IS3 Assessed Exercise 1
Group Project

Olympic Games Data Visualisation

<u>Team Members</u>

Alastair Weir (1101682w)
Matthew Paterson (1102374p)
Lewis Mcgeechan (1101982m)
Fraser Leishman (1102103l)
Josh Mcghee (1101718m)
Stefan Balling (1101356b)
Eddie Graham (1101301g)

Introduction

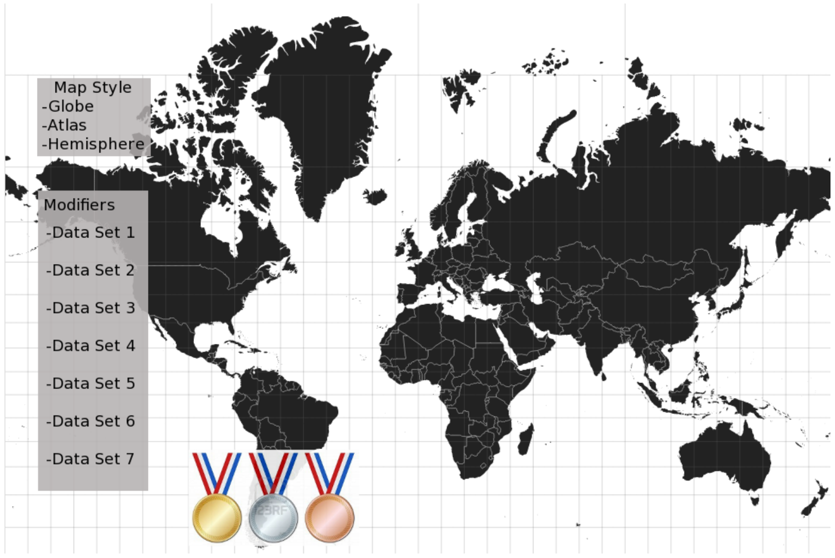
Undertaking the project using multiple technologies, without any experience in any of them, presented a challenge from the outset. Starting with our paper prototypes and the data we had extracted from the given files we aimed to produce a working system which would allow us to view any trends between medal wins at the Olympic games and quality of life quantifiers collected by the World Health Organisation.

Design Requirements

Having successfully prototyped multiple versions of a data visualisation of Olympic Games and World Health Organisation data sets, we advanced the design requirements, using the MoSCoW notation, and laid out our aims for development.

Must Have	**Should Have**	**Could Have**	**Would Have**
-d3 map visualisation from TopoJSON file using SVG graphics -Floating menus to contain controls to switch between data views -Successfully import country data from JSON/CSV file -Project data onto map visualisation -Display the specified statistic to the viewer	-Rotating Globe views -Wrapping website (Created with framework) -Basic click to zoom functionality -Make floating menus hide-able -Portray different values as a choropleth map	-Click to zoom with statistics card -Extra map views	-Responsive design -Database Storage

Reviewing the must have requirements that can all be directly extrapolated from our original prototypes. We wanted to use d3 to draw the map. This map will take up the entire browser window, being the main interface the user interacts with. we would like the world map to dominate even further than previously anticipated. We thought that d3 would be the best way to communicate our information. Non map resources on screen would be floating divs that could be minimised if the user so wishes. It should look something like our final prototype.



Team Organisation
Due to the numerous other pressures and deadlines from other courses, the team worked on an individual basis early on, working on what they felt was needed. As the deadline approached and other assignments were submitted we worked synchronously in the lab implementing pair programming to advance in the must have requirements. Brief meetings were held each day during the submission week to discuss what had been done. A different member of the team took leadership each day, dividing tasks between the team members and helping to solve any problems they were experiencing.

We relied on a GitHub repository as a central point but this was badly maintained, with branches being incorrectly used and multiple versions of the same file being stored in different branches. This was mainly due to lack of experience using GitHub correctly but the team have since realised the importance of keeping a clean and tidy repo as there was crossover on some of the branches causing duplicated work. The master branch was kept clean and now holds our final project with any other resources such as the original data files stored in another folder on this branch.

We aspired to form an agile team organisation but largely tackled problems on an individual basis. This has caused some problems but we have been able to meet up enough to tackle and solve any problems this development process has created.

