加载按钮的行为。

```
"loader": {
  "save_gist": false,
  "save_elasticsearch": true,
  "save_local": true,
  "save_default": true,
  "save_temp": true,
  "save_temp_ttl_enable": true,
  "save_temp_ttl": "30d",
  "load_gist": false,
  "load_elasticsearch": true,
  "load_elasticsearch_size": 20,
  "load_local": false,
  "hide": false
},
```

行数组

`rows` 就是通常放置面板的地方。也是唯一可以通过浏览器页面添加的位置。

```
"rows": [
   {
    "title": "Charts",
    "height": "150px",
    "editable": true,
    "collapse": false,
    "collapsable": true,
```

行对象包含了一个面板列表，以及一些行的具体参数，如下所示：

- title: 行的标题
- height: 行的高度，单位是像素，记作 px
- editable: 真假值代表面板是否可被编辑
- collapse: 真假值代表行是否被折叠
- collapsable: 真价值代表使用者是否可以折叠行

面板数组

行的 `panels` 数组属性包括有一个以自己出现次序排序的面板对象的列表。各特定面板本身的属性列表和说明，阅读 Kibana Panels

```
    "panels": [
      {
        "error": false,
        "span": 8,
        "editable": true,
        "type": "terms",
        "loadingEditor": false,
        "field": "speech_number",
        "exclude": [],
        "missing": false,
        "other": false,
        "size": 10,
        "order": "count",
        "style": {
          "font-size": "10pt"
        },
        "donut": false,
        "tilt": false,
        "labels": true,
        "arrangement": "horizontal",
        "chart": "bar",
        "counter_pos": "above",
        "spyable": true,
        "queries": {
          "mode": "all",
          "ids": [
            0
          ]
        },
        "tmode": "terms",
        "tstat": "total",
        "valuefield": "",
        "title": "Longest Speeches"
      },
      {
        "error": false,
        "span": 4,
        "editable": true,
        "type": "goal",
        "loadingEditor": false,
        "donut": true,
        "tilt": false,
        "legend": "none",
        "labels": true,
        "spyable": true,
        "query": {
          "goal": 111397
        },
        "queries": {
          "mode": "all",
          "ids": [
            0
          ]
        },
        "title": "Percentage of Total"
      }
    ]
  }
],
```

索引设置

索引属性包括了 Kibana 交互的 Elasticsearch 索引的信息。

```
"index": {
  "interval": "none",
  "default": "_all",
  "pattern": "[logstash-]YYYY.MM.DD",
  "warm_fields": false
},
```

- interval: none, hour, day, week, month。这个属性描述了索引所遵循的时间间隔模式。
- default: 如果 `interval` 被设置为 none，或者后面的 `failover` 设置为 `true` 而且没有索引能匹配上正则模式的话，搜索这里设置的索引。
- pattern: 如果 `interval` 被设置成除了 none 以外的其他值，就需要解析这里设置的模式，启用时间过滤规则，来确定请求哪些索引。
- warm_fields: 是否需要解析映射表来确定字段列表。

其余

下面四个也是顶层的仪表板配置项

```
  "failover": false,
  "editable": true,
  "style": "dark",
  "refresh": false
}
```

- failover: 真假值，确定在没有匹配上索引模板的时候是否使用 `index.default`。
- editable: 真假值，确定是否在仪表板上显示配置按钮。
- style: "亮色(light)" 或者 "暗色(dark)"
- refresh: 可以设置为 "false" 或者其他 elasticsearch 支持的时间表达式(比如 10s, 1m, 1h)，用来描述多久触发一次面板的数据更新。

导入纲要

默认是不能导入纲要的。不过在仪表板配置屏的控制(Controls)标签里可以开启这个功能，启用 "Local file" 选项即可。然后通过仪表板右上角加载图标的高级设置，选择导入文件，就可以导入纲要了。

模板和脚本

Kibana 支持通过模板或者更高级的脚本来动态的创建仪表板。你先创建一个基础的仪表板，然后通过参数来改变它，比如通过 URL 插入一个新的请求或者过滤规则。

模板和脚本都必须存储在磁盘上，目前不支持存储在 Elasticsearch 里。同时它们也必须是通过编辑或创建纲要生成的。所以我们强烈建议阅读 The Kibana Schema Explained

仪表板目录

仪表板存储在 Kibana 安装目录里的 `app/dashboards` 子目录里。你会注意到这里面有两种文件：`.json` 文件和 `.js` 文件。

模板化仪表板(.json)

