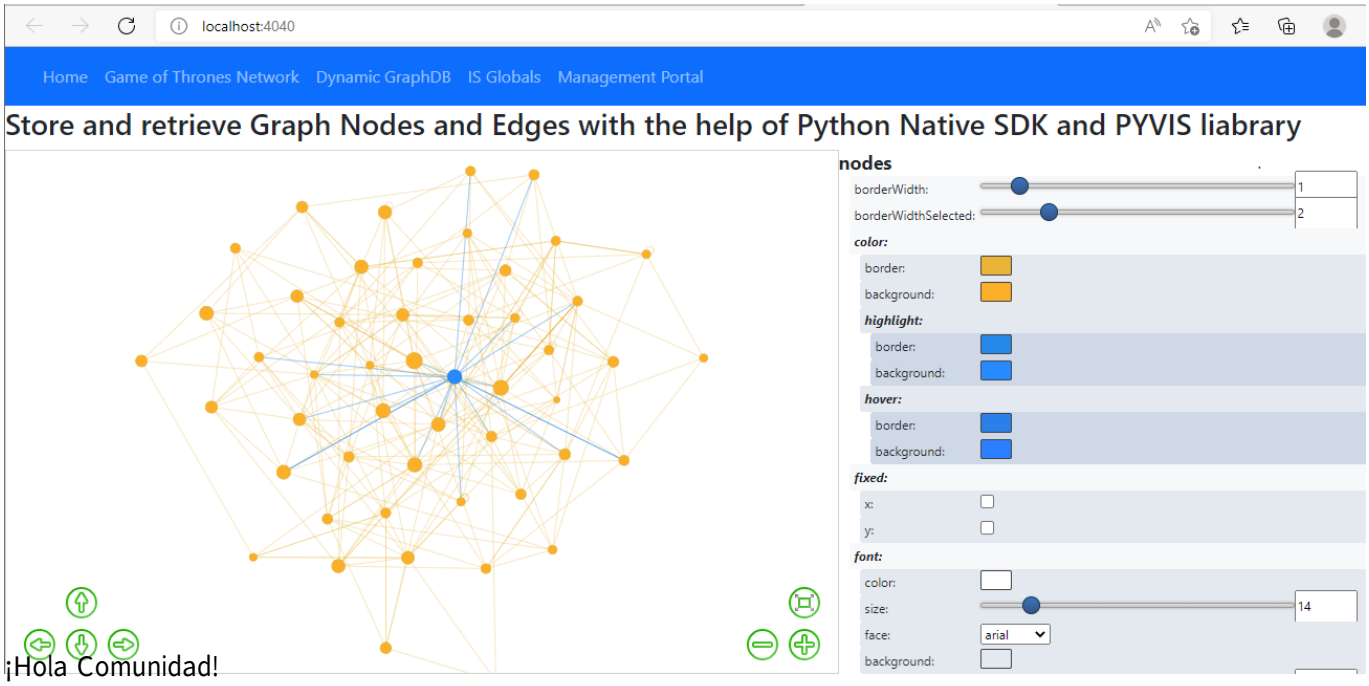
Artículo
Muhammad Waseem · 6 abr, 2022 Lectura de 7 min

Open Exchange

Cómo usar Globals como base de datos gráfica para almacenar y recuperar datos de estructura de gráficos



¡Hola Comunidad!

Esta publicación es una introducción a mi aplicación **iris-globals-graphDB** en Open Exchange.

En este artículo, mostraré cómo guardar y recuperar **Graph Data** en **InterSystems Globals** con la ayuda del *framework* **Python Flask Web** y la librería **PYVIS Interactive network visualizations**.

Recomendación

- Leer la documentación relacionada: Using Globals
- Introducción al SDK nativo
- PYVIS Librería de visualización interactiva de redes

Paso 1: Establecer conexión con IRIS Globals mediante el SDK nativo de Python

```
 #create and establish connection
  if not self.iris_connection:
         self.iris_connection = irisnative.createConnection("localhost", 1972, "USER"
, "superuser", "SYS")

  # Create an iris object
  self.iris_native = irisnative.createIris(self.iris_connection)
  return self.iris_native
```

Paso 2: Guardar datos en *globals* mediante la función irisnative.set( )

```
#import nodes data from csv file
isdefined = self.iris_native.isDefined("^g1nodes")
if isdefined == 0:
    with open("/opt/irisapp/misc/g1nodes.csv", newline='') as csvfile:

 reader = csv.DictReader(csvfile)
 for row in reader:
    self.iris_native.set(row["name"], "^g1nodes", row["id"])

 #import edges data from csv file
 isdefined = self.iris_native.isDefined("^g1edges")
 if isdefined == 0:
    with open("/opt/irisapp/misc/g1edges.csv", newline='') as csvfile:
 reader = csv.DictReader(csvfile)
 counter = 0
 for row in reader:
    counter = counter + 1
    #Save data to globals
    self.iris_native.set(row["source"]+'-'+row["target"], "^g1edges", counter)
```

