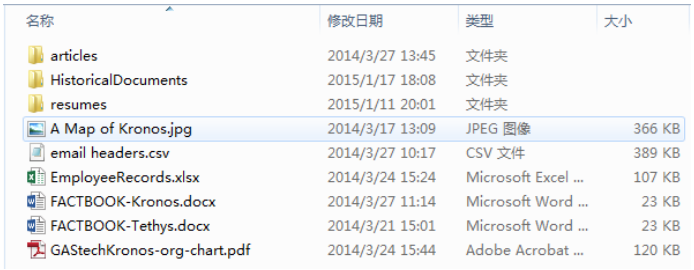
团队项目报告文档

课程：信息可视化　教师：陈为教授
队长：秦昇　　学号：3120000060
队员：胡冯钦　　学号：3120101854
　　黄靖　　学号：3120101911
　　张凯　　学号：3110102981

系统开发说明：

1. 问题来源为 VAST2014，Mini Challenge 1。提供的数据如下图所示



待解决的问题有：1. Who are the leaders，2. Who is part of the extended network 等。

2. 本组开发的项目基于 d3.js 库，html 网页内嵌 JavaScript 语言，将网页组织成为一个信息系统的形式来呈现本小组对所提问题的研究解决过程。

3. 系统实现的主要功能有：

① 呈现本小组在分析问题时实现的若干数据的静态图表。

② 以 timeline 的形式呈现出 HistoricalDocuments 中介绍的 POK 的历史，以熟悉问题背景和具体历史进程。。

③ 以力图的形式呈现出小组解读出的 POK 和 GAStech 等主要人物的关系图，并附以 POK 主要 leader 的关系和具体介绍

④ 以弦图的方式呈现 GAStech 内部员工的邮件往来和信息。

⑤ 统计 844 篇 articles 中词频最高的 keywords 以堆栈图的形式来呈现出 keywords 出现频率随年份变化以及所占百分比的变化，并画出对应 keywords 的词云的图像。

⑥ 附加了一个实现了对 keywords 的检索，并显示出其在 articles 中的出现位置和具体文本信息的软件。

4. 本系统旨在帮助分析并解决问题，实现所拥有数据的可视化以及交互分析，在我们分析和解决问题的过程中起到了重要的作用，亦可用作他人对问题认识和分析的重要工具。

具体实现模块

1. POK timeline

解读 HistoricalDocuments，将重要信息的时间、主要内容和简介记录下来，附以图片，并将记录信息以 timeline.json 的文件格式保存。
Timeline 的设计套用了已有的模板，只是对数据进行了修改和增加了图像。

2. Force Layout——POK 总体架构

力图，又叫力导向（force-directed）布局，模拟了物理学中的力在屏幕上排列元素的机制。力导向布局典型地要使用网状数据，而且可以方便地进行拖动，便于查看组织架构，梳理人物关系。

力图由一组结点（node）和连线（edge）构成。结点代表数据集中的实体，连线代表结点之间的关系，有些结点可能有连线，而有些结点可能没有连线。连线（edges）则至少包含两个值：来源（source）ID 和目标（target）ID。
此处，结点为 POK 组织成员或与 POK 组织成员有关的人物，以不同的图标表示其不同的身份。

 ，表示 POK，属于 POK 的成员直接与此图标相连；

 ，表示 POK 的领导者；

 ，表示 POK 普通成员；

 ，表示 GASTech 员工；

 ，表示已死亡。

Lucio Jakab (2005)

每个结点旁都有名字标识，括号中的年份代表其加入 POK 的时间。
使用嵌套性较好的 JSON 格式表示人物及其关系，具体如下 POK.json：

```
{
    "nodes" :[
        { "name" : "POK" , "image" : "logo.png" , "startyear" : "1997" , "endyear" : "2001" },
        { "name" : "Elian Karel" , "image" : "Leader.png" , "startyear" : "1997" , "endyear" : "2001"  },
        { "name" : "Henk Bodrogi" , "image" : "Leader.png" , "startyear" : "1997" , "endyear" : "" },
        { "name" : "Carmine Osvaldo" , "image" : "pok.png" , "startyear" : "1997" , "endyear" : "" },
        { "name" : "Jeroen Karel" , "image" : "pok.png" , "startyear" : "1997" , "endyear" : "" },
        { "name" : "Ale L. Hanne" , "image" : "pok.png" , "startyear" : "1997" , "endyear" : "" },
        { "name" : "Valentine Mies" , "image" : "pok.png" , "startyear" : "1997" , "endyear" : "" },
        { "name" : "Yanick Cato" , "image" : "pok.png" , "startyear" : "1997" , "endyear" : "" },
        { "name" : "Joreto Katell" , "image" : "pok.png" , "startyear" : "1997" , "endyear" : "" },
        { "name" : "Silvia Marek" , "image" : "Leader.png" , "startyear" : "2005" , "endyear" : "" },
        { "name" : "Mandor Vann" , "image" : "pok.png" , "startyear" : "2004" , "endyear" : "" },
        { "name" : "Isia Vann" , "image" : "pok.png" , "startyear" : "" , "endyear" : "" },
        { "name" : "Lucio Jakab" , "image" : "pok.png" , "startyear" : "2005" , "endyear" : "" },
        { "name" : "Antaura Karel" , "image" : "pok.png" , "startyear" : "" , "endyear" : "" },
```



```
"edges" :[
    { "source" : 0 , "target" : 1},
    { "source" : 0 , "target" : 2},
    { "source" : 0 , "target" : 3},
    { "source" : 0 , "target" : 4},
    { "source" : 0 , "target" : 5},
    { "source" : 0 , "target" : 6},
    { "source" : 0 , "target" : 7},
    { "source" : 0 , "target" : 8},
    { "source" : 0 , "target" : 9},
    { "source" : 0 , "target" : 10},
    { "source" : 0 , "target" : 11},
    { "source" : 0 , "target" : 12},
    { "source" : 0 , "target" : 13},
    { "source" : 0 , "target" : 22},
    { "source" : 1 , "target" : 4},
    { "source" : 1 , "target" : 13},
    { "source" : 2 , "target" : 18},
```


核心步骤如下：

1.1 设置画布长宽

1.2 设置图片结点大小

1.3 创建 SVG

1.4 读取 json 文件中的数据

1.5 创建力图关系

其中结点为 root.nodes，连接边为 root.edges。

经过尝试，将连接边长设为 200，结点间的力设为斥力 650.

1.6 设置连接边样式

.data(root.edges)，即调用数据中的结点关系，对每个 source 和 target 做连接。

连接边的样式设为灰色，宽度为 3。

1.7 创建力图结点

.selectAll(“image”) 将图标作为结点表现形式；

.data(root.nodes)读取结点数据；

.append("image")

.attr("width",img_w)

.attr("height",img_h)

.attr("xlink:href",function(d){ return d.image;})

链接并使用使用图片，设置大小；

.call(force.drag);

```
var nodes_img = svg.selectAll("image")
                    .data(root.nodes)
                    .enter()
                    .append("image")
                    .attr("width",img_w)
                    .attr("height",img_h)
                    .attr("xlink:href",function(d){
                        return d.image;
                    })
                    .call(force.drag);
```

设置结点边的姓名标注

在 CSS 样式中对结点图片进行设置，鼠标未移过时，透明度为 0.7，鼠标移过时透明度为 1.0。从而实现鼠标移过为点击住拖拽时，有突出的效果。

```
<style>

    image{
        opacity: 0.7;
    }
    image:hover{
        opacity: 1.0;
```

3. 交互式人物框架图

根据文档分析可得，POK 早期核心成员为 7 人，Henk Bodrogi 为领导者，后由 Elaine Karel 接棒，未来可能的领导者为 Silvia 和 Mandor，同时文档提供了较丰富的文字描述核心成员的经历。故选择用 d3.js 绘制结构图，并加上一些交互功能，以便更清楚地查看核心成员的资料和关系。

