

Graph Theory

Using Neo 4J

Damien Joyce| Graph Theory | 20 April 2017

# Introduction:

My name is Damien Joyce and I am a third year Computing and Software Development student. This documentation details of how I created a project for one of my current modules as part of my studies while attending GMIT college Galway. The project built on a Neo4J database. This project will show the relationships of the timetabling system in GMIT. Each student and lecturer is allocated to a certain room and certain classes throughout the day. Each subject is related to the lecturer and also the students, they are then allocated to a certain room for the duration of that class. This information will be stored onto a database using Neo4J. I have decided to create a project detailing my current timetable of GMIT. This project will show the relationships and nodes of my Second Semester in Third year while attending GMIT Galway. The topics this document I will cover are shown below:

* Requirements
* What is Node.js
* Neo4j Commands
* How I wish to complete this project
* Conclusion
* Designs
* Data collected

## Requirements:

In order to complete this project I must first determine what is exactly needed in order to complete this project. I must determine the nodes, relationships, labels, relationship types and the properties of all the information I wish to add to the database. I have come to the conclusion that in order to complete this task effectively I must organise the details of what is needed in timetable in order for it to work effectively and efficiently. What I have discovered are described below:

1. Rooms
2. Student groups e.g. Third Year Software Development group A
3. Times e.g. Monday at 09:00 – 10:00 am
4. Lecturer e.g. Ian McLoughlin
5. Subject e.g. Graph Theory

## What is node.js:

Node.js is a platform built on Google Chrome’s JavaScript runtime for creating fast scalable network applications. Node.js uses a non-blocking event driven model that makes it a lightweight efficient project. It is ideal for intensive real time applications that run on many devices. It was developed by Ryan Dahl in 2009. Node.js is an open source, cross-platform runtime environment for developing server side and networking applications. Node.js also provides a large library of various JavaScript modules which simplify the development of web applications.

## Features of node.js:

* **Asynchronous and Event Driven** − All APIs of Node.js library is asynchronous this means it is non-blocking. It essentially means a Node.js server never waits for an API to return data. The server moves to the next API after calling it. The Events of Node.js helps the server to get a response from the previous API call.
* **Very Fast** – Node.js is built on Google Chrome's V8 JavaScript Engine. This allows Node.js to be very fast with its code execution.
* **Single Threaded but Highly Scalable** − Node.js uses a single threaded model with event looping. This event mechanism helps the server to respond with a non-blocking mechanism which makes the server highly scalable as in comparison to older servers which create limited threads to handle its requests. Node.js uses a single threaded program and this program can provide service to a much larger group of requests rather than traditional servers like Apache HTTP Server.
* **No Buffering** − Node.js applications never buffer any data. This sort of applications output the data in large “chunks” of data instead.

## Neo4j Commands:

I used many different commands to complete a graph of my current timetable of GMIT. I had to change the names of the relationships and the nodes but here are some of examples of the commands I used to complete this project:

This code was used to create the node Student Program and give it some values:

* CREATE (n:StudProgram {spNum: "G-KBDIG72", spName: "BSc in Business Computing &amp; Digital Media", spLevel: 7, spYear: 2, spSemester: 4, spGr: "A"}

This code was used to create the Subject node with the value of Server Side RAD:

* CREATE (s:Subject { name: "Server Side RAD" }) RETURN s

This code was used to create a relationship with the lecturs node and the Subject node and match the values Gerard Harrison & Server Side RAD:

* MATCH (l:Lecturs { name: 'Gerard Harrison' }), (s:Subject { name: 'Server Side RAD' }) CREATE (l)-[:TEACHES]->(s)

This code was used to create a relationship with the Groups node and the Day node and match the values All & Friday

* Match(g:Groups { group: 'All' }), (d:Day { day: 'Friday' }) CREATE (d)-[:Days]->(g)

This code was used to create a relationship with the Day node and the Subject node and match the values Monday & Server Side RAD:

* MATCH (d:Day { day: 'Monday' }), (s:Subject { name: 'Server Side RAD' }) CREATE (d)-[:On]->(s)

This code was used to create a relationship with the Groups node and the Subject node and match the values C & Server Side RAD:

* MATCH  (g:Groups { group: 'C' }), (s:Subject { name: 'Server Side RAD' })  CREATE (s)-[:Monday]->(g)

This code was used to create a relationship with the Times node and the Subject node and match the values Database Management & 14:

* MATCH (s:Subject { name: 'Database Management'}), (t:Times { times:14})CREATE (s)-[:Thursday]->(t)

