Process Book

Anime Data Visualization

Alexandre CHAU, Pedro TORRES DA CUNHA, Joachim DUNANT
May 29, 2020

1 Introduction

Animes have become a major source of digital entertainment, often narrating stories in beautiful or dreadful fantastical universes. Like with many works of fiction, one can escape to a different world, empathize with charismatic characters, discover a foreign culture or simply appreciate the art itself. They have grown exponentially in popularity over the years, in particular with Western audiences. The result is a rich byproduct of human creativity, globalization and commercialism.

In this context, we attempt to bring insightful visualizations about the raw numbers obtained from these works of art that we love, but also the people behind their creation. We want to showcase what makes them so unique, and bring forward the artists and companies that produce them. We showcase all animes registered on MyAnimeList from 1917 to 2018, their diversity through their genre classification, and detailed information about the studios and "seiyuus", the voice actors of the industry. Furthermore, we focus on an interactive, elegant and playful user experience so that our visualizations are not only informative, but also aesthetically pleasing.

2 Problem statement

With our data visualization, we aim at answering the following questions:

- How has anime evolved from its earliest years in 1917 until now? How did the number of produced animes increase through time? Is the average number of episodes per anime title now greater or smaller?
- How do we categorize anime? How are anime titles distributed across genres? What are the most popular and most represented genres?
- What are the most popular studios? Which genres are the most prevalent for each of them and how are the works rated by their audience?
- What are the most popular voice actors, for each language? How does the "seiyuu" social connections network look like?

3 Dataset

The dataset used to answer those questions is the MyAnimeList Dataset. It contains data about all animes up to 2018 from MyAnimeList, one of the biggest social platforms to discover, rate and review animes and mangas. Precise and detailed information about each anime can be extracted such as genres, studio, popularity, view count and more.

The voice actors are unfortunately not included in this dataset, although they are listed on the MAL website. However, we scrap them using the Jikan API, an unofficial MyAnimeList API, to complete our data.

4 Project structure

4.1 Tools of the trade

To build our web application, we use the following frameworks and technologies:

- ReactJS: a Javascript library for declarative, reactive user interfaces. ReactJS implements a MVVM pattern around functional principles which describes UI as a function of state. We leverage React to create isolated, reusable components that are composable, and focus on data changes and UI templates that are automatically re-rendered when state changes. With JSX, we can express declarative user interfaces with an HTML-like syntax augmented with inline Javascript features. Its hot-reload live server allows us to develop new features quickly with automatic stateful refreshes.
- Babel: a Javascript compiler with enhanced ECMAScript 2015+ features that outputs browser-compatible ES5 code
- Webpack: a module bundler that performs packing and tree shaking on all Javascript, assets and dependencies to create optimized bundles. For instance, to use a Javascript library or a styling framework, one would usually need to link the complete JS or CSS files, while Webpack allows us to cherry-pick code, assets and styles with ES6 `import` statements and prune away all unused features.
- SASS: a CSS preprocessor with features such as variables, imports and nesting
- D3 and the React D3 wrapper: Javascript library for documents manipulation, used to generate graph stubs
- amCharts a Javascript charting library
- react-router-dom: a collection of navigational React components that emulate URLs and pages for client-side applications
- FontAwesome: a vectorized iconography toolset
- wow.js and Animate.css: CSS transition animations triggered on viewport entry
- MaterialUI and Bulma for selected form inputs styling
- Reset.css: a reset stylesheet to ensure cross-browser consistency of default styling
- Github pages and gh-pages: to easily deploy our React web app on Github pages

4.2 Architecture

The composable nature of React components allows us to naturally separate our implementation in distinct modules. We make heavy use of Higher-Order Components which allow us to easily extend any other component by wrapping it into a function call. We describe here the file structure of the project in the `app/src` root directory:

- `App.js`: The entry point of our React application. It declares the router of pages and augments its components.
- `Config.js`: Global configuration with lifecycle-aware hooks, which allows us to filter the displayed data.
- `animation`: React components and styles related to global animations, implemented as higher-order components.
- `components`: Reusable custom assets and layout (higher-order) components such as loading screens, menus, content view and sidebars.
- `pages`: React components for each page and visualization. They are described in greater detail in the following sections.
- `utils`: Reusable functions in the project