2.1 设置常用长宽

2.2 创建 SVG

2.3 创建主连接线，即竖线

2.4 创建一个人物，以圆角矩形为主体表示

属性 x, y 设置矩形位置；width, height 设置矩形长宽；rx, ry 设置圆角曲度。

2.5 设置连接线连接至框架上

2.6 在该矩形中标上名字

位置由矩形位置相对计算确定。

```
svg.append("text")
    .attr("x", 1*width/8 )
    .attr("y", 2*height/4-40)
    .attr("dy", "20px")
    .attr("dx", "50px")
    .style("text-anchor", "middle")
    .text("Henk")
    .append("tspan")
    .attr("x", 1*width/8 )
    .attr("y", 2*height/4-25)
    .attr("dy", "20px")
    .attr("dx", "50px")
    .text("Bodrogi");
```

其他任务创建方式相似。

2.7 为每个人物结点创建点击动作

点击一个人物时，刷新所有结点的颜色，选中的那个变色。

文本框改为 visible；.text 函数附上文字。

由于 Silvia 和 Mandor 是潜在领导者，因此只有在点击现任领导者 Elain 时才会显示。同时创建相应箭头和文字说明。

```
Henk_Bodrogi.on("click", function() {
    Carmine_Osvaldo.transition()
        .style("fill", "#6699FF");
    Jeroen_Karel.transition()
        .style("fill", "#6699FF");
    Elain_Karel.transition()
        .style("fill", "#6699FF");
    Ale_Hanne.transition()
        .style("fill", "#6699FF");
    Joreto_Katell.transition()
        .style("fill", "#6699FF");
    Yanick_Cato.transition()
        .style("fill", "#6699FF");
    Valentine_Mies.transition()
        .style("fill", "#6699FF");
    Henk_Bodrogi.transition()
        .style("fill", "#003366");

    textplace.style("visibility", "visible");
    textplace.transition().style("fill", "#003366");
    texttitle.text("The grassroots organization coalesced under
    svg.selectAll(".invisible").data([]).exit().remove();
    Silvia_Marek.style("visibility", "hidden");
    Mandor_Vann.style("visibility", "hidden");
    d3.event.stopPropagation();
});
```

4. 邮件往来弦图

GAStech 邮件往来以一份 csv 格式文件给出，参与往来的人数不多，邮件往来可以通过主题来分类，所以可以用弦图来呈现。

首先对数据进行处理，具体所需 json 格式如下图所示

```
[{
    "name":"POK.Isia_Vann",
    "mails":[
            {"type":"Stalking",
             "subject":"This weekend?",
             "replies":0,
             "to":"Rachel_Pantanal"
             },
            {"type":"Stalking",
             "subject":"Karoake night",
             "replies":0,"to":"Rachel_Pantanal"
             },
            {"type":"Stalking",
             "subject":"This weekend?",
             "replies":0,
             "to":"Rachel_Pantanal"
             },
             {"type":"Stalking","subject":"Still busy?",
              "replies":0,"to":"Rachel_Pantanal"
              },
    "imports":[
              {"name":"Facilities.Claudio_Hawelon","size":3},
              {"name":"Administration.Rachel_Pantanal","size":2},
              {"name":"Administration.Rachel_Pantanal","size":2},
              {"name":"Administration.Rachel_Pantanal","size":2},
```

可通过如下过程处理数据

```
 for each mail do
 begin
    if source not exists
      begin
        add a new person s
        add a mail to s
        if end not exists
          begin
            add a new person e
            add an import to e
          end
        else
          add an import to e
      end
    else
      begin
       add a mail to source
       if end not exists
          begin
            add a new person e
            add an import to e
          end
        else
          add an import to e
      end
 if mail is a reply
    begin
         search mail whose source and end is contrary to
 the current mail
     add a reply to the mail
   end
   end
```

上述过程类似于建立一张无向图，每次添加一条边，然后检索顶点如果不存在则添加新的顶点，并且每添加一条边，对应同主题 mail 的数目增加 1，若是 reply，则 reply 增加 1 这样一个过程。

成功处理数据以后开始建图。图像主要由两部分构成，体现 mail 往来关系的弦图和对应 mail 信息的表格。

弦图的边和顶点分别通过如下代码创建

```
     link = link
         .data(bundle(links))
         .enter().append("path")
```

```
        .each(function(d) { d.source = d[0], d.target =
d[d.length - 1]; })
        .attr("class", "link")
        .attr("d", line);



 node = node
        .data(nodes.filter(function(n)
{ return !n.children; }))
        .enter().append("text")
        .attr("class", function(d) { return "node "
+d.parent.key })
        .attr("dx", function(d) { return d.x < 180 ? 8 : -
8; })
        .attr("dy", ".31em")
        .attr("transform", function(d) { return "rotate("
+ (d.x - 90) + ")translate(" + d.y + ")" + (d.x < 180 ?
"" : "rotate(180)"); })
        .style("text-anchor", function(d) { return d.x <
180 ? "start" : "end"; })
        .text(function(d) { return d.key; })
        .on("mouseover", mouseovered)
        .on("mouseout", mouseouted);
```

设置鼠标悬停动作

```
function mouseovered(d) {
        node
            .each(function(n) { n.target = n.source =
false; });

        link
            .classed("link--target", function(l) { if
(l.target === d) return l.source.source = true; })
            .classed("link--source", function(l) { if
(l.source === d) return l.target.target = true; })
            .filter(function(l) { return l.target === d ||
l.source === d; })
            .each(function()
{ this.parentNode.appendChild(this); });

        node
            .classed("node--target", function(n) { return
n.target; })
```

```
            .classed("node--source", function(n) { return
n.source; });
        svg1.selectAll(".table").style("visibility",
"visible");
        svg1.selectAll(".invisible").style("visibility",
"visible");
        fillthetable(d);
    }
```

其中 filltable 为在表格中显示具体信息的函数，首先要创建表格，如

```
var repliesText = svg1.append("foreignObject")
    .attr("class", "table")
    .attr("x", diameter+310)
    .attr("y", 120)
    .attr("width", 10)
    .attr("height", 150);
```

然后实现 filltable 函数

```
function fillthetable(d){
        var to ="";
        var subject ="";
        var replies = "";
        if (d.mails !== undefined){
        d.mails.forEach(function(element, index, array){
            to    +=    "<span    class='"+element.type+"'
style='width:90;'>"+element.to+"</span><br/>";
            subject   +=   "<span   class='"+element.type+"'
style='width:140;'>"+element.subject+"</span><br/>";
            replies                                  +="<span
class='"+element.type+"'style='width:10;'>"+
element.replies+"</span><br/>";
        });
        }
        toText.html("<div                class='tableText'
style='width:180;      height:150;word-wrap:      break-
word;'>"+to+"</div>");
        subjectText.html("<div           class='tableText'
style='width:180;height:150;word-wrap:            break-
word;'>"+subject+"</div>");
        repliesText.html("<div
class='tableText'>"+replies+"</div>");

    }
```

至此主要内容已完成，还可添加对人物按照部门分类，以及鼠标移开 Mouseout 的动作等。

5. 关键词查找引擎

主要涉及的数据：
844 篇 article（具体包括新闻稿，个人微博，博客短讯的形式）

5.1 具体代码分析:
文章的遍历：

```
file = new File(root); //确定文件目录
filename = file.list(); //利用 JAVA 的库得到目录下所有文件的列表（按照字符串数据的形式储存）
```

无用词语的过滤：

```
for(String s : unuseword)
if(cmp.equalsIgnoreCase(s))
return true;
/*for-each 语句判断这个单词是否在无用词语的列表中，是的返回 true*/
```

句子的分割&关键词检索：

```
if(readin.contains(".")) {
/*分割句子*/
sen = readin.split(".");
for (String eachs : sen) {
/*查找是否有关键词*/
if (contains(eachs, keywords)) {
/*统计出现次数*/
cnt++;
/*找到关键词输出这句话*/
System.out.println(s + ": " + eachs + ".");
}
}
```

6. Keywords 的 StackArea 图以及 wordcloud 图

首先通过改写上述关键词检索软件，进行所有词语的统计，并且按照年份记录其出现的频率。在过滤掉无用词语之后将词和出现总次数作为分量，建立词频的向量空间，并且将数据导入生成词云的软件，得到词云图像。