`.json` 文件就是模板化的仪表板。模板示例可以在 `logstash.json` 仪表板的请求和过滤对象里找到。模板使用 handlebars 语法，可以让你在 json 里插入 javascript 语句。URL 参数存在 `ARGS` 对象中。下面是 `logstash.json` (on github) 里请求和过滤服务的代码片段：

```
"0": {
  "query": "{{ARGS.query || '*'}}",
  "alias": "",
  "color": "#7EB26D",
  "id": 0,
  "pin": false
}

[...]

"0": {
  "type": "time",
  "field": "@timestamp",
  "from": "now-{{ARGS.from || '24h'}}",
  "to": "now",
  "mandate": "must",
  "active": true,
  "alias": "",
  "id": 0
}
```

这允许我们在 URL 里设置两个参数， `query` 和 `from` 。如果没设置，默认值就是 `||` 后面的内容。比如说，下面的 URL 就会搜索过去 7 天内 `status:200` 的数据：

注意：千万注意 url `#/dashboard/file/logstash.json` 里的 `file` 字样

```
http://yourserver/index.html#/dashboard/file/logstash.json?query=status:200&from=7d
```

脚本化仪表板(.js)

脚本化仪表板比模板化仪表板更加强大。当然，功能强大随之而来的就是构建起来也更复杂。脚本化仪表板的目的就是构建并返回一个描述了完整的仪表板纲要的 javascript 对象。 `app/dashboards/logstash.js` (on github) 就是一个有着详细注释的脚本化仪表板示例。这个文件的最终结果和 `logstash.json` 一致，但提供了更强大的功能，比如我们可以以逗号分割多个请求：

注意：千万注意 URL `#/dashboard/script/logstash.js` 里的 `script` 字样。这让 kibana 解析对应的文件为 javascript 脚本。

```
http://yourserver/index.html#/dashboard/script/logstash.js?query=status:403,status:404&from=7d
```

这会创建 2 个请求对象， `status:403` 和 `status:404` 并分别绘图。事实上这个仪表板还能接收另一个参数 `split`，用于指定用什么字符串切分。

```
http://yourserver/index.html#/dashboard/script/logstash.js?query=status:403!status:404&from=7d&split=!
```

我们可以看到 `logstash.js` (on github) 里是这么做的：

```
// In this dashboard we let users pass queries as comma separated list to the query parameter.
// Or they can specify a split character using the split aparameter
// If query is defined, split it into a list of query objects
// NOTE: ids must be integers, hence the parseInt()s
if(!_.isUndefined(ARGS.query)) {
 queries = _.object(_.map(ARGS.query.split(ARGS.split||','), function(v,k) {
  return [k,{
   query: v,
   id: parseInt(k,10),
   alias: v
  }];
 }));
} else {
 // No queries passed? Initialize a single query to match everything
 queries = {
  0: {
   query: '*',
   id: 0,
  }
 };
}
```

该仪表板可用参数比上面讲述的还要多，全部参数都在 `logstash.js` (on github) 文件里开始的注释中有讲解。

配置

config.js 是 Kibana 核心配置的地方。文件里包括得参数都是必须在初次运行 kibana 之前提前设置好的。

参数

elasticsearch

你 elasticsearch 服务器的 URL 访问地址。你应该不会像写个 `http://localhost:9200` 在这，哪怕你的 Kibana 和 Elasticsearch 是在同一台服务器上。默认的时候这里会尝试访问你部署 kibana 的服务器上的 ES 服务，你可能需要设置为你 elasticsearch 服务器的主机名。

注意：如果你要传递参数给 http 客户端，这里也可以设置成对象形式，如下：

```
+elasticsearch: {server: "http://localhost:9200", withCredentials: true}+
```

default_route

没有指明加载哪个仪表板的时候，默认加载页路径的设置参数。你可以设置为文件，脚本或者保存的仪表板。比如，你有一个保存成 "WebLogs" 的仪表板在 elasticsearch 里，那么你就可以设置成：

```
default_route: /dashboard/elasticsearch/WebLogs,
```

kibana-int

默认用来保存 Kibana 相关对象，比如仪表板，的 Elasticsearch 索引名称。

panel_name

可用的面板模块数组。面板只有在仪表板中有定义时才会被加载，这个数组只是用在 "add panel" 界面里做下拉菜单。

面板

Kibana 仪表板由面板(panels)块组成。面板在行中可以起到很多作用，不过大多数是用来给一个或者多个请求的数据集结果做可视化。剩下一些面板则用来展示数据集或者用来为使用者提供插入指令的地方。