## How I wish to complete this project:

I have many designs of how I could use to complete this project the designs are shown below in the design section. I will try to complete this project to the best of my ability and so that it is efficient but also to be appealing to the eye of the viewer. I wish to create a centre node where all relationships and other nodes will be connected. I decided on design number 2. This was designed to create a node of my current college course in GMIT Galway (Computing & Software Development) and make that the centre node where all other nodes will be connected. I will then create the nodes for each day of the week where classes will be held which happens to be from Monday to Friday. The next step was to figure out which node would be best suited to create a relationship between the day nodes and all the other nodes that I will have to create. I decided that the most efficient node to create next would be the groups node. There is currently four groups in my course which are All(everyone), A, B and C. I will then link the day nodes to the group nodes with relationships labelled the days of the week to stop confusion between subjects and times. I must then decide how to make each relationship different from the other in appearance. Each relationship contains a different colour so that it is more attractive to the person viewing the graph. The next step would be deciding on which node would come next after the groups. I have decided that the subject node would be the most effective for my desired outcome. I must then create a relationship between the groups and the subjects. The next step I think which will be the most difficult will be creating the times node. I believe this will be the most difficult as each time node I create will have multiple relationships with other subjects as some subjects are run parallel to others. I will use a 24 hour clock to show the times of each class as I think it is the most effective. The final node which I will create is the room node. This node will just contain the number of each room and will have a relationship connected to the time node. I am willing to change some of the order of these nodes to best suit the requirements of this project.

## Conclusion:

After completing this project I have realised that it is much more difficult than first expected. I made many different attempts in the order of the nodes and how to create the relationships that would be best suited to each node. I found that there was a lot of information that I had to put into the project and I had to be careful with the labelling of the nodes as there was many nodes that were needed to create this project. When I first began this project I didn’t realise that after creating a few nodes that the relationships would make it so hard to understand and the direction of each node was complicating at that time so I decided to create a small template to start with and then create my official project so there would be no mistakes but I still have some mistakes in my official project and I can’t seem to delete them without deleting all the relationships or node I have already created. If I were to do this project again I would take more time to distinguish the different type of nodes I would use but also I would try find a more direct way creating relationships between the nodes. I spent much of my time trying to fix the mistakes I created and this caused me to delete much of my project so in future attempts I would be more cautious on attempting a different project.

On another note I found this project to be quite enjoyable and I believe I have learned a lot after completing this project. I learned that in order to complete a successful project one must have a detailed outline of how they wish to complete such a large scale project. Each detail must be mentioned and sketches are a must in order to complete a project similar to this one.

## Design 1:

## Design 1:

## Data collected:

|  |  |
| --- | --- |
| **Staff** | **Subject** |
| Deirdre O’Donovan | Database management |
| Ian McLoughlin | Graph Theory |
| Damien Costello | Mobile Applications 2 |
| Gerard Harrison | Server Side RAD |

|  |  |  |  |
| --- | --- | --- | --- |
| **Times** | **Monday** | **Rooms** | **Group** |
| 10:00-11:00 | Database Management | 994 | All |
| 12:00-14:00 | Mobile Applications 2 | 223 | All |
| 14:00-15:00 | Graph Theory | PF05 | A |
| 16:00-18:00 | Software Testing | 481 | A |
| 16:00-18:00 | Server Side RAD | 436 | C |
| **Times** | **Tuesday** | **Rooms** | **Group** |
| 09:00-11:00 | Software Testing | 436 | C |
| 09:00-11:00 | Database Management | CR4 | A |
| 11:00-13:00 | Database Management | 482 | C |
| 11:00-13:00 | Mobile App | 470 | A |
| 12:00-13:00 | Graph Theory | 379 | B |
| 14:00-16:00 | Software Testing | 436 | B |
| 16:00-18:00 | Server Side RAD | 436 | B |
| **Times** | **Wednesday** | **Rooms** | **Group** |
| 09:00-10:00 | Graph Theory | 162 | C |
| 10:00-11:00 | Graph Theory | 938 | All |
| 13:00-14:00 | Server Side RAD | 997 | All |
| 15:00-16:00 | Software Testing | 939 | All |
| 16:00-17:00 | Database Management | 995 | All |
| **Times** | **Thursday** | **Rooms** | **Group** |
| 9:00-11:00 | Mobile Applications 2 | PF18 | B |
| 11:00-13:00 | Mobile Applications 2 | PF18 | C |
| 14:00-16:00 | Server Side RAD | 436 | A |
| 14:00-16:00 | Database Management | 483 | C |
| **Times** | **Friday** | **Rooms** | **Group** |
| 09:00-11:00 | Database Management | 482 | B |
| 11:00-12:00 | Graph Theory | 208 | All |