[TOC]

#综述

       graphStream可以可视化顶点和边的结构，支持边、顶点、图的自定义样式，以及样式可以对点击和选择事件（仅仅这两个事件）作出响应式的改变。

        graphStream可以和graphx很好的结合使用，这个在《Apache Spark Graph Process》Chapter3 的例子中有很好的体现。

        graphStream可以通过Scala和Java来编程。

#What is GraphStream？

GraphStream is a dynamic graph library written in Java that provides Java developers a way to easily represent dynamic graphs in memory, on screen or in files.

GraphStream provides a framework to handle the evolution of graphs, that is the changes on values stored on edges and nodes of a graph during time, but also the “topology” changes of the graph, that is the addition, removal and modification of nodes and edges during time.

#Vedio of Demos

Version: GraphStream 1.0

上传日期：2011年5月24日

地址： <https://www.youtube.com/watch?v=XX5rRF6uxow&noredirect=1>

#网站重要位置导航

##Getting Started

<http://graphstream-project.org/doc/Tutorials/Getting-Started/>

#CSS reference[1]

##render

    The default viewer of GraphStream is only able to handle colors and sizes of elements, however the gs-ui module provides another viewer that fully support de CSS described here.

##操作列表

graph.addAttribute("key", “value");

graph.removeAttribute(“key”);

node.addAttribute("key", “value");

##style selectors

##响应clicked and selected事件

To mark an element as clicked, it must have the attribute ui.clicked. This is done automatically by the viewer (for nodes and sprites).

to define the style of elements when a specific event occurs, you can write selectors like node:clicked or sprite::selected.

##Properties common to nodes, edges and sprites

内容很多，涉及具体的使用示例，详见官网。

##z-index:

An integer number indicating the “layer” inside which elements are rendered.

##怎样展现无向边

edge { arrow-shape: none; }

##疑问

1、能否给点击行为绑定事件？

已知的资料没有体现这一点的可行性。

2、能否在后台执行一个命令，如改变最短路径相应节点的样式，界面进行相应动态的改变？

编程验证一下。

#FAQ

1、Why CSS property X does not work ? Why multiple edges between two nodes show as a single edge ?

Make sure to use the advanced viewer:

System.setProperty(“org.graphstream.ui.renderer","org.graphstream.ui.j2dviewer.J2DGraphRenderer");

#计算功能和内置算法

<http://graphstream-project.org/doc/Tutorials/Working-with-algorithms-and-generators/>

graphStream有执行算法和计算的功能。

##Spanning-tree

* [Base for spanning-tree algorithms](http://graphstream-project.org/doc/Algorithms/Spanning-tree/Base-for-spanning-tree-algorithm/)
* [Kruskal](http://graphstream-project.org/doc/Algorithms/Spanning-tree/Kruskal/)
* [Prim](http://graphstream-project.org/doc/Algorithms/Spanning-tree/Prim/)

**##**[**Spanning-tree**](http://graphstream-project.org/doc/Algorithms/Spanning-tree/)

* [A\*](http://graphstream-project.org/doc/Algorithms/Shortest-path/A-star/)
* [All Pair Shortest Path](http://graphstream-project.org/doc/Algorithms/Shortest-path/All-Pair-Shortest-Path/)
* [Bellman-Ford](http://graphstream-project.org/doc/Algorithms/Shortest-path/BellmanFord/)
* [Dijkstra](http://graphstream-project.org/doc/Algorithms/Shortest-path/Dijkstra/)

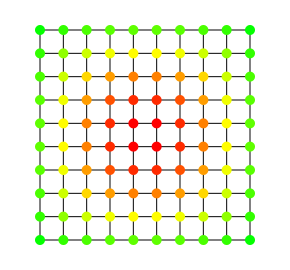
##[Connected Components](http://graphstream-project.org/doc/Algorithms/Connected-Components/) 无向图连通分量

* 通过阅读示例知道可以简单的计算出计算连通分量的数量。

[##Centroid](http://graphstream-project.org/doc/Algorithms/Centroid/) 计算图心

* This algorithm needs that APSP algorithm has been computed before its own computation.

