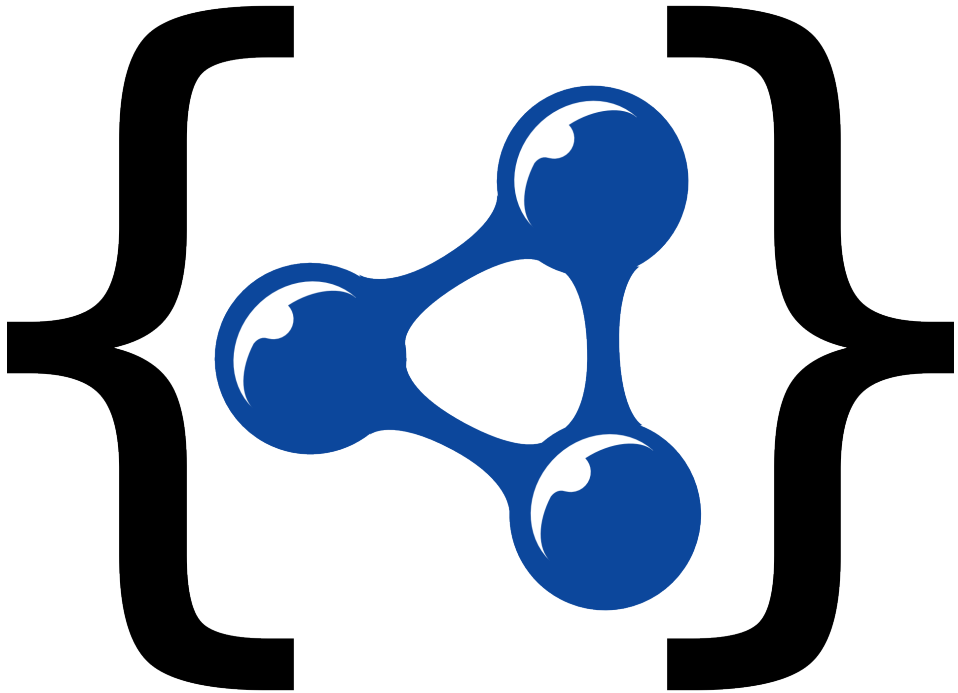
XML 原理与应用

Summer X

RUC

CH10 XML 的应用与挑战——SVG & JSON



本章学习目标

- 了解 SVG 的特点
- 能够基于 SVG 进行简单的图形表示
- 了解 d3.js 的功能特点
- 掌握 JSON 的数据结构和类型
- 能够通过 JavaScript 解析 JSON 字符串

目录

10.1 概述

- SVG
  - 可缩放矢量图形（Scalable Vector Graphics）
  - 1999 年推出
- JSON
  - JavaScript Object Notation
  - 尤其适合作为数据传输格式
- SVG、JSON、XML 是 Web 技术体系的重要组成部分

PNG 与 SVG 文件缩放对比

	
PNG 文件	SVG 文件
	
放大后的 PNG 文件	放大后的 SVG 文件

10.2 SVG

- 通过 XML 定义线段、圆形、矩形等形状

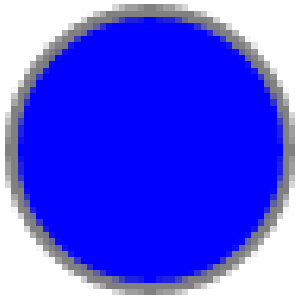
```
1 <svg xmlns="http://www.w3.org/2000/svg" width="50" height="50">
2   <circle cx="25" cy="25" r="22" fill="blue" stroke="gray"
      stroke-width="2"/>
3 </svg>
```

SVG 示例 I

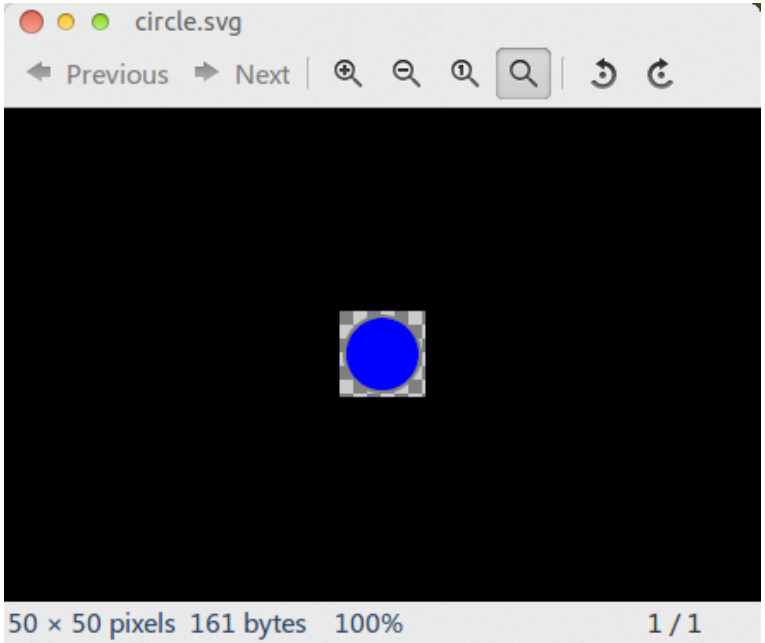
Code 1: ”circle.html”

```
1  <!DOCTYPE html>
2  <html>
3    <head>
4      <title>SVG Circle</title>
5    </head>
6    <body>
7      <svg width="50" height="50">
8        <circle cx="25" cy="25" r="22" fill="blue" stroke="gray"
       stroke-width="2"/>
9      </svg>
10   </body>
11 </html>
```

