TDE
DIGITAL CREATIVES IN SPORTS

F1 Available Data Analysis

Improving the Grand Prix experience for F1 viewers at home

S8 Graduation FHICT

4 Sept 2023 - 16 Jan 2024

By Jordi Franssen

18 September 2023

www.tde.nl

Introduction

One of this project's research questions asks what data is available from an F1 race. The answer for this question is required for the Double Diamond's ideation phase, where a solution for the problem statement will be defined.

This question will primarily be answered using library research. However, RacingNews already makes use of an API for Formula 1 data. This API is undocumented however, and not much can be found online about it. So information about this API will have to come from TDE itself.

Also, in this research, I'm also looking for data that could be interesting to visualize with ThreeJS. Think of GPS data of Formula 1 cars for example.

F1 Data sources

There are multiple data sources that provide F1 racing data. Below is the undocumented API summarized plus two data sources found online.

Undocumented FIA API

This is the API RacingNews365 already uses in their website. According to TDE, this API actually belongs to FIA itself, the company behind Formula 1. This is an undocumented API however, which means that there's very little known about the use of this datasource. This also makes using this API risky, as it's unknown for how long this API will be live. This is also a live-data api that provides information in real-time.

Data in this API is documented in JSON, which is perfect for web applications. An entire racing weekend generates about 1.5gb of data which is stored in thousands of JSON files, each containing about 5000 lines of code.

Below is a list of what information can be found in these files:

- Information about the racing session including the name of the race, location, beginning and end time, starting time of each lap and track status during the race.
- A list of all the drivers and corresponding information.
- Timing data per driver including fastest lap times, positions, speed traps and intervals between drivers.
- Pitstop data including tyre compounds and tyre age.

Ergast API

In contrast to the undocumented API above, Ergast is an API that provides historical racing data and is therefore not a real-time datasource. The API provides data all the way from the beginning of the world championships, starting in 1950. The API outputs both JSON and XML which works well with web applications.

The major drawback from this datasource is that it will be shut down after the 2024 season, so this won't be a long term solution. The API is free to use however and doesn't require any API keys.

Data available in this API:

- An overview of each season since 1950 and corresponding racing schedule.
- Results of each race.
- Results of each qualifying.
- Results of each qualifying race.
- Driver standings between each race.
- Constructor standings between each race.
- Driver information.
- Constructor information.
- Circuit information.
- Finishing status.
- Lap time data (available from 1996).
- Pitstop data (available from 2012).

Fast-F1

https://docs.fastf1.dev/index.html

Fast-F1 could be seen as a combination between the two data sources discussed above. It uses Ergast to provide historical racing data, but also provides live-timing data. The drawback of this api is that it's quite complex and it's built around Python, which is not directly suitable for web applications. Also, how live is this live data? There's possibly a delay between the general broadcast and data provided in this API.

Data available in this API:

- Event data including event names, countries, locations, dates, scheduled starting times (previous and current season including upcoming events).
- Results including driver names, team names, finishing and grid positions, points, finishing status.
- Timing data including sector times, lap times, pit stops, tyre data and much more.
- Track status including flags and safety car.
- Session Status.
- Race control messages including investigations, penalties and restart announcements.
- Telemetry data including speed, rpm, gear and track position.
- Track markers including corner numbers, marshall sectors and marshall lights.
- Historical racing data by Ergast.

Test data for development

One issue with this project is that it finishes well after the last race of this years' season, which is in November. This means that a solution needs to be found to be able to test the product and demonstrate it when we're using live data. And that we have to somehow "record" data from a live datasource during a race, in order to "replay" it for testing and demoing.

After a search online, it became clear that an all in one solution isn't really available. Most mock API's only record API requests and not a continuous datastream of a dynamic datasource. Therefore, I decided to build my own solution instead.

I figured that it would be possible to write an API response to a local JSON file in a simple Node server. If this can be done with a similar refresh rate of that of the original API, then this series of JSON files is essentially a recording of a datasource.

Next, to replay this recording, I built a REST Node server that reads the JSON files with the same interval as the recording. This way, I was able to build myself a solution in less than 100 lines of code.

The proof of concept I built currently records the position of the International Space Station. The code can be found in this Git repo:
https://github.com/Lampenkap007/API-recorder

Conclusion

It seems like two types of data can be used for this project: live data and historical data. Further research will tell what data will eventually be used. However, it's good to know what data is available to get a better understanding of what's possible to build.

The three APIs found are all very interesting. Although Fast-F1 is the most reliable datasource, it isn't an option as it's unsuitable for web applications. The other two APIs will work fine with web apps, but the undocumented API is risky to use, and Ergast will shut down next year.

Both Ergast and the undocumented API don't provide any location data that can be used in ThreeJS. Further research will tell what data can be visualized using ThreeJS and how.

Rectification

October 30 2023

Written above it is concluded that there isn't any exact location data from Formula 1 cars available that can directly be used by ThreeJS. Therefore, Yuka.JS would have to simulate the car behavior based on timing data like lap times, sector times and velocities. However, this conclusion turned out to be false. During an update with John, a colleague was able to tell that there is in fact exact location data available from the FIA API, it just wasn’t present in the data samples I received. This alters the course of the development phase massively, as it will now be much easier to position the cars with ThreeJS. Also, I found that complete datastreams of this data can be downloaded, so recording the data isn't necessary anymore as well.

Summary

One of this project's research questions asks what data is available from an F1 race. The answer for this question is required for the Double Diamond's ideation phase, where a solution for the problem statement will be defined.

This question will primarily be answered using library research. However, RacingNews already makes use of an API for Formula 1 data. This API is undocumented however, and not much can be found online about it. So information about this API will have to come from TDE itself.

Also, in this research, I'm also looking for data that could be interesting to visualize with ThreeJS. Think of GPS data of Formula 1 cars for example.

It seems like two types of data can be used for this project: live data and historical data. Further research will tell what data will eventually be used. However, it's good to know what data is available to get a better understanding of what's possible to build.

All three data sources found are interesting, although Fast-F1 is the most reliable data source, it cannot be used in the project, as it isn't suitable for web applications. Ergast is unsuitable to use as well as it will probably shut down next year. This will result in an unsustainable product. The undocumented API is also risky to use as we don't know what will happen with this data source in the future. However, this is the most promising data source for the project, as RN365 already uses this data source for their live timing data, and this data source also provides positioning data of Formula 1 cars.

Learning Outcome Clarification

- Learning Outcome 1: Professional Duties
- Learning Outcome 2: Situation-Orientation
- Learning Outcome 3: Future-Oriented Organisation
- Learning Outcome 4: Investigative Problem Solving

This deliverable is a professional duty on a bachelor level in the activities of Analysis and Advise as I analyzed available data sources and advised on the use of these data sources in the project. This is in line with IT-area User Interaction. Therefore, Learning Outcome 1: Professional Duties applies.

This deliverable is relevant and valuable as it plays a role in the orientation phase of the project and the results will have an impact on the concept that's being formed in the concepting phase. I also worked in a methodological and structured way. Therefore, Learning Outcome 2: Situation-Orientation applies.

In my advice for the best data source to use, I took sustainable development into consideration, as one of the data sources found will result in an unsustainable product when used. I included this in my advice for the data source to use in my project. Therefore, Learning Outcome 3: Future-Oriented Organization applies.

This deliverable is a research method on ictresearchmethods.nl. and it is an effective approach to find out what data sources are available for my project. I also identified problems with the data sources found and used the results from this research as an input for the focus group discussion. Therefore, Learning Outcome 4: Investigative Problem Solving applies.