面板可以很容易的通过 Kibana 网页界面配置。为了了解更高级的用法，比如模板化或者脚本化仪表板，这章开始介绍面板属性。你可以发现一些通过网页界面看不到的设置。

每个面板类型都有自己的属性，不过有这么几个是大家共有的。

span

一个从 1-12 的数字，描述面板宽度。

editable

是否在面板上显示编辑按钮。

type

本对象包含的面板类型。每个具体的面板类型又要求添加新的属性，具体列表说明稍后详述。

trends

Status: Beta

A stock-ticker style representation of how queries are moving over time. For example, if the time is 1:10pm, your time picker was set to "Last 10m", and the "Time Ago" parameter was set to "1h", the panel would show how much the query results have changed since 12:00-12:10pm

参数

- ago

  A date math formatted string describing the relative time period to compare the queries to.

- arrangement

  ‘horizontal’ or ‘vertical’

- spyable

  设为假，不显示审查(inspect)按钮。

请求**(queries)**

- 请求对象

  这个对象描述本面板使用的请求。

  - queries.mode

    在可用请求中应该用哪些？可设选项有：`all, pinned, unpinned, selected`

  - queries.ids

    如果设为 `selected` 模式，具体被选的请求编号。

文本(text)

状态：稳定

文本面板用来显示静态文本内容，支持 markdown，简单的 html 和纯文本格式。

参数

- mode

  ‘html’, ‘markdown’ 或者 ‘text’

- content

  面板内容，用 `mode` 参数指定的标记语言书写

terms

状态：稳定

基于 Elasticsearch 的 terms facet 接口数据展现表格，条带图，或者饼图。

参数

- field

  用于计算 facet 的字段名称。

- exclude

  要从结果数据中排除掉的 terms

- missing

  设为假，就可以不显示数据集内有多少结果没有你指定的字段。

- other

  设为假，就可以不显示聚合结果在你的 `size` 属性设定范围以外的总计数值。

- size

  显示多少个 terms

- order

  terms 模式可以设置：count, term, reverse_count 或者 reverse_term；terms_stats 模式可以设置：term, reverse_term, count, reverse_count, total, reverse_total, min, reverse_min, max, reverse_max, mean 或者 reverse_mean

- donut

  在饼图(pie)模式，在饼中画个圈，变成甜甜圈样式。

- tilt

  在饼图(pie)模式，倾斜饼变成椭圆形。

- lables

  在饼图(pie)模式，在饼图分片上绘制标签。

- arrangement

  在条带(bar)或者饼图(pie)模式，图例的摆放方向。可以设置：水平(horizontal)或者垂直(vertical)。

- chart

  可以设置：table, bar 或者 pie

- counter_pos

  图例相对于图的位置，可以设置：上(above)，下(below)，或者不显示(none)。

- spyable

  设为假，不显示审查(inspect)按钮。

请求**(queries)**

- 请求对象

  这个对象描述本面板使用的请求。

- queries.mode

  在可用请求中应该用哪些？可设选项有：`all, pinned, unpinned, selected`

- queries.ids

  如果设为 `selected` 模式，具体被选的请求编号。

- tmode

  Facet 模式：terms 或者 terms_stats

- tstat

  Terms_stats facet stats 字段。

- valuefield

  Terms_stats facet value 字段。

table

状态：稳定

表格面板里是一个可排序的分页文档。你可以定义需要排列哪些字段，并且还提供了一些交互功能，比如执行 terms 聚合查询。

参数

- size

  每页显示多少条

- pages

  展示多少页

- offset

  当前页的页码

- sort

  定义表格排序次序的数组，示例如右：['@timestamp','desc']

- overflow

  css 的 overflow 属性。'min-height' (expand) 或 'auto' (scroll)

- fields

  表格显示的字段数组

- highlight

  高亮显示的字段数组

- sortable

  设为假关掉排序功能

- header

  设为假隐藏表格列名

- paging

  设为假隐藏表格翻页键

- field_list

  设为假隐藏字段列表。使用者依然可以展开它，不过默认会隐藏起来

- all_fields

  设为真显示映射表内的所有字段，而不是表格当前使用到的字段

- trimFactor

  裁剪因子(trim factor)，是参考表格中的列数来决定裁剪字段长度。比如说，设置裁剪因子为 100，表格中有 5 列，那么每列数据就会被裁剪为 20 个字符。完整的数据依然可以在展开这个事件后查看到。