将向量空间中频率分量排前 30 位的向量数据导入按照格式导入 json 文件，绘制 StackArea 图像。

StackArea 图像的绘制基于 nv.d3 库，nv.d3 库是对 d3.js 库的封装，主要用于绘制统计图表等。根据以下代码

```
function defaultChartConfig(container, data, useGuideline) {
  if (useGuideline === undefined) useGuideline = true;
  nv.addGraph(function() {
    var chart;
    chart = nv.models.stackedAreaChart()
                .useInteractiveGuideline(useGuideline)
                .x(function(d) { return d[0] })
                .y(function(d) { return d[1] })
                .color(keyColor)
                  ;

    chart.xAxis
        .tickFormat(d3.format(',.r'));

    chart.yAxis
        .tickFormat(d3.format(',.2f'));

    d3.select('#' + container + ' svg')
          .datum(data)
        .transition().duration(500).call(chart);

    nv.utils.windowResize(chart.update);
```

```
    return chart;
  });
}
```

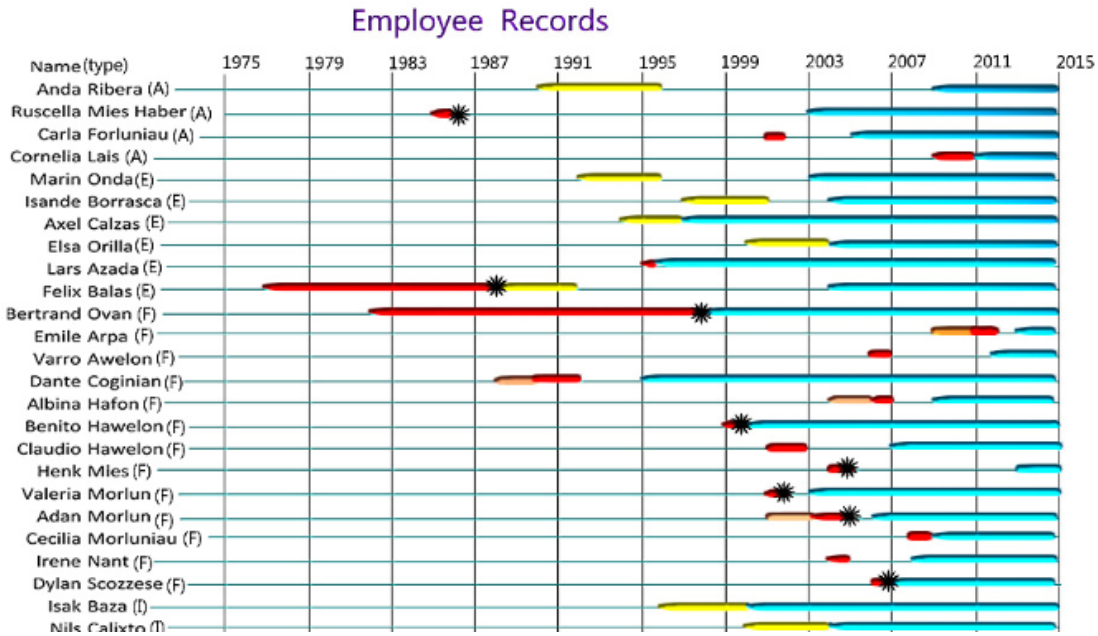
调用 nv.models 中的 stackAreaChart 函数，data 数据中的 key 作为绘图的标识，年份和词频分别作为 x 和 y 值绘制填充状态的 StackArea 图。

7. 部分数据的图表

7.1 对 EmployeeRecord 文件实现可视化

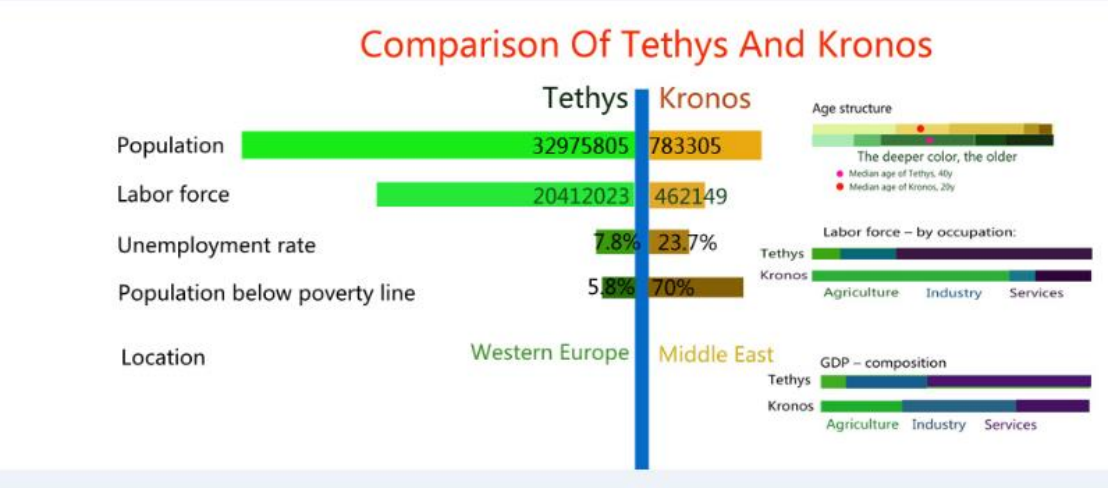
EmployeeRecord 文件是 GAStech 公司的人事档案，里面记载了公司所有成员包括大学，兵役，开始雇佣的时间等多维信息，而且人数又众多。要在有限的空间内将复杂的数据展示出来所以对数据按照对破案的有效性出发进行了筛选，并考虑图表的方式。

图表纵列为人物，对应横行为时间轴，在时间轴上用不同的颜色将不同的事件标出，得到比较直观的图像。



7.2 用成对条形图实现 Kronos 与 Tethys 的对比

数据来源： FACTBOOK-Kronos 与 FACTBOOK-Tethys。由于数据有文字有比例有大数字，比较杂乱，不能很好系统化的可视化。



年龄结构，劳动力在三个产业的分布以及 GDP 中三个产业的比重，这几个都是总量为 1 的比列，所以两个国家的数据用总长相同，颜色相同放在一起比较，数据与长度精确比例，可以比较直观的看出差异

重点难点以及交互模块

1. Timeline 的显示形式

在 timeline 页面中，为了实现信息和 timeline 模块的联动，在页面最下方是时间轴，并将对应的每一份信息嵌入了一个 frame 框架，可通过屏幕左右的箭头来移动要显示的框架，亦可通过点击时间轴上的事件的图标定位到对应的 frame。

2. 设置鼠标动作对应效果

实现交互的主要方式在于对鼠标动作的响应，鼠标动作具体有悬停，点击和移开等。这里主要通过两种方式来完成。

方法一：若对应要操作的对象是通过 d3 绘制的一个 svg，那么它自带如.on("mouseover",function();.on("mouseout",function())；等属性，我们只需要定义对应情况下的 function 函数的动作即可。

方法二：如果对应的操作对象是 html 中的某个元素，比如在设置按钮、设置图片切换以及设置页面跳转等情况的时候，我们可以给对应元素添加 onclick 属性，并对应一个函数，在嵌入的<script>中添加对对应函数的实现即可。

3. 关键词检索的算法

难点一、实现全文章的遍历（数据量大）

解决方式：利用 JAVA 的已有的库将一个目录下的文件的文件名以字符串数组的形式存储在内存中，利用 for-each 语句遍历所有文件，每次取出一个文件中的每一行进行解析。

难点二、无用词语的过滤以及符号的过滤（例如 the, me, you 等）
解决方式：利用 JAVA 的字符串的 spit 函数将文本的单词一个个解析出来。思路：)。对单词的进行简单的处理，去掉开头或者末尾的符号。然后判断它是否在无用词库（开始自己定义无意思词语的列表中）。若不是，则加入容器中。这样全部遍历一遍，所有的关键词都存在了内存中了。