Implementation Notes
To begin, team members quickly worked through basic Javascript tutorials leveraging

tools such as Code Academy and information found on Stack Exchange. As described in the first report we looked to Mike Bostock's code snippets to help with our implementation, modifying them to suit our needs. We found the documentation for d3 to be subpar and difficult to find the solution to several issues we ran into.

Our main implementation issue was importing information from data files to then be used in synchronisation with the world data Json file that we were using . Starting with a CSV containing our desired datasets linked to their countries we tried multiple approaches to get these into a state that d3 could easily read them in.

The first approach was to combine the data values as attributes of each country in the .json file which held the data to draw the map. To do this we first took the .json file holding the map data with a key identifier which could be linked to our dataset. We used the 3-letter ISO codes for the countries as a standardised identifier. Using country names could raise variations and synchronization issues. The next stage was to convert our dataset to JSON. From there we tried to link them together using that identifier, leaving the map data and our data in separate files. This was initially unsuccessful due to the map data we originally chose.

The second approach involved finding a shape file from a geographical data website which was accurate. These files have their own database into which we placed our values. This was then converted from a shapefile to a GeoJSON. A special type of JSON. This GeoJSON file could be used but the file size was extremely large. We then converted it to a TopoJSON file which is much more compact. Running this file through Mapshaper.org allowed us to to have granular control and reduce the detail on the map (in turn, decreasing the file size) until it suited. This master file was meant to be used to construct our map and place the data. Unfortunately this caused issues such as the United Kingdom being identified as Switzerland when the corresponding data was pulled from the file.

We also tried to use jQuery to pull the data in from the respective files but this created errors which were hard to troubleshoot and weren't easily fixable within our timescale for completion. Our other challenge was to leverage Javascript successfully to store all the data correctly and then be able to refer to it effectively.

In the end we implemented the two file system, with one of the files being as CSV. By this point we had successfully written custom map views needed for the visualisation which were hacked together with our minimal knowledge of Javascript. These successfully implemented click to zoom, colouring the individual countries and the globe view spun.

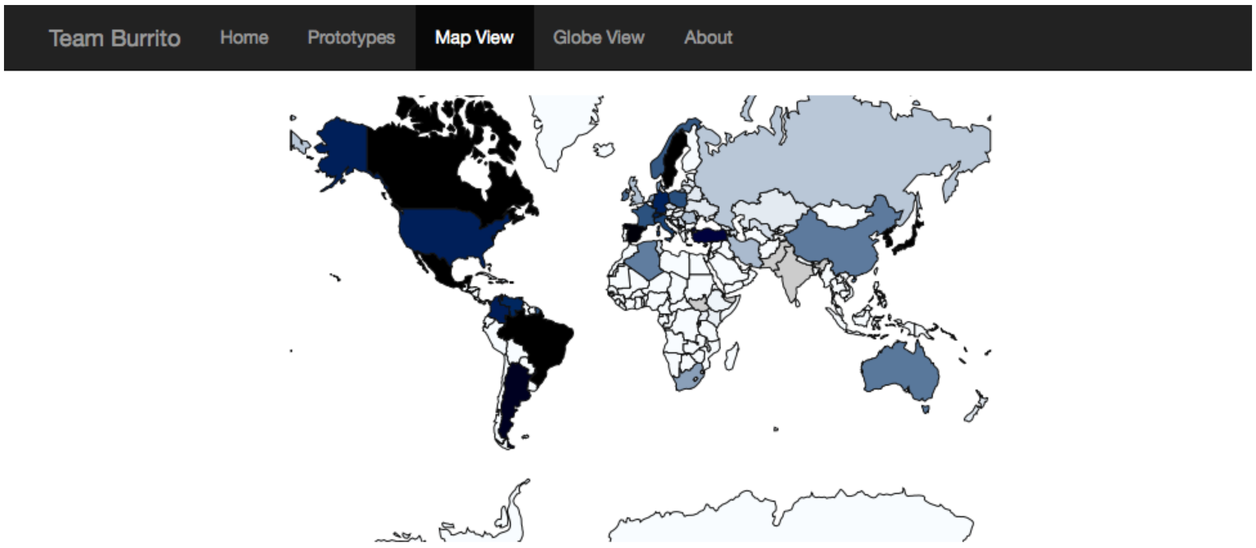
These had to be replaced with maps which we created from information found on TechSlides. This resource was found on the last day of implementation and extensively discussed and explained a lot of the things we had gotten stuck on throughout the project. Changing to this allowed us to get the import working with the data files and be able to successfully create our choropleth maps. As the team were now using a different Json file that would draw the map, new data had to be implemented into the csv file as we needed to have a unique identifier to help map the data to each country correctly.