- localTime

  设为真调整 `timeField` 的数据遵循浏览器的本地时区。

- timeField

如果 `localTime` 设为真，该字段将会被调整为浏览器本地时区。

- spyable

  设为假，不显示审查(inspect)按钮。

请求(queries)

- 请求对象

  这个对象描述本面板使用的请求。

  - queries.mode

    在可用请求中应该用哪些？可设选项有：`all, pinned, unpinned, selected`

  - queries.ids

    如果设为 `selected` 模式，具体被选的请求编号。

sparklines

Status: Experimental

The sparklines panel shows tiny time charts. The purpose of these is not to give an exact value, but rather to show the shape of the time series in a compact manner

参数

- mode Value to use for the y-axis. For all modes other than count, `value_field` must be defined. Possible values: count, mean, max, min, total.
- time_field x-axis field. This must be defined as a date type in Elasticsearch.
- value_field y-axis field if mode is set to mean, max, min or total. Must be numeric.
- interval Sparkline intervals are computed automatically as long as there is a time filter present. In the absence of a time filter, use this interval.
- spyable Show inspect icon

请求**(queries)**

- 请求对象 这个对象描述本面板使用的请求。
  - queries.mode 在可用请求中应该用哪些？可设选项有：`all, pinned, unpinned, selected`
  - queries.ids 如果设为 `selected` 模式，具体被选的请求编号。

map

状态：稳定

map 面板把 2 个字母的国家或地区代码转成地图上的阴影区域。目前可用的地图包括世界地图，美国地图和欧洲地图。

参数

- map

  显示哪个地图：world, usa, europe

- colors

  用来涂抹地图阴影的颜色数组。一旦设定好这 2 个颜色，阴影就会使用介于这 2 者之间的颜色。示例 ['#A0E2E2', '#265656']

- size

  阴影区域的最大数量

- exclude

  排除的区域数组。示例 ['US','BR','IN']

- spyable

  设为假，不显示审查(inspect)按钮。

请求**(queries)**

- 请求对象

  这个对象描述本面板使用的请求。

  - queries.mode

    在可用请求中应该用哪些？可设选项有：`all, pinned, unpinned, selected`

  - queries.ids

    如果设为 `selected` 模式，具体被选的请求编号。

hits

Status: Stable

The hits panel displays the number of hits for each of the queries on the dashboard in a configurable format specified by the ‘chart’ property.

参数

- arrangement

  在条带(bar)或者饼图(pie)模式，图例的摆放方向。可以设置：水平(horizontal)或者垂直(vertical)。

- chart

  可以设置：none, bar 或者 pie

- counter_pos

  图例相对于图的位置，可以设置：上(above)，下(below)

- donut

  在饼图(pie)模式，在饼中画个圈，变成甜甜圈样式。

- tilt

  在饼图(pie)模式，倾斜饼变成椭圆形。

- lables

  在饼图(pie)模式，在饼图分片上绘制标签。

- spyable

  设为假，不显示审查(inspect)图标。

请求**(queries)**

- 请求对象

  这个对象描述本面板使用的请求。

  - queries.mode

    在可用请求中应该用哪些？可设选项有：`all, pinned, unpinned, selected`

  - queries.ids

    如果设为 `selected` 模式，具体被选的请求编号。

histogram

状态：稳定

histogram 面板用以显示时间序列图。它包括好几种模式和变种，用以显示时间的计数，平均数，最大值，最小值，以及数值字段的和，计数器字段的导数。

参数

轴**(axis)**参数

- mode

  用于 Y 轴的值。除了 count 以外，其他 `mode` 设置都要求定义 `value_field` 参数。可选值为：count, mean, max, min, total。

- time_field

  X 轴字段。必须是在 Elasticsearch 中定义为时间类型的字段。

- value_field

  如果 `mode` 设置为 mean, max, min 活着 total，Y 轴字段。必须是数值型。

- x-axis

  是否显示 X 轴。

- y-axis

  是否显示 Y 轴。

- scale

  以该因子规划 Y 轴

- y_format

  Y 轴数值格式，可选：none, bytes, short

注释

- 注释对象

  可以指定一个请求的结果作为标记显示在图上。比如说，标记某时刻部署代码了。

  - annotate.enable

    是否显示注释(即标记)

  - annotate.query

    标记使用的 Lucene query_string 语法请求

  - annotate.size

    最多显示多少标记

  - annotate.field

    显示哪个字段

  - annotate.sort

    数组排序，格式为 [field,order]。比如 [‘@timestamp’,‘desc’]，这是一个内部参数。

- auto_int

是否自动调整间隔

- resolution

  如果 `auto_int` 设为真，shoot for this many bars.