SVG 示例 II



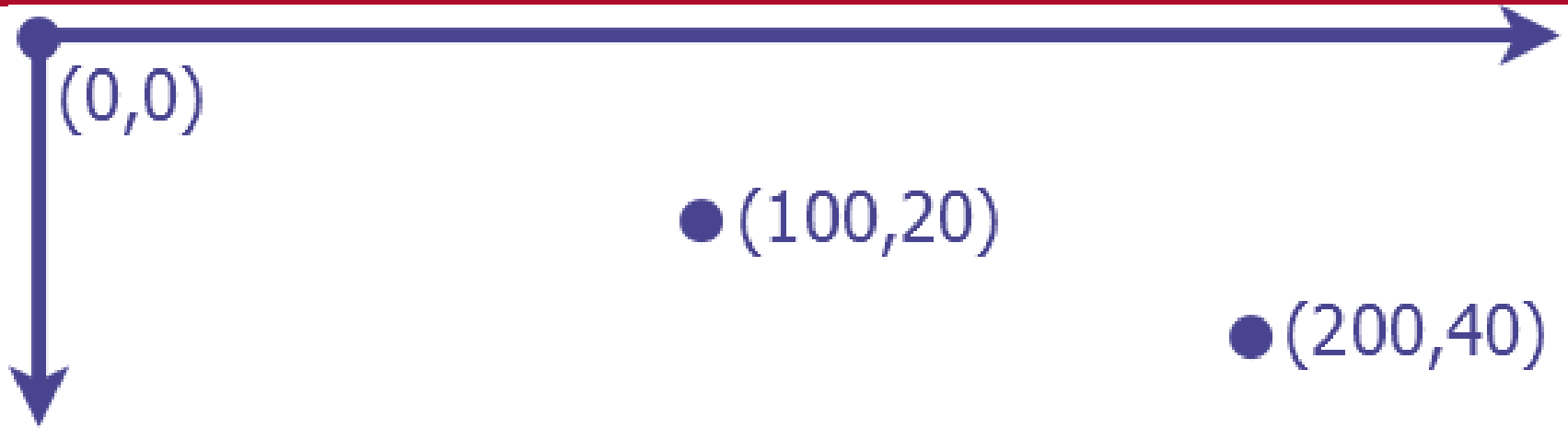
Linux Imagte Viewer



10.2.1 SVG 的基本形状 I

- 基本形状
  - 矩形：rect
  - 圆形：circlue
  - 椭圆：Ellipse
  - 线条：line
  - 文本：Text
- 坐标系

10.2.1 SVG 的基本形状 II



绘制矩形 I

- 代码

```
<rect x="0" y="0" width="300" height="50"/>
```

- 效果

绘制矩形 II

- 代码

```
<rect x="0" y="0" width="300" height="50" rx="15" ry="15"/>
```

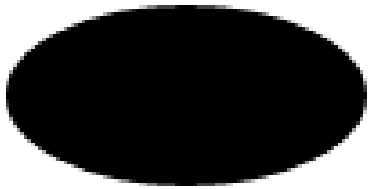
- 效果

绘制椭圆 I

- 代码

```
<ellipse cx="250" cy="25" rx="50" ry="25"/>
```

- 效果



绘制线条 I

- 代码

```
<line x1="0" y1="0" x2="350" y2="50" stroke="black"/>
```

- 效果

绘制文本 I

- 代码

```
<text x="250" y="25" font-family="华文隶书" font-size="30"
    fill="navy">可缩放矢量图形</text>
```