To bring our disparate map views and various other project data together we initially tried to use Pure CSS, a new CSS library from YUI, to improve the look and design of our project. The lightweight library allows the creation of a richly designed website with a footprint of 5kb. While looking great and allowing us to implement more attractive buttons and menu bars was ultimately abandoned as it was proving difficult to first time web developers to work with. We changed to Twitter's Bootstrap framework as it had better documentation compared to Pure and is heavily used meaning it was a lot easier to troubleshoot online. This gave us an attractive website which allowed us to bring structure to the project.

As can be seen in the source code, we have index.html as the only visible file with styling, scripts and other web pages stored in their respective folders. This structure makes it easier to understand what is happening and make changes to the system as a whole.

Many of the problems we ran into throughout development would have seemed trivial to anyone with experience in web development. It was a risk to choose Javascript and d3 as the environment to develop the data visualisation but the possibilities offered by the language and library cemented the choice amongst the team even when running into these difficult problems.

<u>Description of Submitted Version</u>



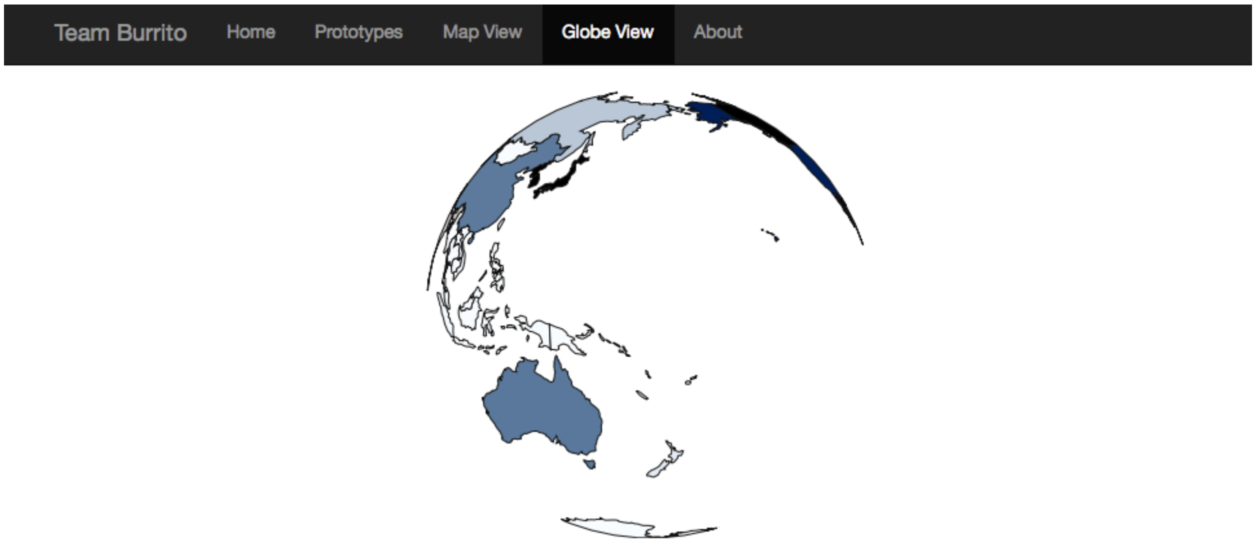
Our final version stays close to the prototype but unfortunately doesn't function correctly.

We were unable to make the datasets switch and the click to zoom functionality we had hoped to implement would not work due to d3 issues. To replace this we have to click to the centre.

Our system only shows one data set, we have included a folder (“ALTERNATIVE”) which has multiple .html files showing different data visualisations which were hard coded but we were unable to implement because we couldn't work out how to switch the data sets dynamically using javascript. The dataset that is shown is gold medals compared to GDP.

The biggest feature we had to remove was the statistic card which would pop up when a user zoomed in on a specific country. This added a lot of work and took some specialist knowledge we did not have the time to gather and learn in the short development cycle. Instead if the user hovers over the country they will be shown the specific value.

By having a minimalist design in the prototype stage it made our implementation a lot easier.