5 Visualizations

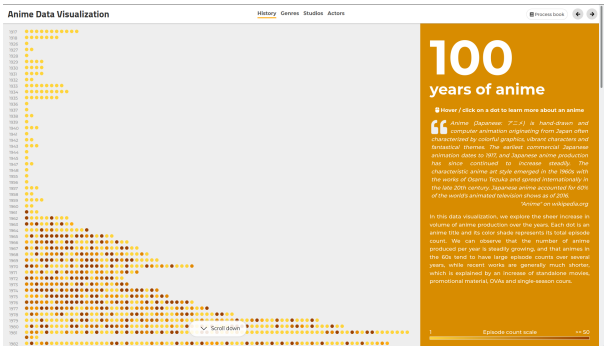
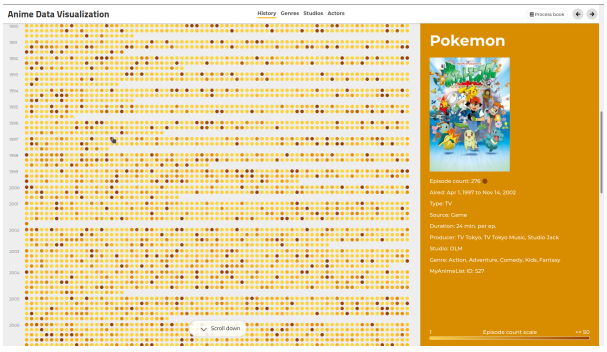
5.1 Histogram

100 years of anime

With this visualization, we want to showcase the huge amount of existing anime and the increase in volume of anime production over the years. For this visualization, we were inspired by the Github contributions calendar but scaled up to the thousands of anime titles in the dataset. We represent each anime by a dot, which color shade is derived from its episode count. The user is able to hover over every dot of the visualization to preview details about an anime, and click on one to persist it in the sidebar.

We built it from scratch using only standard `divs` and React's state management. We fetch the data `history.json` and save it as state of the component once ready. Then, `render` is automatically triggered, and the histogram consists of multiple container `divs` of class `row` for each year, containing multiple `divs` of class `dot` for all anime produced during that year. Each `dot` is a circle which color is shaded given the following color scaling: `(maxShade - minShade) / (maxThreshold - minThreshold) * (episodeCount % maxThreshold) + minShade` given our shading function defined in `ColorUtils.shade`, with base color `#F59B00`. Each dot is registered an `onMouseEnter`, `onMouseLeave` and `onClick` listener to change its respective state in the sidebar, which will then display the selected anime details. We add CSS transition classes for additional animations when rows enter the viewport.

One tricky part in the implementation was to make sure that the scrollable zones of the sidebar always behaved correctly, given the fixed scale gradient in the bottom right corner. We were worried about the performance of the browser given so many event listeners, but the production build proved to be fast enough, although somewhat CPU intensive.

Histogram visualization: main	Histogram visualization: click on anime
	

5.2 Bubble chart

43 genres to classify them all

The goal was to show the distribution of genres across animes. Given the non-negligible number of genres, we thought a bubble chart would be a good fit as it quickly highlights the most represented genres and how they compare to each other.

From the main anime dataset, we were able to directly access the genres of each anime. Inspired by some d3 example, we were able to generate the positions and radii for the bubbles.

To illustrate the genres, we decided to display the image of the anime with the most user favorites for each genre. However by doing so, if two genres share the same anime as most favorite, then they will also both show the same illustration. For instance, all genres listed by the most favorite anime title will share the same image. To prevent that behaviour, we select the image of a genre in decreasing order of its population by

selecting an anime only if it has not yet been selected by another genre. Although each genre may hence not display its first most popular anime, we thus maximally increase the image diversity to avoid dull repetitions. Some of the image links were broken from the dataset, but given there were only a few of them, we manually replaced them.

We also wanted to give the ability for a user to filter the genres that are displayed. To do so, we implemented a dropdown list of toggle-able buttons, each corresponding to one genre. We also provide an `ALL` button and a `NONE` button to quickly show all the bubbles or unselect everything so that one can then precisely choose the bubbles he is interested in. Every time the genre selection changes, the bubble positions and radii are recomputed. The bubbles then smoothly transition by translation and scaling from their previous position to the new one.

Another feature we wanted to include is the display of the most popular animes for each genre on demand. In order to increase the interactivity of the visualization, we implemented the feature with a zoom into the selected bubble, so that it covers most of the screen. We then create another bubble chart inside consisting of the most popular animes for that genre. The user can then jump from one genre bubble to another by clicking on its neighboring genres. If there are animes that appear in both genres, they will also smoothly travel from the first genre to the other.

A first implementation of this feature was to simply focus on the selected genre and scale every other bubble alongside proportionally. However, this proved to cause performance issues when selecting small genre bubbles as it would then scale the bigger genres to enormous sizes. The issue was almost unnoticable on Firefox but Chromium-based browsers had a hard time dealing with it. To address the issue, we simply push/offset the bubbles by the size of the bubble chart so that they lie close to the boundary and remain at the original size, instead of scaling them up. The difference between both implementations is shown below.