- 效果

10.2.2 SVG 的样式设置

- 默认样式为黑色填充、无 stroke
- fill
  - 填充色，取值为有效的 CSS 颜色值，如颜色名称 red、blue，或者 RGB、RGBA 值。
- stroke
  - 线条的颜色值
- stroke-width
  - 数值，通常采用像素单位
- opacity
  - 透明度，介于 0.0 到 1.0 之间的数值，0.0 表示完全透明，1.0 表示完全不透明
- font-family、font-size
  - 应用于文本

样式关联方式

- 内联方式

```
<circle cx="50" cy="50" r="30" fill="yellow" stroke="orange"
    stroke-width="10"/>
```

- CSS 样式属性关联方式

```
<circle cx="50" cy="50" r="30" class="pumpkin"/>
```

  - 需要设置 pumpkin 样式

- 效果

示例网页 I

```
<!DOCTYPE html>
<html>
  <head>
    <title>SVG Style</title>
    <style>
      .pumpkin {
        fill: yellow;
        stroke: orange;
        stroke-width: 10;
      }
    </style>
  </head>
  <body>
    <svg width="500px" height="200px">
      <circle cx="50" cy="50" r="30" class="pumpkin"/>
    </svg>
```

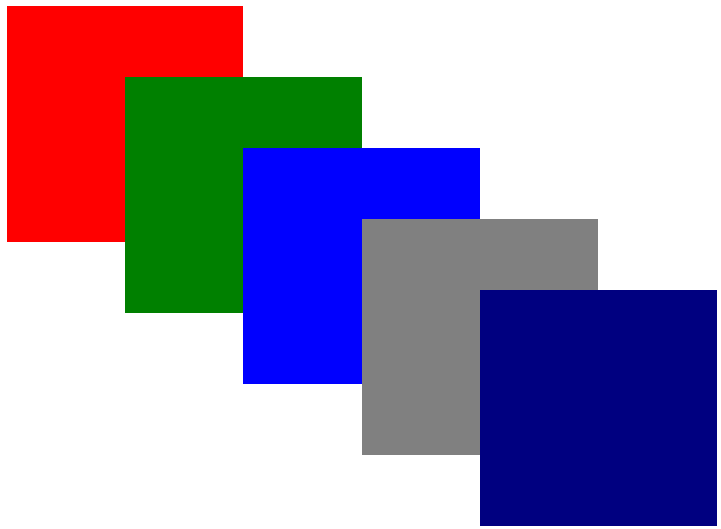
示例网页 II

```
17    </body>
18 </html>
```

10.2.3 SVG 的层与重叠

- 根据绘制的先后顺序予以覆盖

```
1 <rect x="0" y="0" width="50" height="50" fill="red"/>
2 <rect x="25" y="15" width="50" height="50" fill="green"/>
3 <rect x="50" y="30" width="50" height="50" fill="blue"/>
4 <rect x="75" y="45" width="50" height="50" fill="gray"/>
5 <rect x="100" y="60" width="50" height="50" fill="navy"/>
```



10.2.4 SVG 的透明度

- 使用带 alpha 的 RGB 颜色函数 rgba()
- 设置 opacity 属性值。

rgba() I

- 代码：

```
1 <circle cx="50" cy="50" r="40" fill="rgba(0, 150, 150, 1.0)"/>
2 <circle cx="100" cy="50" r="40" fill="rgba(0, 0, 255, 0.75)"/>
3 <circle cx="150" cy="50" r="40" fill="rgba(0, 255, 0, 0.5)"/>
4 <circle cx="200" cy="50" r="40" fill="rgba(255, 255, 0, 0.6)"/>
5 <circle cx="250" cy="50" r="40" fill="rgba(255, 0, 0, 0.3)"/>
```

- 显示效果：

rgba() II

- 代码：

```
<circle cx="50" cy="50" r="40" fill="rgba(0, 150, 150, 1.0)"
    stroke="rgba(0, 255, 0, 0.25)" stroke-width="10"/>
<circle cx="100" cy="50" r="40" fill="rgba(0, 0, 255, 0.75)"
    stroke="rgba(0, 0, 255, 0.25)" stroke-width="10"/>
<circle cx="150" cy="50" r="40" fill="rgba(0, 255, 0, 0.5)"
    stroke="rgba(0, 0, 255, 0.25)" stroke-width="10"/>
<circle cx="200" cy="50" r="40" fill="rgba(255, 255, 0, 0.6)"
    stroke="rgba(255, 0, 0, 0.25)" stroke-width="10"/>
<circle cx="250" cy="50" r="40" fill="rgba(255, 0, 0, 0.3)"
    stroke="rgba(55, 255, 0, 0.25)" stroke-width="10"/>
```