- interval

  如果 `auto_int` 设为假，用这个值做间隔

- intervals

  在 `View` 选择器里可见的间隔数组。比如 ['auto','1s','5m','3h']，这是绘图参数。

- lines

  显示折线图

- fill

  折线图的区域填充因子，从 1 到 10。

- linewidth

  折线的宽度，单位为像素

- points

  在图上显示数据点

- pointradius

  数据点的大小，单位为像素

- bars

  显示条带图

- stack

  堆叠多个序列

- spyable

  显示审核图标

- zoomlinks

  显示 'Zoom Out' 链接

- options

  显示快捷的 view 选项区域

- legend

  显示图例

- show_query

  如果没设别名(alias)，是否显示请求

- interactive

  允许点击拖拽进行放大

- legend_counts

  在图例上显示计数

- timezone

  是否调整成浏览器时区。可选值为：browser, utc

- percentage

  把 Y 轴数据显示成百分比样式。仅对多个请求时有效。

- zerofill

  提高折线图准确度，稍微消耗一点性能。

- derivative

  在 X 轴上显示该点数据在前一个点数据上变动的数值。

- 提示框(tooltip)对象

  - tooltip.value_type

    控制 tooltip 在堆叠图上怎么显示，可选值：独立(individual)还是累计(cumulative)

  - tooltip.query_as_alias

    如果没设别名(alias)，是否显示请求

- 网格(grid)对象

  Y 轴的最大值和最小值

  - grid.min

    Y 轴的最小值

  - grid.max

    Y 轴的最大值

请求**(queries)**

- 请求对象

  这个对象描述本面板使用的请求。

  - queries.mode

    在可用请求中应该用哪些？可设选项有：`all, pinned, unpinned, selected`

  - queries.ids

    如果设为 `selected` 模式，具体被选的请求编号。

goal

Status: Stable

The goal panel display progress towards a fixed goal on a pie chart

参数

- donut Draw a hole in the middle of the pie, creating a tasty donut.
- tilt Tilt the pie back into an oval shape
- legend The location of the legend, above, below or none
- labels Set to false to disable drawing labels inside the pie slices
- spyable Set to false to disable the inspect function.

query

- query object
  - query.goal the fixed goal for goal mode

请求**(queries)**

- 请求对象 这个对象描述本面板使用的请求。
  - queries.mode 在可用请求中应该用哪些？可设选项有：`all, pinned, unpinned, selected`
  - queries.ids 如果设为 `selected` 模式，具体被选的请求编号。

column

状态：稳定

这是一个伪面板。目的是让你在一列中添加多个其他面板。虽然 column 面板状态是稳定，它的限制还是很多的，比如不能拖拽内部的小面板。未来的版本里，column 面板可能被删除。

参数

- panel

  面板对象构成的数组

bettermap

状态：实验性

Bettermap 之所以叫 bettermap 是因为还没有更好(better)的名字。Bettermap 使用地理坐标来在地图上创建标记集群，然后根据集群的密度，用橘色，黄色和绿色作为区分。

要查看细节，点击标记集群。地图会放大，原有集群分裂成更小的集群。一直小到没法成为集群的时候，单个标记就会显现出来。悬停在标记上可以查看 `tooltip` 设置的值。

注意: bettermap 需要从互联网上下载它的地图面板文件。

参数

- field

  包含了地理坐标的字段，要求是 geojson 格式。GeoJSON 是一个数组，内容为 `[longitude,latitude]` 。这可能跟大多数实现([latitude, longitude])是反过来的。

- size

  用来绘制地图的数据集大小

- spyable

  设为假，不显示审查(inspect)按钮。

- tooltip

  悬停在标记上时显示哪个字段。

请求**(queries)**

- 请求对象

  这个对象描述本面板使用的请求。

  - queries.mode

    在可用请求中应该用哪些？可设选项有： `all, pinned, unpinned, selected`

  - queries.ids

    如果设为 `selected` 模式，具体被选的请求编号。

Table of Contents