Analytic Findings
With implementation complete we found a number of interesting findings. The most visible would be from the comparisons of the continents and how many medals (as a percentage) they attained. Europe had the highest medal count and percentage by quite a large margin, with them taking 50% of all silver medals. Asia was the next continent with about 22% of the silver medals. Originally, we assumed that the continent with the most countries will have the highest percentage. This is not the case however as Africa, which

has the highest number of countries, has the lowest medal percentage.

When comparing other sets of data such as population per medal, typically, the smaller country, the better the ratio, as even if they won only one medal, that value would still be better than that obtained from countries with large populations and large medal counts(US).

China has lowest debt to medal wins. extends to most developer countries. (ethiopia random exception)
China is shown to have the lowest debt in ratio to medal count. This extends to most of the western world countries (USA, UK, Germany). However, there is an exception that is Ethiopia which has the second lowest ratio in the data set.

Continuing with the western world, the amount spent on tertiary education in relation to gold medals is also lower among these countries. However, this trend doesn't really continue the higher the ratio becomes.

Many countries, notably, African, Asian and some small islands off the coast of countries do not actually achieve any medals at all.

Tools Used
The team used many different tools to vary degrees of success. Listed below are the ones which we used to create our final version.

Bootstrap - Used as website's framework, using its menu and button stylings in the application
d3 - Used to display the map and then project the data onto the map
GDAL - Specifically ogr2ogr command. Used to convert a Shape file with database to a GeoJSON.
js-beautify - Used to format JSON files in a readable way.
TopoJSON - used to convert GeoJSON to smaller TopoJSON file to improve performance. Developed by d3 creator, Mike Bostock.
npm - Part of Node.js framework. Used as a package manager for TopoJSON and js-beautify.
jsFiddle - Used for early rapid prototypes and for learning Javascript.
mapshaper - Used to adjust detail of map file in JSON format.
JSON Validator - Used to test the JSON file and check if valid. Due to the flexibility of JSON this returned false negatives and ultimately caused more issues than it solved.
Pure CSS - Used for buttons in floating menus.
MyGeoData Convertor - Online tool that was ultimately used to successfully convert the shape to JSON.

Distribution of Work

As is with most team projects, the effort put in by some members was far more than others.

Part 1

Initial prototype - All of the team
Final prototype design - All of the team
Data wrangling - Lewis Mcgeechan, Matthew Paterson and Alastair Weir
Video editing - Alastair Weir
Report writing - Alastair Weir and Eddie Graham

Part 2

Requirements - Alastair Weir
System
  Website - Alastair Weir
  Initial d3 implementations - Josh McGhee and Lewis Mcgeechan
  Second d3 implementations - Lewis Mcgeechan
  Final d3 implementation - Josh McGhee, Matthew Paterson and Lewis Mcgeechan
  Data wrangling - Matthew Paterson, Lewis Mcgeechan and Alastair Weir
  Data conversion - Matthew Paterson and Alastair Weir
  Data import into Javascript - Matthew Paterson, Lewis Mcgeechan and Alastair Weir
  Menu design - Fraser Leishman, Eddie Graham and Stefan Balling
Analysis - Matthew Paterson
Report writing - Alastair Weir with small contributions from Eddie Graham and Stefan Balling

Execution Instructions

The entire project has been delivered as a .tar.gz file. Extract the folder from within this and store it in your desired path. Adhering to the submission requirements, no extra software or plug-ins are required. All scripts are included or referenced to.

Open a terminal in this directory and run the command “***python -m SimpleHTTPServer***”. This will start a local server which will allow the project to run correctly in most browsers. In the project’s folder open index.html. This will display the homepage from which you can find the two map views. Our system does not run reliably on Chrome, but should work fine on Firefox or Safari.

One of the menus on screen will allow you to change which data set you are currently viewing and the other will allow you to change between the three grades of medals to see how each influences the data portrayed.

Any issues please email ally.pcgf@gmail.com and we will try and troubleshoot any issues.

Conclusion

In conclusion with a lot of effort from certain team members we managed to create an almost working system with a lot of it coming together in the eleventh hour. Unfortunately we underestimated the difficulty of learning new technologies and have been unable to complete the task. As shown in our findings a number of interesting trends were discovered using the choropleth style map. It successfully displayed the information in an attractive way and instantly allowed us to find information from just looking at the maps. If more time had been available, a more full system could have been implemented - matching the prototyped system more appropriately.