- 显示效果：

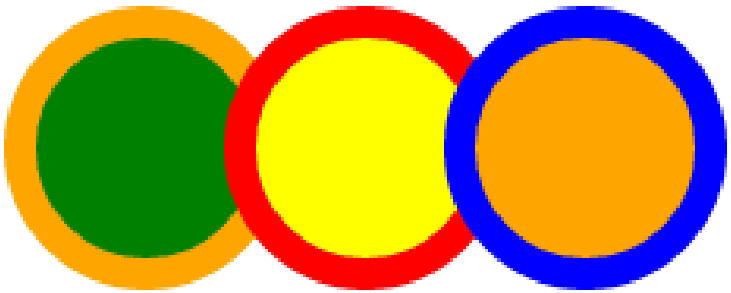
rgba() III

opacity I

- opacity 设置为 1.0 表示完全不透明，为默认值

```
1 <circle cx="50" cy="50" r="40" fill="green" stroke="orange"
      stroke-width="10" opacity="1.0"/>
2 <circle cx="120" cy="50" r="40" fill="yellow" stroke="red"
      stroke-width="10"/>
3 <circle cx="190" cy="50" r="40" fill="orange" stroke="blue"
      stroke-width="10"/>
```

- 显示效果：

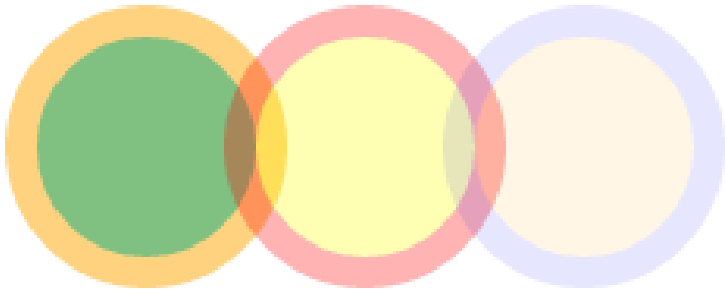


opacity II

- 代码：

```
1 <circle cx="50" cy="50" r="40" fill="green" stroke="orange"
      stroke-width="10" opacity="0.5"/>
2 <circle cx="120" cy="50" r="40" fill="yellow" stroke="red"
      stroke-width="10" opacity="0.3"/>
3 <circle cx="190" cy="50" r="40" fill="orange" stroke="blue"
      stroke-width="10" opacity="0.1"/>
```

- 显示效果：

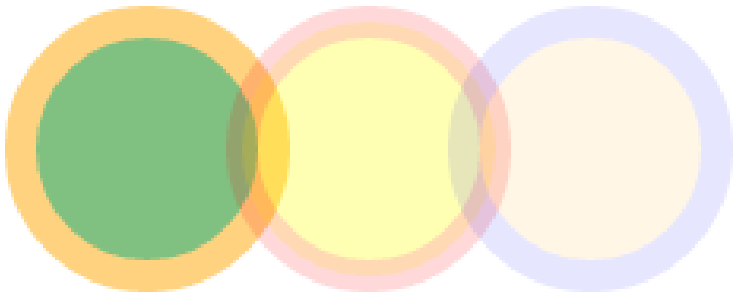


同时设置 rgba 与 opacity I

- 代码：

```
1 <circle cx="50" cy="50" r="40" fill="green" stroke="orange"
      stroke-width="10" opacity="0.5"/>
2 <circle cx="120" cy="50" r="40" fill="yellow"
3     stroke="rgba(255, 0, 0, 0.5)" stroke-width="10" opacity="0.3"/>
4 <circle cx="190" cy="50" r="40" fill="orange" stroke="blue"
      stroke-width="10" opacity="0.1"/>
```