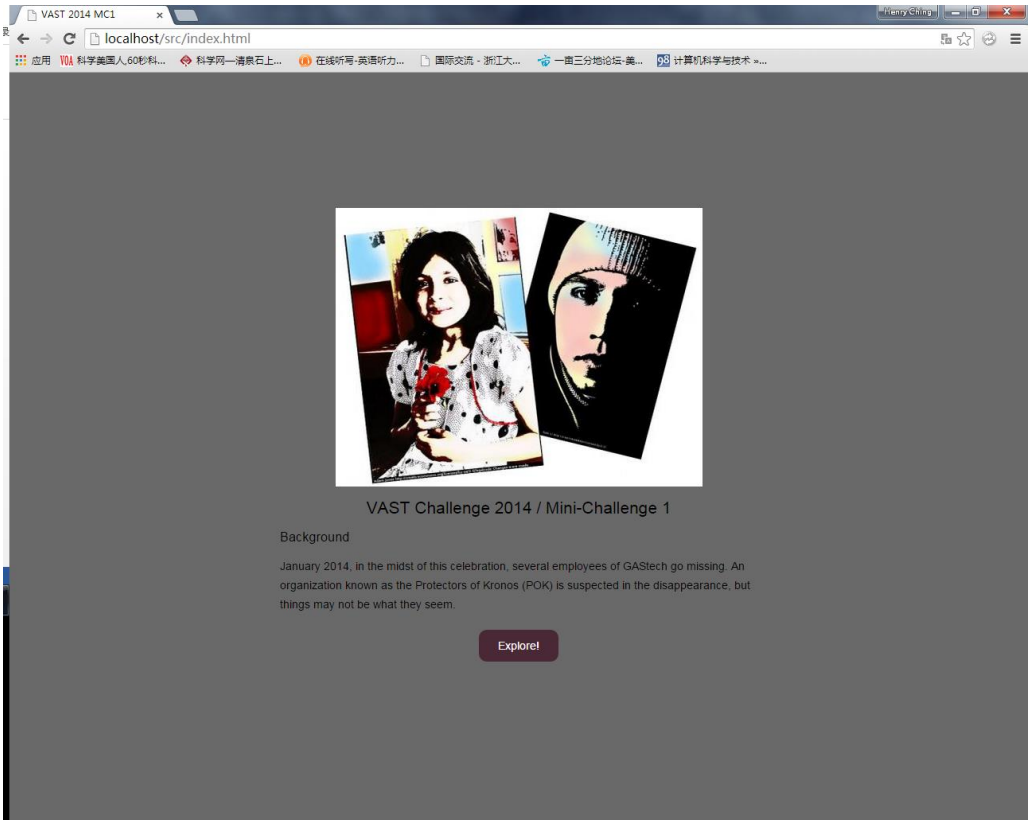
难点三、一句话的识别（多种文本截取混合处理一段文本）
解决方式：先对每行按照句子进行分割（一般是'.'分割），然后在句子中分割出一个个单词，这样就实现显示关键词所在的句子的功能。

成果展示

需要特别说明，由于直接打开 html 不支持对 json 格式文件的读写，所以需要将整个系统文件夹放入本地网络服务器下，键入 localhost/src/index.html 进入主页面

1. Index.html 界面

包含对问题背景的介绍。



点击 Explore 进入对数据的分析。

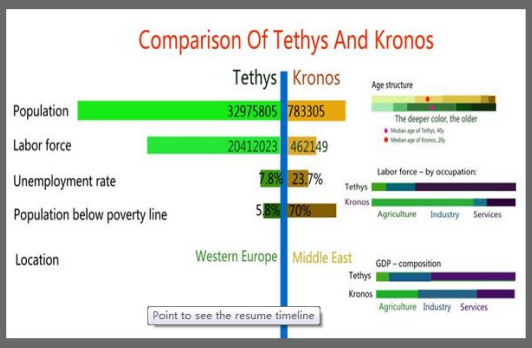
2. Introduce.html

包含了三张静态图表和对应的介绍，鼠标悬停图片显示相关信息，点击图片可实现切换。

Map of Tethys and Kronos

Tethys is in the north west to Kronos, which has two main cities, Abila and Elodis.

POK History

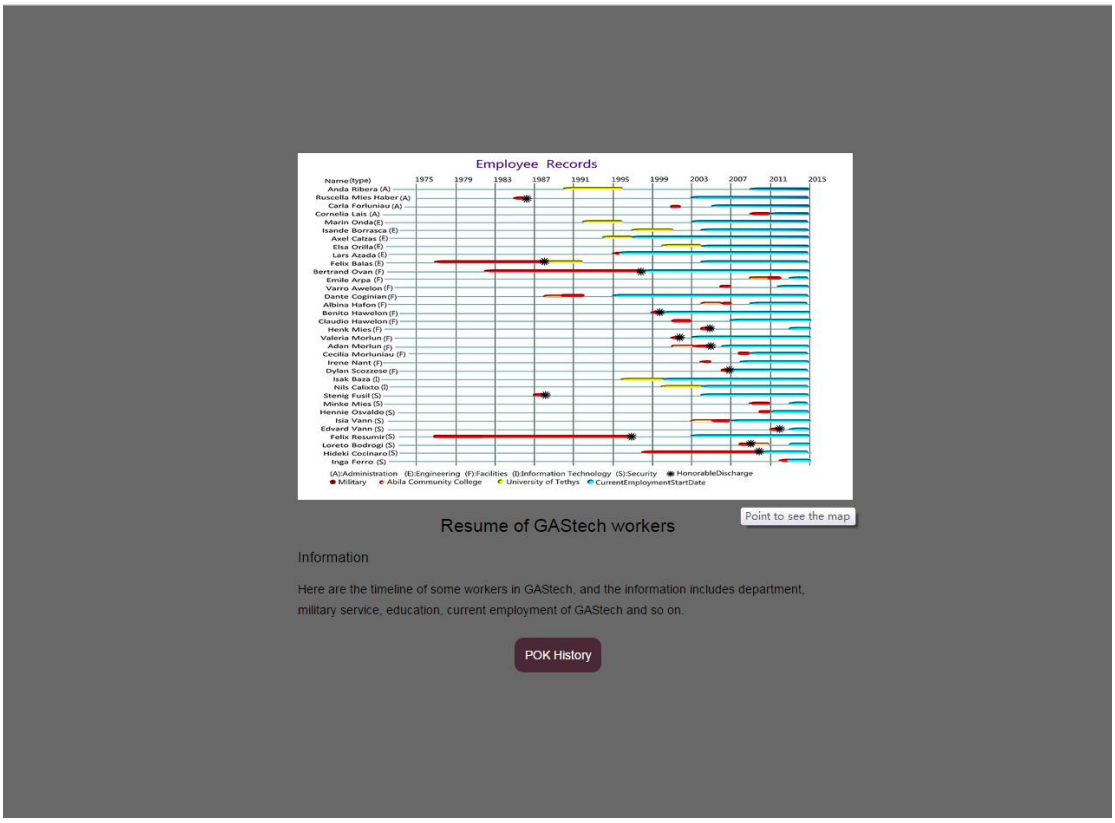


comparison of Tethys and Kronos

Information

It can be inferred from the graph that Kronos' economy and people's quality are much lower than Tethys

