gexfBuilder V0.2.2 User manual

This manual covers the functions used for formatting data to the gexf file format use by gephi. This module uses the elementTree module to build the file. All input data is typecasted to strings to allow writing out to a gexf file. The module uses a gexftree object whose attributes build the gexf tree.

### gexftree(mode, edgetype, timeF=None, timeR=None)

Constructs skeletal frame for a gexftree and determines properties graph will have.

##### **Caveats**: when making a dynamic graph, a time format and time representation must be chosen

**parameters:**

* **mode:** *(keyword arguments)* “dynamic” | “static”
* **edgetype:**  *(keyword arguments)* “directed”| ”undirected”
* **timeF:** *(keyword arguments)* “double”|”date”
* **timeR:** *(keyword arguments)* “timestamp” | “timeformat”

### gexftree.addAttribute(lss, id, title, type, mode=None, default=None)

Add an attribute for a graph to the gexfTree

**caveats:** the default value may be a deprecated feature as of gexf v1.3

**parameters:**

* **clss:** (*keyword arguments)*“node” | “edge”
* **id:** *(string)* id for attribute to be referenced by
* **title:** *(string)* id attribute to be called
* **type:** (*keyword arguments)* “integer” | “double” | “boolean” | “string” | “list-string”
* **mode:** (*keyword arguments)(defaults to ‘static’)* “static” | “dynamic” (defaults to static)
* **default:***(string)* value for attribute( may be deprecated)

### gexftree.addNodes(id, label=None, timeStamps=None, intervals=None)

Adds a list of nodes to the gexfTree. A list of spells or time intervals can be added to each node. The node will appear along with attributes on the graph when the timeline reaches its timestamps and intervals. Raises an error if the node already exists.

**parameters:**

* **id:** (*string)* id for node to be referenced by, must be unique
* **label:** *(string)* for node to called, needn’t be unique
* **timeStamps**: *(list)*  timestamps to add to each node in the node list
* **intervals:** *(list)* of intervals to add to each node in the node list

### gexftree.addEdge(id, source, target, label=None, kind=None, timeStamps=None, intervals=None)

Adds a list of edges to the gexfTree. A list of spells or time intervals can be added to each node. The edge will appear along with attributes on the graph when the timeline reaches its timestampss and intervals. Raises an error if the edge already exists

Parameters:

* **id:** *(string)*id edge to be referenced by
* **source:** node id that edge comes from if graph is directed
* **target:** node id that edge goes to if graph is directed
* **label:** *(string)* edge to be called, needn’t be unique
* **kind:** *(string)* for edge kind, this allows for multigraph support and stops gephi from combining edges that have same source and target nodes
* **timeStamps:** (*list)* timestamps to add to each node in the node list
* **intervals:** *(list)* intervals to add to each node in the node list

### gexftree.**addAttvalues(**clss, clssIds, attId, value, timeStamps**)**

Adds attvalue to list of nodes or edges. These attvalues reference a specific attribute defined by addAttribute. A list of timestamps or intervals can be specified for the attribute’s value to range over.

**Parameters:**

* **clss:** *(keyword arguments)* “node” | “edge”
* **clssIds:** *(list)* ids for nodes or edges to add the list of attvalues to
* **attId:** id of attribute the attvalue refers to.
* **value:** value for attribute
* **timeStamps:** *(list)* of stamps to add to each node in the node list

### gexftree.**addSpells(**clss, clssIds, timeStamps**)**

Adds a list spells for a list of nodes or edges. These spells are just more timestamps or intervals for the nodes/edges. Spells are feature of gexf and do not work for attvalues.

**parameters:**

* **clss:** *(keyword arguments)*“node” | “edge”
* **clssIds:** *(list)* of string ids for nodes or edges to add the list of spells to
* **timeStamps:** *(list)* of stamps to add to each node in the node list

### gexftree.**write(**outfile**)**

Writes the current gexftree to a gexf file.

**caveats:** output is not printed prettily and is difficult to read. Use the function pretty() to see in terminal what the gexftree looks like  
  
**parameters:**

* **outfile:** *(string)* filename to write gexftree to, do not include extension

### **pretty(**targetfile**)**

Reads a gexf file and prints the contents in a formatted manner

**caveats:** the function reads from already existing xml file, not from gexftree object

**parameters:**

* **targetfile:** *(string)* name of gexf file