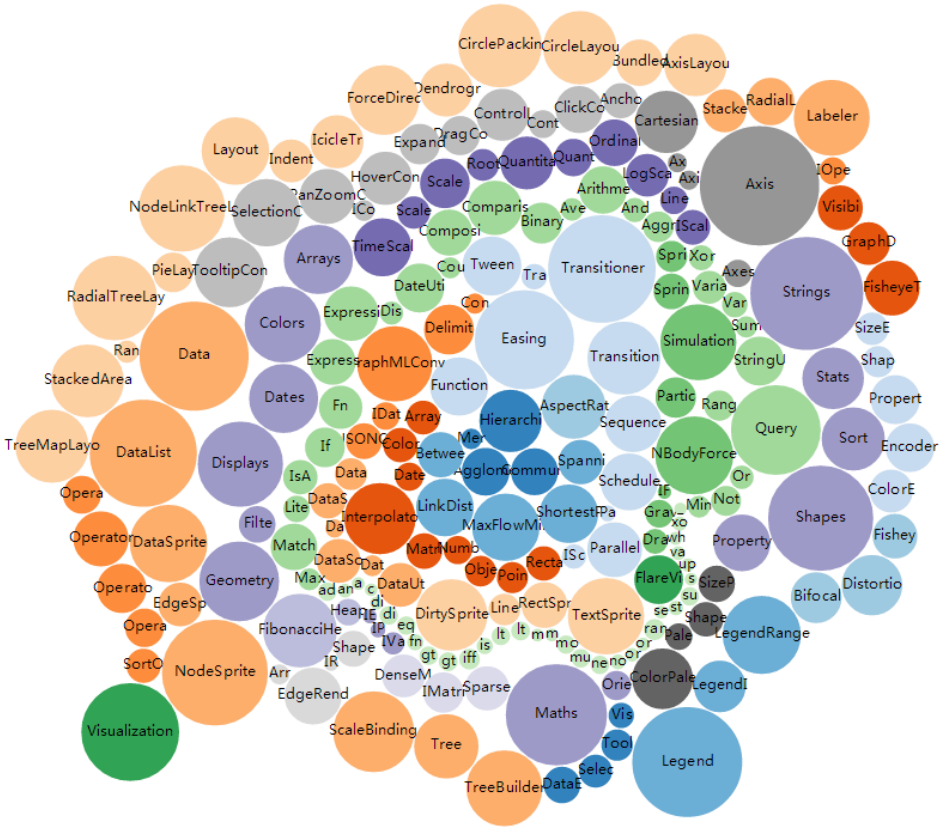
- 显示效果：



10.2.5 d3.js

- d3
  - Data-Driven Documents 的缩写
  - Data 表示用户提供的数据
  - Documents 代表可以被浏览器解析呈现的文档，
  - 支持 SVG

效果图 I


CirclePackin
CircleLayou
Bundled
AxisLayou
ForceDirec
Dendrogr
Stacke
RadialL
Labeler
Layout
Indent
IcicleTr
Expand
DragCo
ControlL
Cont
ClickCo
Ancho
Cartesian
Ordinal
Quantile
Quant
Root
HoverCon
Scale
LogSca
Axis
Visibi
NodeLinkTreeL
SelectionC
Arithme
Line
Comparis
Ave
And
Binary
Composi
AggrIScal
GraphD
Arrays
TimeScal
Tween
Transitioner
Spri
Xor
PieLayTooltipCon
DateUti
Tra
Axes
Strings
FisheyeT
RadialTreeLay
Colors
Expressi
Delimit
Easing
Varia
Var
SizeE
Data
Transition
Simulation
StringU
Shap
StackedArea
Dates
Function
Stats
Propert
Fn
IDat
Array
Hierarchi
AspectRat
Sequence
Partic
Rang
Query
Sort
Encoder
TreeMapLayo
DataList
Displays
If
Data
Date
Betwee
Spanni
NBodyForce
Schedule
Or
ColorE
Opera
IsA
DataS
Interpolato
LinkDist
ShortestPa
Min
Not
Shapes
Fishey
Operator
DataSprite
Filte
Match
Da
MaxFlowMi
Gray
Parallel
Property
Geometry
DataSc
Dat
Numb
Recta
ISc
Operato
Max
DataUt
Poin
FlareVi
SizeB
Distortio
Bifocal
EdgeSp
Hear
DirtySprite
Line
RectSpr
TextSprite
Shape
Opera
FibonacciHe
Shape
IVa
fn
LegendRange
SortO
NodeSprite
Arr
IR
DenseM
gt
iff
ColorPale
LegendI
EdgeRend
IMatr
Sparse
Maths
Vis
Visualization
ScaleBinding
Tree
Tool
Legend
TreeBuilder
DataE
Selec