Bubble chart zoom: version 1	Bubble chart zoom: version 2 (final)
	

Finally, depending on the bubble that the user is hovering / clicking on, information about the respective chart / genre / anime is displayed in the sidebar.

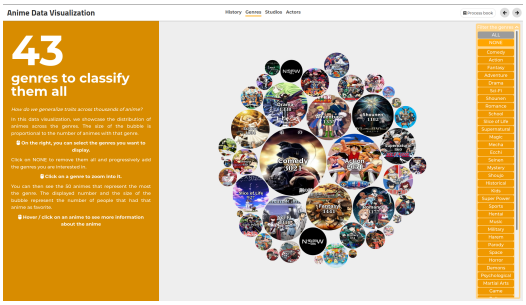


Figure 1: Bubble visualization

5.3 Sankey diagram

711 studios captivating the world

For this diagram, the aim was to show information about the most consequential anime studios. A Sankey diagram allows us to highlight the affinity of a given studio to the genres of the animes it produces while simultaneously comparing them to their user ratings, which are sorted per score mean (rounded to the closest integer).

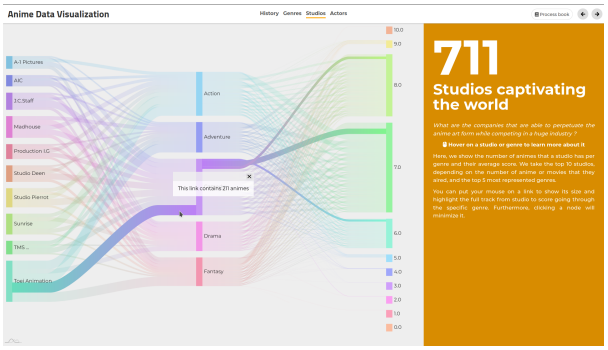
For this interactive diagram, we used the `amcharts` library, as documented here. Its rich API offers us many possibilities for representation, interaction and animation of the data out-of-the-box.

The main difficulty in this implementation was to show the connection between each group of links when hovering on it, as we want to increase the opacity on one hand of the link from a studio to a genre, and at the same time from that genre to all its attached ratings on the other. To do so, we use an identifier for each link which consists of the concatanation of the name of the studio (2 first letters and the last) with the genre name and a number which discriminates if the link connects to a genre or a rating (for instance, the link going from Toei Animation to Action will have its `id=TonAc-0` and for the same studio going to a note `id=TonAc-1`). Then we can extract the first part of the `id` to know if it belongs to the same group, and highlight all the other that share the same identifier when hovering on any of the associated links.

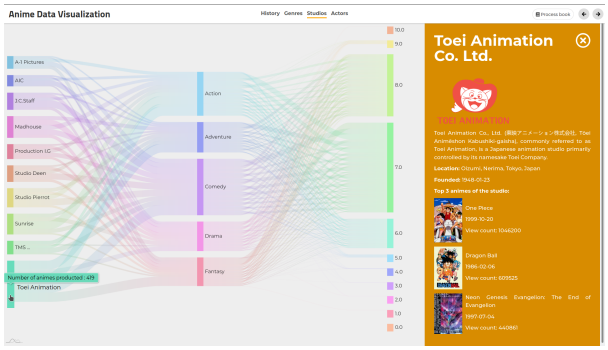
We arbitrarily chose the number of studios and genres present in the diagram, and stroke a balance between information density and performance. Indeed, each new node adds a sizable number of links, inherently affecting performance. After some testing, we settled on showing 10 studios (which have produced the biggest amount of anime titles) and 5 genres (which contain the most anime) to keep the diagram relevant.

Similarly to the other diagrams, we put additional information in the sidebar depending on what node is hovered, and show related information for each studio and a short description for each genre.

Sankey visualization: main



Sankey visualization: hover on studio



5.4 Chord diagram

11292 voices giving life to characters

Finally, the last diagram is about the voice actors, their popularity and links to other actresses or actors. This was done by using a chord diagram, so that we could easily have a node per actor and a link if they played in the same anime or movie at least once.