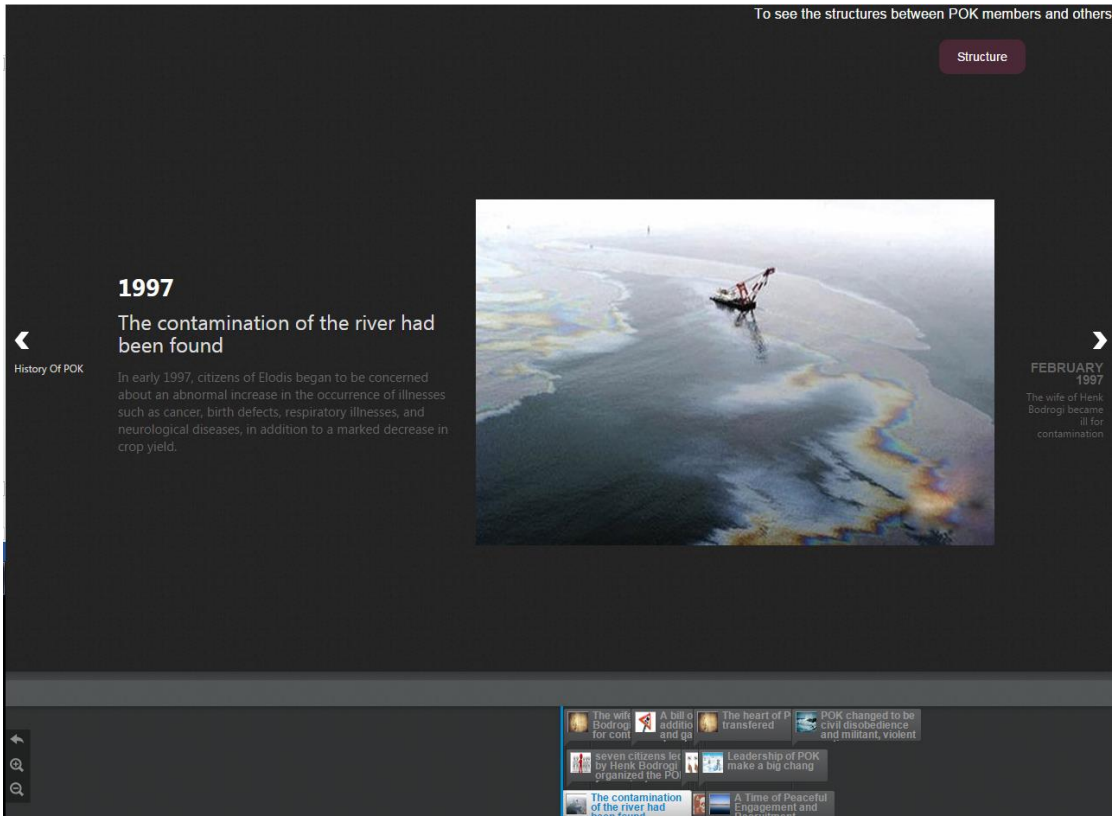
POK History



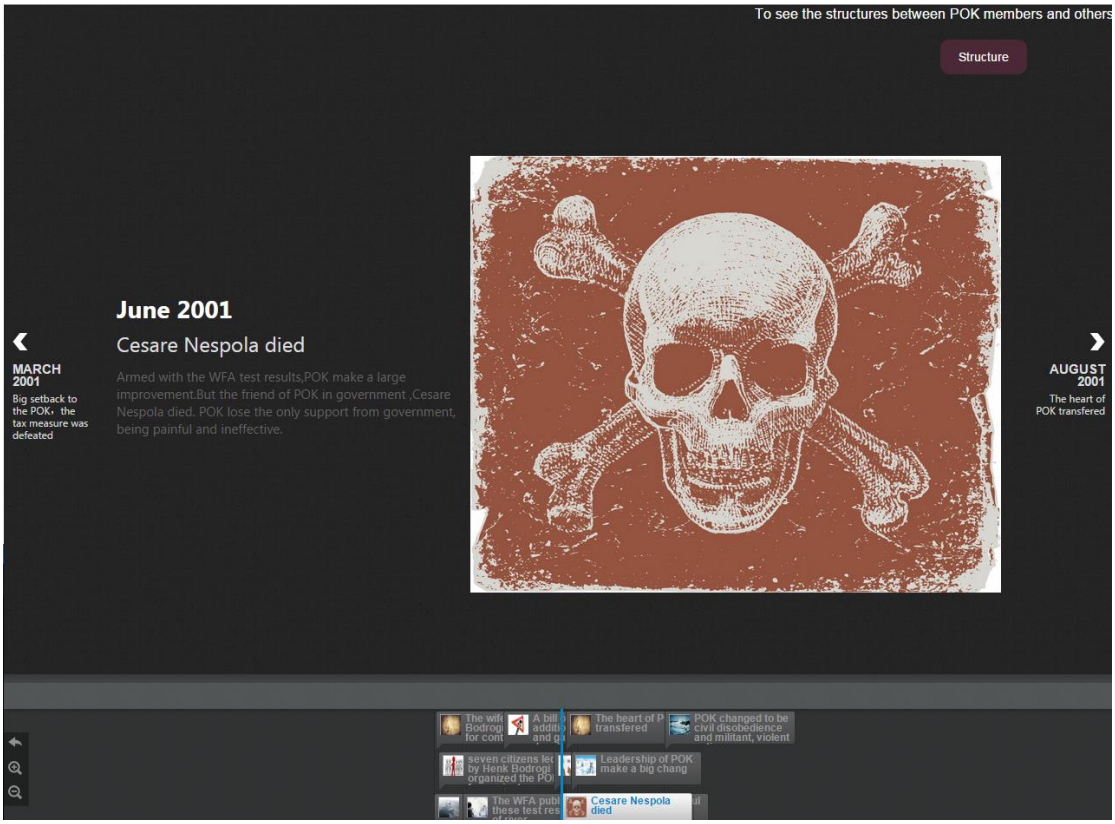
3. Timeline.html

解读 POK 的历史，实现重要信息随时间变化而呈现

可以通过箭头来按照时间顺序浏览具体时间



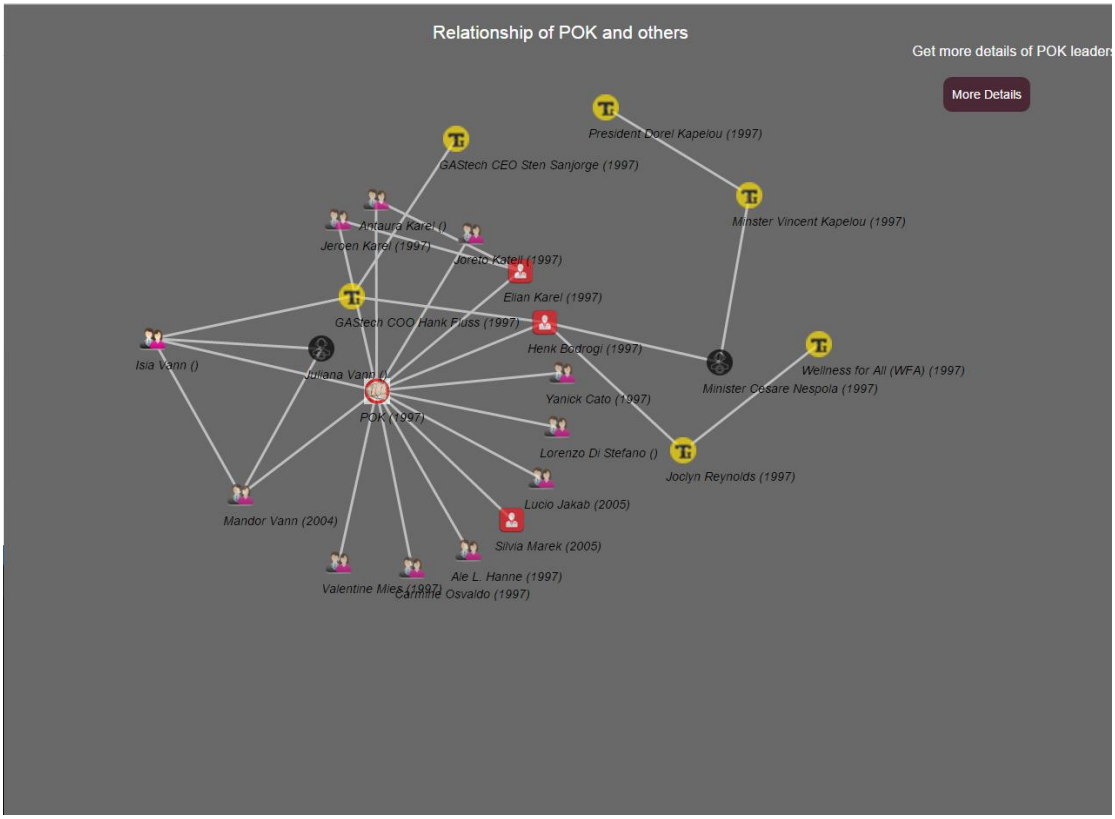
也可以通过点击时间轴，直接定位具体事件



4. Pokstructure.html

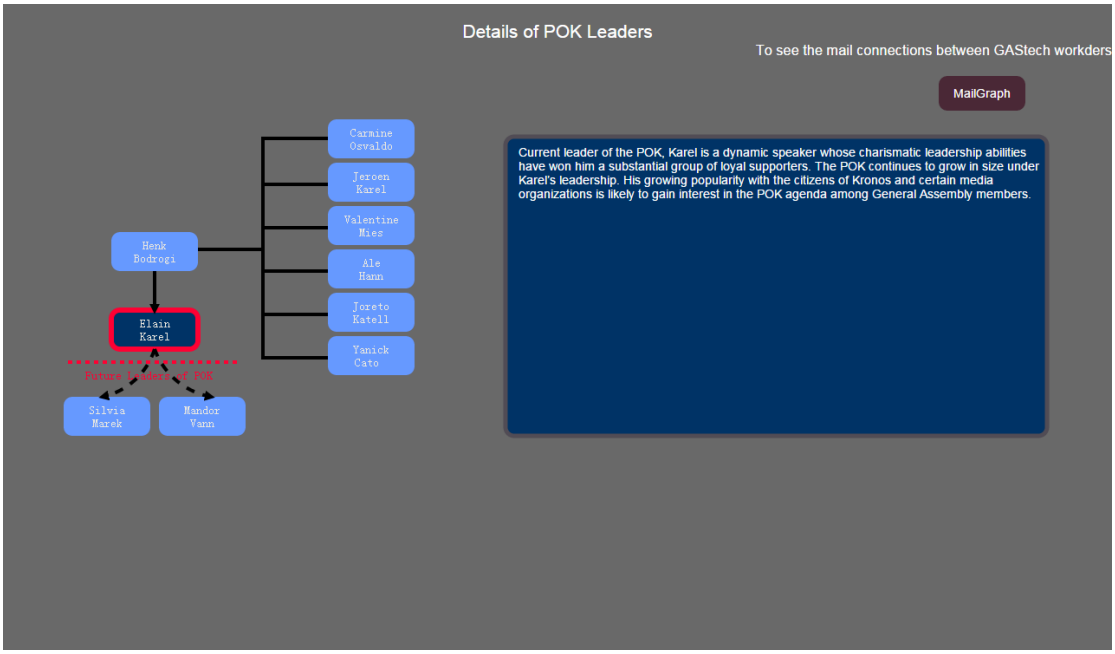