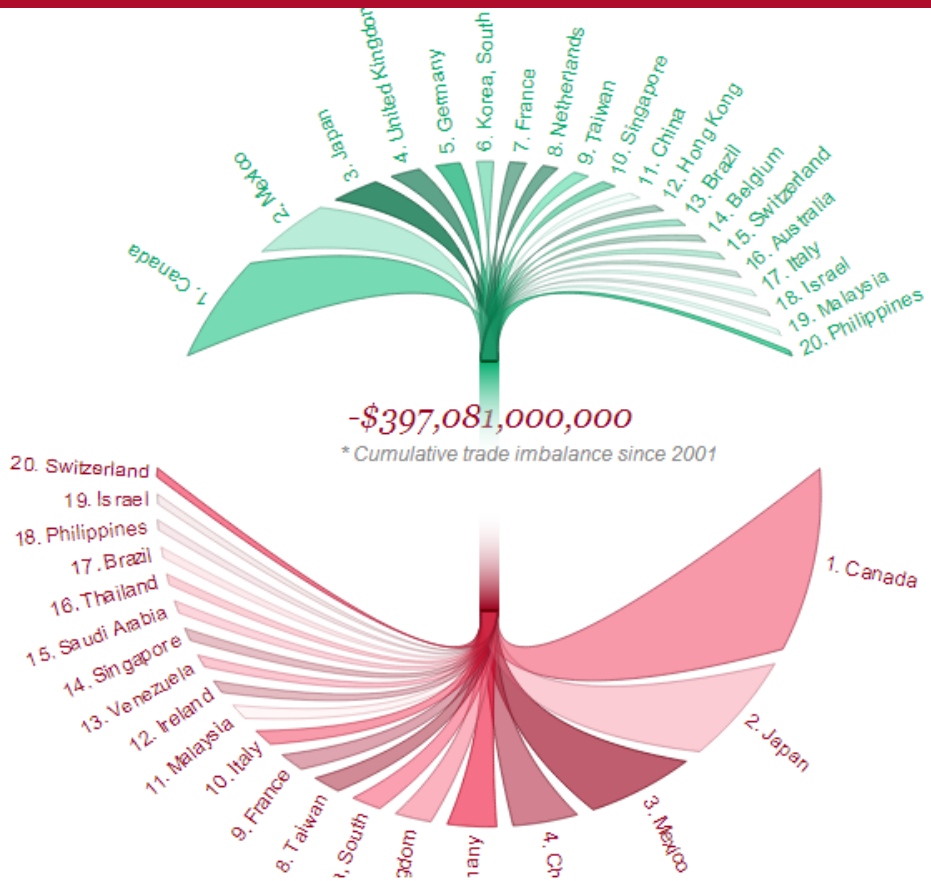
效果图 II


Easing
Transitio
Display
Geometr
Sort
Maths
Stats
Shapes
Colors
Strings
NBodyF
Simula
Query
DirtyS
Graph
Axis
Legend
Visualiz
Circle
Data
DataList
TreeBu
DataS
ScaleBi
Aspec
Aggl

效果图 III

toy
new
day
put
"How
tell
rake
mess
find
shake
big
Now
little
net
fear
wish
cake
ship
kites
"Why
box
sat
sit
get
fast
Make
Thing
two
One
go
house
do"
hall
hold
bit"
sun
good
let
hear
know
fan
fish
pot
cat
"Oh
"I
cup
"Look
that"
went
top
fall
like
things
hat
"Now
bump
ran
red
wet
"That
look
mother
"Two"
"They
came
Sally
hook
shut
pick
saw
fun
away
game
milk
play
ball
call
books
lots
hands
"No
gown
tricks
show
cold
"Do
"Have
"With
dish
shame
wall
want
another

效果图 IV


1. Canada
2. Mexico
3. Japan
4. United Kingdom
5. Germany
6. Korea, South
7. France
8. Netherlands
9. Taiwan
10. Singapore
11. China
12. Hong Kong
13. Brazil
14. Belgium
15. Switzerland
16. Australia
17. Italy
18. Israel
19. Malaysia
20. Philippines
-$397,081,000,000
* Cumulative trade imbalance since 2001
20. Switzerland
19. Israel
18. Philippines
17. Brazil
16. Thailand
15. Saudi Arabia
14. Singapore
13. Venezuela
12. Ireland
11. Malaysia
10. Italy
9. France
8. Taiwan
1. Canada
2. Japan
3. Mexico

如何引用 d3.js 进行测试

- 下载最新版本的 d3.js
- 在网页中引用 d3.js
- 在 Web Server 中测试（可选）

基于 d3.js 绘制 SVG 图形 I

- 创建 svg 元素，设置属性

```
var svg = d3.select("body").append("svg");
svg.attr("width", 500);
svg.attr("height", 200);
```

- 采用链式语法改写

```
var svg = d3.select("body")
        .append("svg")
        .attr("width", 500)
        .attr("height", 200);
```

- 继续增加若干圆形

基于 d3.js 绘制 SVG 图形 II

```
var dataset = [ 20, 15, 10, 5, 10, 15, 20 ];
var circles = svg.selectAll("circle")
        .data(dataset)
        .enter()
        .append("circle");
```

- 继续指定图形的位置和大小

```
circles.attr("cx", function(d, i) {
          return (i * 50) + 25;
        })
        .attr("cy", 50)
        .attr("r", function(d) {
          return d;
        });
```

基于 d3.js 绘制 SVG 图形 III

- 代码中 d 表示传入函数的绑定数据，i 代表图形顺序编号，起始编号为 0

- 显示效果

- 继续指定圆的样式

```
.attr("fill", "orange")
.attr("stroke", "red")
.attr("stroke-width", function(d, i) {
    return d/3;
});
```