##[Eccentricity](http://graphstream-project.org/doc/Algorithms/Eccentricity/)

* Compute the eccentricity of a connected graph.
* In a graph G, if d(u,v) is the shortest length between two nodes u and v (ie the number of edges of the shortest path) let e(u) be the d(u,v) such that v is the farthest of u. Eccentricity of a graph G is a subgraph induced by vertices u with minimum e(u).
* This algorithm needs that APSP algorithm has been computed before its own computation.
* ##[Betweenness Centrality](http://graphstream-project.org/doc/Algorithms/Betweenness-Centrality/) 中介中心性
* Compute the *betweenness* centrality of each vertex of a given graph.
* The betweenness centrality counts how many shortest paths between each pair of nodes of the graph pass by a node. It does it for all nodes of the graph.
* 
* The above graph shows the betweenness centrality applied to a grid graph, where color indicates centrality, green is lower centrality and red is maximal centrality.
* ##[Several often used algorithms on graphs](http://graphstream-project.org/doc/Algorithms/Several-often-used-algorithms-on-graphs/)
* 度计算
* 密度计算
* **Clustering coefficient**
* **Random nodes and edges**
* **Nodes position**
* **Diameter**
* ##[Random walks on graphs](http://graphstream-project.org/doc/Algorithms/Random-walks-on-graphs/)  随机游走
* ##[Welsh-Powell](http://graphstream-project.org/doc/Algorithms/Welsh-Powell/)
* ##[Tarjan Strongly Connected Components](http://graphstream-project.org/doc/Algorithms/Tarjan-Strongly-Connected-Components/)
* ##[PageRank](http://graphstream-project.org/doc/Algorithms/PageRank/)

2、预定义的属性名有哪些？

* x for the abscissa of nodes (a float value).
* y for the ordinate of nodes (a float value).
* z for the depth of nodes (a float value).
* xy for the 2D position of nodes (pass two float values).
* xyz for the 3D position of nodes (pass three float values). It is a good idea to always use xyz.
* ui.label for node, sprite and edge labels. Setting this attribute will display the text on the node or edge according to the style sheet and text visibility modes.
* ui.color for node, sprite and edge color interpolation when the style defines at least two colors. The value must be a real number between 0 and 1. The style must define at least two fill colors and the fill mode must be dyn-plain, for example : node { fill-mode: dyn-plain; fill-color: red, blue; }. In this example, if the ui.colorvalue is 0 the node will be red, if the value is 1 the node will be blue and if the value is 0.5 the node will be a mix of the two, some kind of purple.
* ui.style for node, edge and sprite styling. This merges the style to the actual style sheet.
* ui.stylesheet for graph styling. Each new addition or change of this attribute will merge the style definitions to the actual style sheet. To completely clear the style, remove this attribute from the graph.
* ui.hide to hide a node, edge or sprite (only its presence is tested, not its value).
* ui.quality to enable slower but better rendering.
* ui.antialias to enable anti-aliasing of shapes drawn by the viewer.
* ui.screenshot for graphs, to quickly take a screen shot of the viewer. The value must be the name of a the image file you want to save. Once the screen shot has been saved, the attribute is automatically removed. For examplegraph.addAttribute("ui.screenshot", "/some/place/image.png");.

#其他

##API

<http://graphstream-project.org/doc/API/>

##最新版源码

<http://graphstream-project.org/doc/FAQ/General-Questions/GraphStream-versions-GIT/>

##提问邮箱

<http://graphstream-project.org/doc/FAQ/General-Questions/Where-do-I-start/>

#一个接口多个实现

<http://graphstream-project.org/doc/FAQ/The-Graph-Class/What-are-the-Graph-implementations/>

DefaultGraph 不允许多重边

MultiGraph  允许多重边

AdjacencyListGraph  使用内存少， but may be slower for some uses.

ConcurrentGraph. An implementation of a Graph with multi-thread capabilities.

#**layout algorithms**

[**http://graphstream-project.org/doc/FAQ/Attributes/Is-there-a-list-of-attributes-with-a-predefined-meaning-for-the-layout-algorithms/**](http://graphstream-project.org/doc/FAQ/Attributes/Is-there-a-list-of-attributes-with-a-predefined-meaning-for-the-layout-algorithms/)

#把图的动态变化过程做成小视屏

<http://graphstream-project.org/doc/Tutorials/Creating-a-movie-with-FileSinkImages/>

#参考资源链接

[1]<http://graphstream-project.org/doc/Advanced-Concepts/GraphStream-CSS-Reference/>

**Betweenness Centrality 可以尝试做一做**

边以无箭头形式展现

顶点随鼠标拖动定位到新的位置

<http://graphstream-project.org/doc/FAQ/The-graph-viewer/Why-node-positions-do-not-update/>

动态改变顶点和边的颜色（如简单路径中）

<http://graphstream-project.org/doc/FAQ/The-graph-viewer/How-do-I-dynamically-change-color-and-size-in-the-viewer/>