呈现了 POK 成员以及部分 GAStech 员工和政治人物之间的纽带关系，鼠标拖动可以改变力

图的格局



5. Detail.html

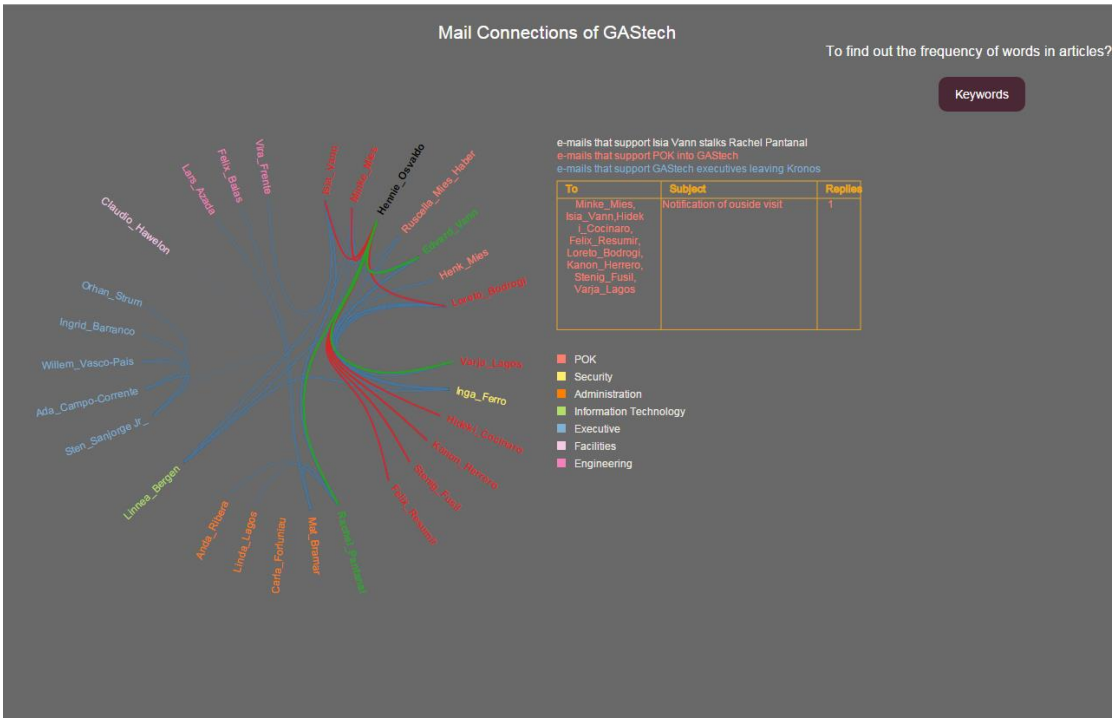
呈现 POK 早期、中期、以及未来可能的领导人关系图，具体人物可显示对它的介绍



6. Mail.html

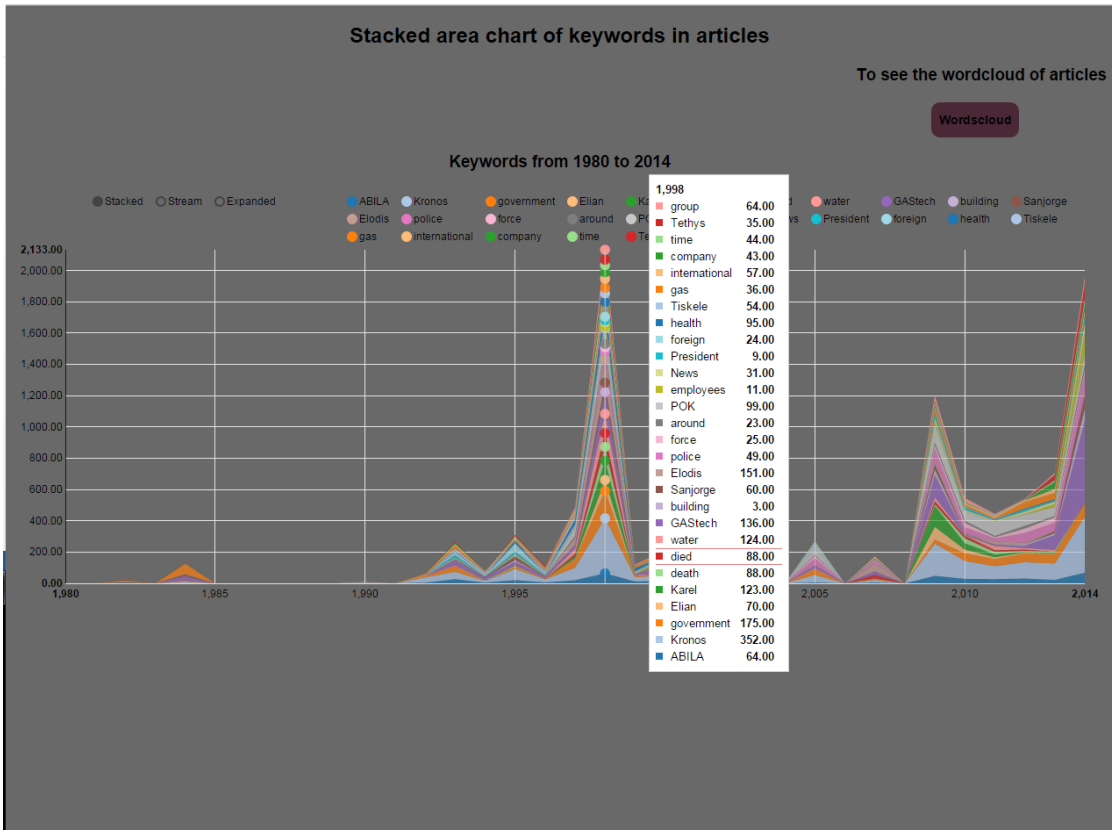
呈现了 GAStech 内部员工的邮件关系，按照 POK、security、administration 等进行人物分类并标以不同颜色。