基于 d3.js 绘制 SVG 图形 IV

- 显示效果

代码清单 I

```
1  <!DOCTYPE html>
2  <html>
3    <head>
4      <meta charset="utf-8" />
5      <title>D3 Circles</title>
6      <script type="text/javascript" src="d3/d3.v3.js"></script>
7    </head>
8    <body>
9      <script type="text/javascript">
10       var svg = d3.select("body")
11               .append("svg")
12               .attr("width", 500)
13               .attr("height", 200);
14       var dataset = [20, 15, 10, 5, 10, 15, 20];
15       var circles = svg.selectAll("circle")
16                        .data(dataset)
```

代码清单 II

```
17                  .enter()
18                  .append("circle");
19
20          circles.attr("cx", function(d, i) {
21              return (i * 50) + 25;
22          })
23          .attr("cy", 50)
24          .attr("r", function(d) {
25              return d;
26          })
27          .attr("fill", "orange")
28          .attr("stroke", "red")
29          .attr("stroke-width", function(d, i) {
30              return d/3;
31          });
32      </script>
```

代码清单 III

```
33     </body>
34 </html>
```

10.3 数据传输的挑战者—JSON

- JSON
  - JavaScript Object Notation
  - 一种轻量级、基于文本、语言无关的数据交换格式
  - JavaScript（Standard ECMA-262）的一个子集
  - 重要贡献者：Douglas Crockford
  - 2006 年 7 月——RFC 4627

10.3.1 JSON 的数据结构

- 无序的键值对集合

```
{ "a":1,"b":2,"c":3 }
```

- 值的有序列表

```
[ 1, 2, 3, "hello world" ]
```

10.3.2 JSON 的值类型

- 符串类型
  - JSON 字符串采用双引号包括所表示的字符内容，字符内容本身包含的双引号使用反斜线进行转义；
- 数值类型
- 对象类型
  - 键值对的无序集合；
- 数组类型
  - 值的有序列表；
- 布尔类型
  - true 表示真、false 表示假；
- 空类型 null

JSON 示例 I

Code 2: contact.json

```
{
  "name": "王某某",
  "age": 30,
  "spouse": {
    "name": "张某某",
    "age": 28
  },
  "addresses": [
    {
      "description": "工作住址",
      "street": "胜利大街9号",
      "city": "潍坊",
      "province": "山东"
    }
```

JSON 示例 II

```
15    ],
16    "phoneNumbers": [
17       {
18          "description": "办公电话",
19          "number": "6666-7777"
20       },
21       {
22          "description": "手机",
23          "number": "1XX-1234-5678"
24       }
25    ]
26 }
```

10.3.3 JSON 与 XML 的对比

- XML 比 JSON 更易于人工阅读
- XML 是典型的标记语言，而 JSON 不是
- JSON 格式简洁，文件的体积较小，有利于数据传输。
- JSON 与 JavaScript 结合紧密，构造和解析都比较容易，而通过 JavaScript 解析 XML，则需要借助于额外的库

contact.json 对应的 XML I

Code 3: contact.xml

```xml
<contact>
  <name>王某某</name>
  <age>30</age>
  <spouse>
    <name>张某某</name>
    <age>28</age>
  </spouse>
  <addresses>
    <address>
      <description>工作住址</description>
      <street>胜利大街9号</street>
      <city>潍坊</city>
      <province>山东</province>
    </address>
```

contact.json 对应的 XML II

```
15    </addresses>
16    <phoneNumbers>
17      <phoneNumber>
18        <description>办公电话</description>
19        <number>6666-7777</number>
20      </phoneNumber>
21      <phoneNumber>
22        <description>手机</description>
23        <number>1XX-1234-5678</number>
24      </phoneNumber>
25    </phoneNumbers>
26  </contact>
```