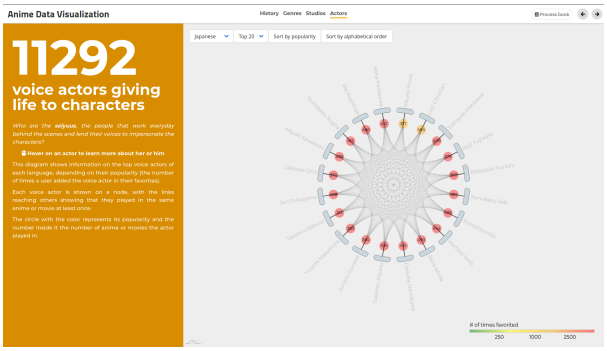
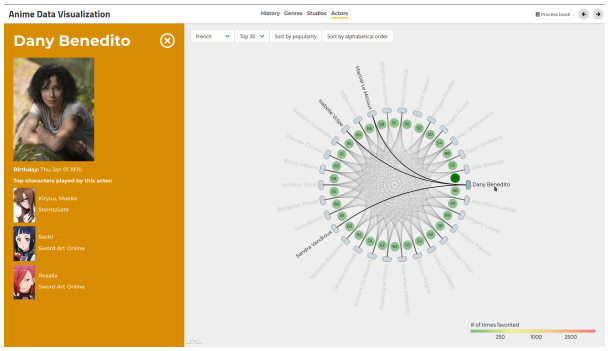
Again, we used `amcharts` library with the documentation of the chord diagram here.

We sorted the voice actors depending on the language they dub with a button to be able to go from one language to another, and chose the top ones after the number of `member favorites` they have. We also created a dropdown list to change the number represented, by using a dictonary on the dataset.

The issue with this diagram was on the script to fetch and sort the data, to take into account all possibilities

of dropdowMenus. We had to fetch every anime information to find the actors per language and character, to be able to retrieve then information per actor, before sorting them, taking the different top ones per language, checking who played in the same anime than another one and putting informations on a JSON file for each node and link each pair. Another file was made for the sidebar which stores the informations for each voice actor, after finding the top 3 characters they dubbed (depending on the popularity of the anime the character was in).

Then, we thought about showing more information directly on the diagram, to have a more general view of it all. A circleBullet to show the popularity and number of anime/movies dubbed by each, as well as the possibility ot highlight every actress or actor linked to the one hovered. With all of that, someone looking at the diagram can directly see insightful information in a single glance. Finally, we added buttons to change how the diagram is sorted or which top and language we want to show, which was done by getting the right data from its dictionary (for example `top 10 -> "English" -> data`) and sorting it in a different way, as `amcharts` will create nodes depending on their appearance on the dataset.

Chord visualization: main	Chord visualization: hover on voice actor
	

6 Future work and limitations

The main issue of the visualizations is that most of them are not adapted to display correctly on mobile. We could work on it and show a modified version of the website when using a phone to avoid conflicts in the display of the charts.

Furthermore we used only one dataset based on a website that is more popular in some countries than other, giving biased information depending on the general anime culture of each region. For example this dataset has Poland and Brazil being on the top number of users while France is on the bottom, even though the latter is one of the top countries worldwide watching anime. We could have used more datasets coming from other platforms: Crunchyroll, ADN or AniList. However, using their datasets would have been more challenging and taken more time to scrupulously gather and merge informations.

Each of the visualizations could be further improved by adding specific features:

- Histogram:
  - Add an anime search bar so that the user can easily find the animes he is interested in.
  - Add a year range selector to only show the range of years you are interested in.
  - Show the top anime per year or range of years.
- Bubble:
  - Add an anime search bar to only display the genres of the chose anime.
- Sankey:
  - Add a list of studios and genres to choose from, to have them appear on the diagram.
- Chord:
- Adding a threshold about the number of anime played in by two seyuus to create a link between them.

Otherwise, we could show other diagrams to get more informations about (by a worldmap, ...) or evolution of the popularity of animes through time, or for example the evolution of genres, studios, and so on.

7 Peer assessement

Each member worked mainly on its main visualization(s) and the necessary data processing. We all contributed in the general visualization ideas and decisions that lead to the final website result.

Alexandre Chau:

- Web app setup, global structure, layout, styles and animations
- Histogram (data processing and visualization)
- Helped integrate elements in the React ecosystem
- Website design

Joachim Dunant:

- Chord diagram (data processing, visualization and animations)
- Sankey diagram (data processing, visualization and animations)
- Website design

Pedro Torres Da Cunha:

- Bubble chart (data processing, visualization and animations)
- Helped with data processing (pandas)
- Website design

8 Conclusion

In this project we worked on different visualizations about anime and their specifity, to show how interesting and complex they can be, who are their main actors, their popularity across different metrics and to get a general view of all of them. Going for a page template across the website give us the possibility to show more detailled information about each diagram, while having a general narration between them. We preferred this approach than a single page, to avoid the confusion of scrolling and to have tabs to go from one diagram to the other in an instant.

Hopefully you could explore and experiment with the different visualizations to have an overall fun and engaging experience that could inspire or even tempt you to know more about this industry and anime in a general way.