鼠标悬停可显示对应人物所有的邮件联络，将按照接收者、主题和回复数显示对应的信息。



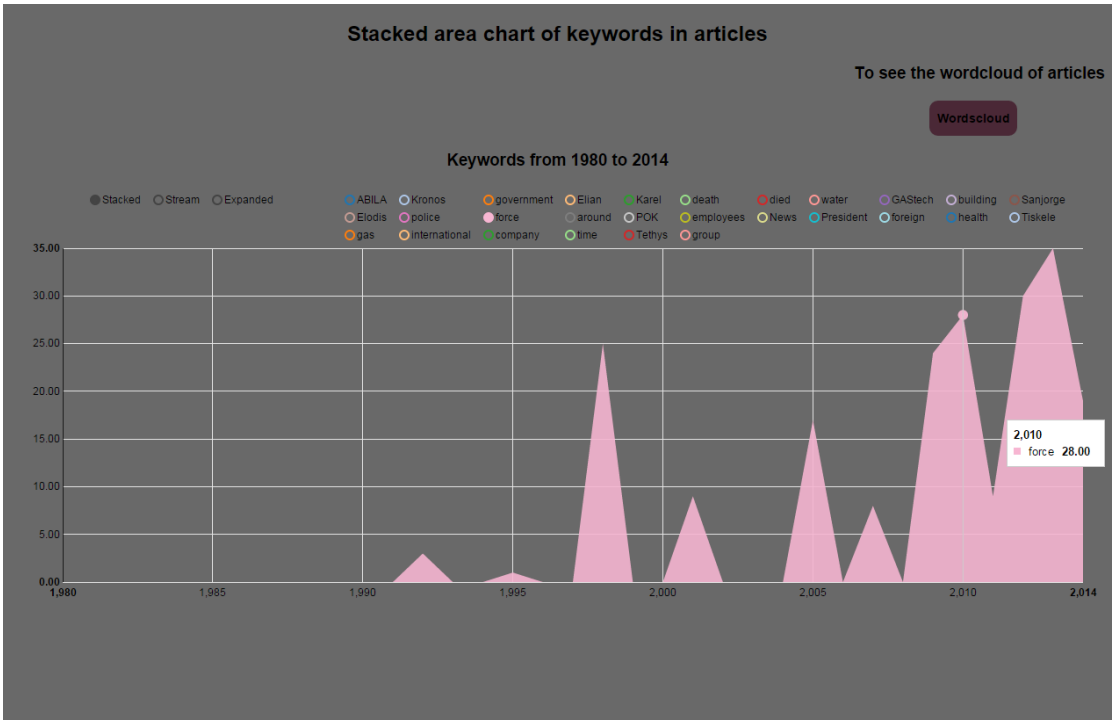
7. Keywords.html 和 wordscloud.html

将 1980 至 2004 年关键词通过 stackarea 图来呈现，可配合 keyword 检索工具来使用。

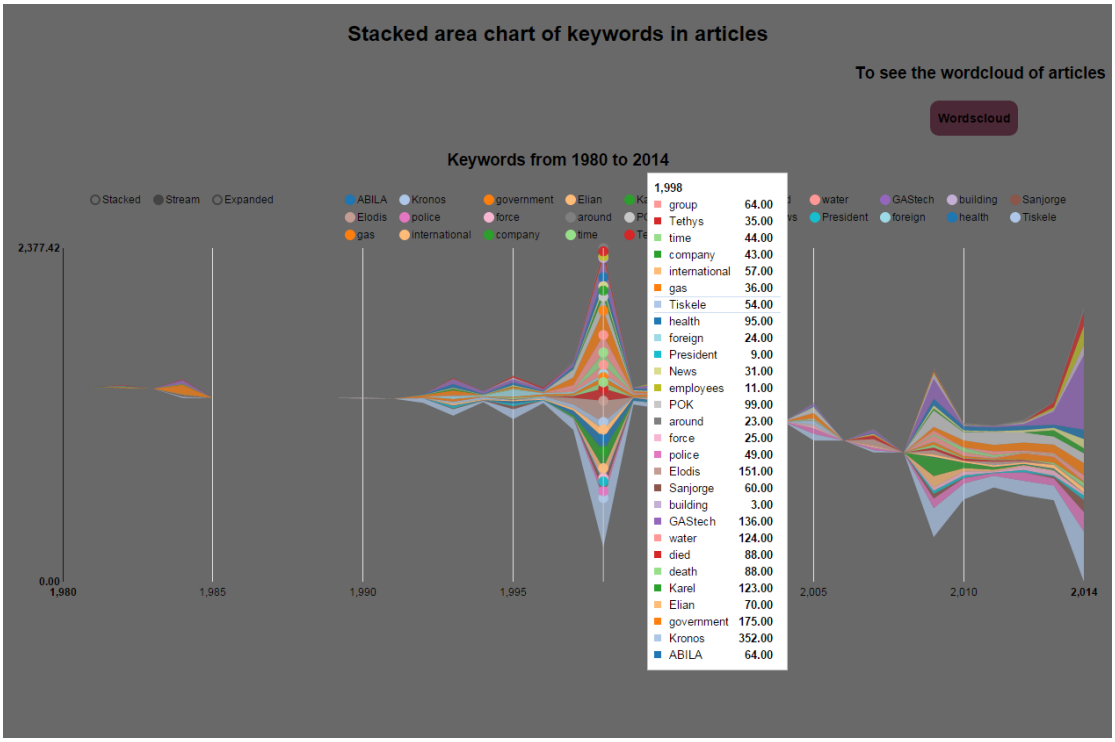


在 stacked 模式下显示的是具体关键词每一年出现的频率，通过引导线可以查看所有词语的概况，也可以点击对应词语或者图像上的区域，来过滤得到我们所需要的某一或某些词语的