Paso 3: Transferir datos de nodos y bordes a PYVIS desde *globals* mediante la función irisnative.get()

```
 #Get nodes data for basic graph
  def get_g1nodes(self):
        iris = self.get_iris_native()
        leverl1_subscript_iter = iris.iterator("^g1nodes")
        result = []
        # Iterate over all nodes forwards
        for level1_subscript, level1_value in leverl1_subscript_iter:
            #Get data from globals
            val = iris.get("^g1nodes",level1_subscript)
            element = {"id": level1_subscript, "label": val, "shape":"circle"}
            result.append(element)
        return result

    #Get edges data for basic graph
    def get_g1edges(self):
        iris = self.get_iris_native()
        leverl1_subscript_iter = iris.iterator("^g1edges")
        result = []
        # Iterate over all nodes forwards
        for level1_subscript, level1_value in leverl1_subscript_iter:
            #Get data from globals
            val = iris.get("^g1edges",level1_subscript)
            element = {"from": int(val.rpartition('-')[0]), "to": int(val.rpartition(
'-')[2])}
            result.append(element)
        return result
```

Paso 4: Usar PYVIS Javascript para generar datos gráficos

```
<script type="text/javascript">
    // initialize global variables.
    var edges;
    var nodes;
    var network;
    var container;
    var options, data;

    // This method is responsible for drawing the graph, returns the drawn network
    function drawGraph() {
        var container = document.getElementById('mynetwork');
        let node = JSON.parse('{{ nodes | tojson }}');
        let edge = JSON.parse('{{ edges | tojson }}');

        // parsing and collecting nodes and edges from the python
        nodes = new vis.DataSet(node);
        edges = new vis.DataSet(edge);

        // adding nodes and edges to the graph
        data = {nodes: nodes, edges: edges};

        var options = {
            "configure": {
                "enabled": true,
                "filter": [
                "physics","nodes"
            ]
            },
            "nodes": {
                "color": {
                  "border": "rgba(233,180,56,1)",
                  "background": "rgba(252,175,41,1)",
                  "highlight": {
                    "border": "rgba(38,137,233,1)",
                    "background": "rgba(40,138,255,1)"
                  },
                  "hover": {
                    "border": "rgba(42,127,233,1)",
                    "background": "rgba(42,126,255,1)"
                  }
                },

                "font": {
                  "color": "rgba(255,255,255,1)"
                }
              },
            "edges": {
                "color": {
                    "inherit": true
                },
                "smooth": {
                    "enabled": false,
                    "type": "continuous"
                }
            },
            "interaction": {
                "dragNodes": true,
```

```
                "hideEdgesOnDrag": false,
                "hideNodesOnDrag": false,
                "navigationButtons": true,
                "hover": true
            },

            "physics": {
                "barnesHut": {
                    "avoidOverlap": 0,
                    "centralGravity": 0.3,
                    "damping": 0.09,
                    "gravitationalConstant": -80000,
                    "springConstant": 0.001,
                    "springLength": 250
                },

                "enabled": true,
                "stabilization": {
                    "enabled": true,
                    "fit": true,
                    "iterations": 1000,
                    "onlyDynamicEdges": false,
                    "updateInterval": 50
                }
            }
        }
        // if this network requires displaying the configure window,
        // put it in its div
        options.configure["container"] = document.getElementById("config");
        network = new vis.Network(container, data, options);
        return network;
    }
    drawGraph();
</script>
```

Paso 5: Llamar a los códigos anteriores desde el archivo principal app.py

```
#Mian route. (index)
@app.route("/")
def index():
    #Establish connection and import data to globals
    irisglobal = IRISGLOBAL()
    irisglobal.import_g1_nodes_edges()
    irisglobal.import_g2_nodes_edges()

    #getting nodes data from globals
    nodes = irisglobal.get_g1nodes()
    #getting edges data from globals
    edges = irisglobal.get_g1edges()

    #To display graph with configuration
    pyvis = True
    return render_template('index.html', nodes = nodes,edges=edges,pyvis=pyvis)
```

Espero que os resulte útil.

#Bases de datos #Concurso #CSS #Modelo de datos #Python #Caché #InterSystems IRIS for Health #Open Exchange #VSCode
Ir a la aplicación en InterSystems Open Exchange