10.3.4 利用 JavaScript 解析 JSON

- 浏览器内置的 JSON.parse()

```
JSON.parse(text [, reviver])
```

示例 I

- 在浏览器控制台输入以下代码

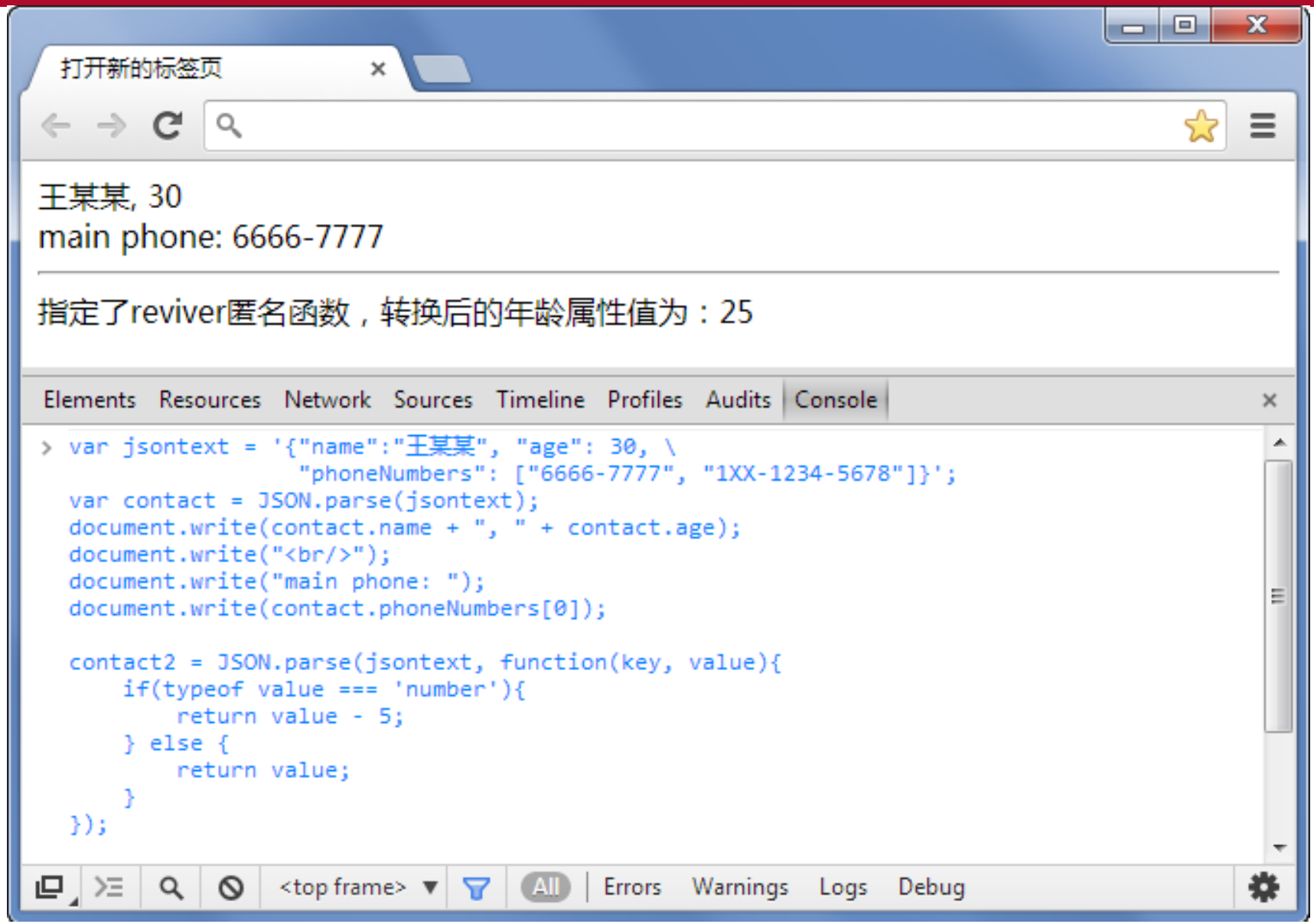
```
var jsontext = '{"name":"王某某", "age": 30, \
            "phoneNumbers": ["6666-7777", "1XX-1234-5678"]}';
var contact = JSON.parse(jsontext);
document.write(contact.name + ", " + contact.age);
document.write("<br/>");
document.write("main phone: ");
document.write(contact.phoneNumbers[0]);

contact2 = JSON.parse(jsontext, function(key, value){
   if(typeof value === 'number'){
      return value - 5;
   } else {
      return value;
   }
```

示例 II

```
});
document.write("<br/><hr/>
        指定了reviver匿名函数，转换后的年龄属性值为：");
document.write(contact2.age);
```

- 浏览器解析结果

示例 III


打开新的标签页
王某某, 30
main phone: 6666-7777
指定了reviver匿名函数，转换后的年龄属性值为：25
Elements Resources Network Sources Timeline Profiles Audits Console
> var jsontext = '{"name":"王某某", "age": 30, \
"phoneNumbers": ["6666-7777", "1XX-1234-5678"]}';
var contact = JSON.parse(jsontext);
document.write(contact.name + ", " + contact.age);
document.write("
");
document.write("main phone: ");
document.write(contact.phoneNumbers[0]);
contact2 = JSON.parse(jsontext, function(key, value){
if(typeof value === 'number'){
return value - 5;
} else {
return value;
}
});
<top frame>
All
Errors
Warnings
Logs
Debug

END