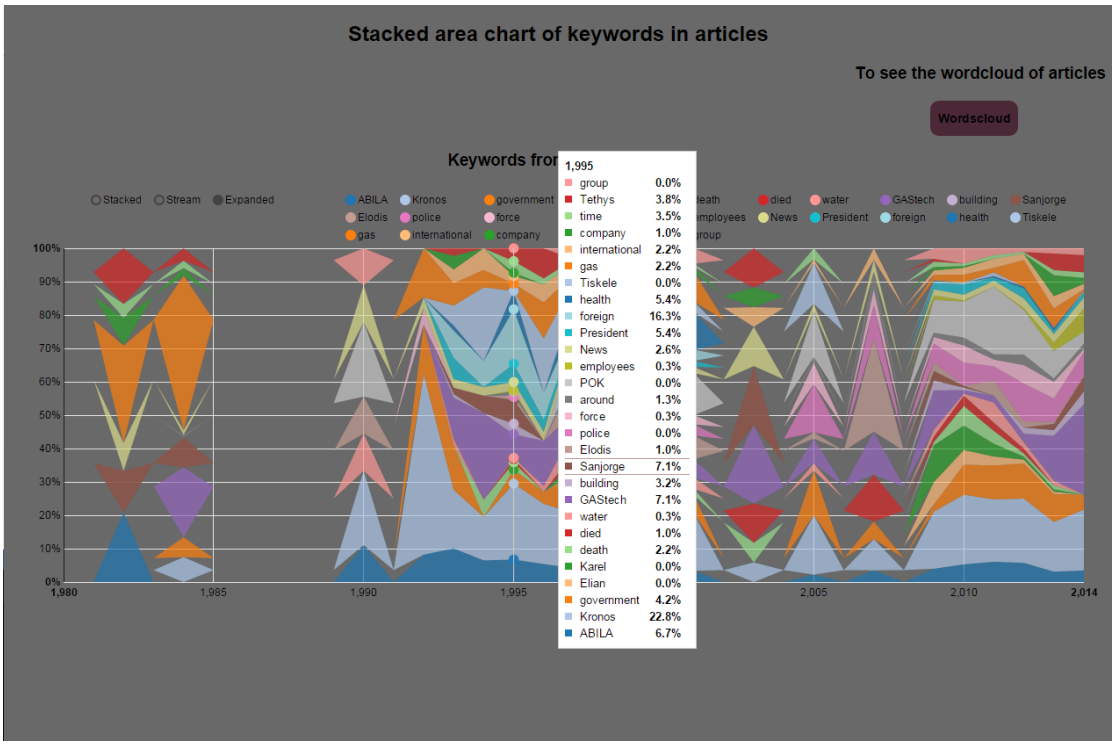
具体图像



切换到 Stream 模式则变为流状图



切换到 Expanded 模式，则显示的是 keywords 每年所占百分比，较上述两图更加清晰。



Wordscloud.html 则呈现了对应的词云图像

8. Keywords 检索软件

输入任意词语检索，可以显示出其总出现的频率并显示出在每篇文章中出现的句子

例：检索”death”

```
Death
Result:
109.txt: * The death rate and the incidence of the chronic disease increased more than 178% of the former years.
158.txt: IN MEMORY:  THE DEATH OF ELIAN KAREL, A LATER YEAR.
213.txt: ABILA, Kronos - With the third anniversary of Elian Karel's death approaching - former member of the Protectors of Kronos - the question is: who are
 the protesters?  .
257.txt: IN REMEMBRANCE:  THE DEATH OF ELIAN KAREL, A YEAR LATER.
286.txt: IN REMEMBRANCE:  THE DEATH OF ELIAN KAREL, A YEAR LATER .
300.txt: ABILA, Kronos - with the third anniversary to approach of the death of Elian Karel - previous member of the protectors of Kronos - the question is:
 who are the demonstrators?  But not only who are they, which and because they are protesting? .
523.txt: IN THE MEMORY:  The DEATH Of ELIAN KAREL, ONE YEAR AFTERWARDS.
54.txt: ABILA, Kronos - With the third anniversary of Elian Karel's death approaching - former member of the Protectors of Kronos - the question is: who are
 the protesters? But not only who are they, what and why are they protesting? .
566.txt: NO PROGRESS ON INVESTIGATION OF POK LEADER DEATH.
709.txt: ABILA, Kronos - with the third anniversary of the death of Elian Karel closer coming - former member of the guards of Kronos - the question is: who
 are protesters? But not only who she, what are and why do they protest? .
770.txt: IN REMEMBRANCE:  THE DEATH OF ELIAN KAREL, A YEAR LATER.
833.txt: KAREL DEATH ANNIVERSARY.
num: 12
```

编程体会

秦昇：

1. **如何领导一个团队**

   作为这次团队项目的组长，我深深地感到了比别人要多很多的压力和责任感，无论是统一讨论的思路和结论还是编写代码，都需要我做很多的统筹工作，觉得自己做的好的地方主要是能够组织起整个团队的工作，包括在第一次展示后觉得思路和作业的要求有所偏差之后能够尽快改进并且开始第二次的开发。但是也存在不少的问题，对很多工作的细节没有安排好，会导致大家提交给我的结果不符合要求，我还需要很多的时间来整合。尽管自己负担了较多的工作，但是还是要得益于整个团队的齐心协作，也让我们在领略可视化魅力的同时学会了团队协作去开发软件。

2. **编写代码能力、实现数据交互可视化的能力**

   首先这一门课让我收获了很多，无论是对数据进行可视化的方法和思想，还是亲自动手实践来完成一个个或小或大的项目的能力，都使我受益匪浅。此外，大作业的核心思想——可视化的交互所展现的魅力也让我领略了很多。

   数据可视化的方法有很多，但是越来越多的人愿意去选择可交互的方式，这是因为它对用户更加友好，也更能够表现出数据的特征，更加方便地去从宏观和具体的角度对数据进行解析。设想如果将所有国家的信息改变为

一张一张单独的折线图，不仅将工作量加大了上百倍，而且每一张折线图也没有生命力，工作量也更加复杂。

而有了交互我们可以轻松选择我们需要看的信息，我们可以将折线图的细节变得更加清晰，数据传递给我们的信息也更加明晰和完整。

3. **要善于学习和借鉴，并加以创造**

在编写大作业的过程中大部分时候我需要在网上查阅相关资料以及其他人的经验总结，这在近期的学习中起到了很大的作用，在学习一项新的知识的时候很多时候无异于大海捞针，善于借鉴他人的经验，可以减少很多不必要的弯路。
在学习的过程中我们同样进行着创造，无论是对他人的经验进行组合，还是加入自己的思考，都是创造的手段，这样不仅加深我们对知识的理解，也在动手的同时提升了编程、代码能力。

胡冯钦：

信息可视化是应用日益广泛的一种数据、信息显示方式，将技术和艺术、创意紧密结合，很有趣，也很吸引人。完成这个 Mini Challenge 1 花费了不少的时间和精力，Challenge 本身的故事背景设置就很有趣，提供的数据也丰富多样，而且量不小。一点点抽丝剥茧，提取有用信息，或对大量文本做处理，并创想新颖的图表形式去展现是个很有趣的过程。但是在实际完成过程中，由于缺乏经验，而且没有足够的范例和指导，将精力和重点更多地放在了挖掘信息、分析等，而没有花足够时间设计更漂亮精致的可视化形式。在答辩后，得到了老师和助教的颇多建议，并重新设计、整合，虽然还是略显粗糙，但已有一定体系的样子。希望以后若有需要处理数据，或需要用可视化方式帮助理解和表达时，能用上今时今日所学，并有所精进。

黄靖：

主要负责文本检索器的制作，顺便担负起主要数据处理的重任，所以上传的代码中应该会附带很多数据处理的函数。在这里顺便理一下，Email 关系的 json 格式文件主要通过解析 csv 文件各个部分，解析出发件人的姓名，收件人的姓名，发信的日期，信件的具体内容。存储到相应的数据结构中，然后再通过写文件的方式按照 json 的嵌套格式写入文件。还有关键词在每年出现的频率的统计的，也是通过在文本检索器的基础上统计出所

有有效关键词在每年出现的次数，然后封装在相应数据类里面，按照可视化制作需要的 json 格式写入到文件中。以及截取出每篇 article 的内容，然后按照 csv 的格式写入到文件中（这个最后没有采用实际作图交互）。
参与了部分 JavaScripe 的编程。精通 JavaScripe 的编程的确需要不少的功夫，在这么短的时间内，只能算上勉强摸到 JavaScripe 的一角。要熟悉 JavaScripe 的编程需要不断地练习（主要是前端的训练），才能达到游刃有余的境界。

张凯：

一开始接触可视化方面的东西还是很茫然的，之前没有接触过 D3。在完成任务的过程中，渐渐体会到了他给可视化带来的巨大便利。有着丰富的模型样板，确实使本来平淡的数据变得非常的炫。应该说通过这样一个大程，我学习到了很多，不仅仅局限于编程方面，也包括团队合作方面。能按时完成任务跟同组同学的努力是分不开的。

具体分工

秦昇：

项目整体框架设计、html 网页的编写整合
POK 历史 timeline、mail 往来图、keywords 的 stackarea 图设计。
POK 历史解读、邮件、POK 关系图数据处理
总体说明文档的撰写和整合

胡冯钦：

POK 结构力图、POKleader 简介图设计
个人工作文档撰写

黄靖：

关键词检索软件设计
Email、pokstructure、stackarea 图数据处理
个人工作文档撰写

张凯：

Tethy Kronos 对比图、Resume 解读和作图
Timeline 关键事件总结和配图
个人工作文档撰写