Overview

1. The code initializes a Google Map with specific properties, such as the center coordinates and zoom level.

2. It makes an AJAX request to retrieve GeoJSON data from a JSON file (`/data/geo.json`).

3. If the AJAX request is successful and the retrieved data contains at least one feature, it adds markers to the map based on the coordinates and properties provided in the GeoJSON data.

4. Each marker is associated with a click event listener that opens an info window when clicked. The info window displays information about the location, including its name, thumbnail image (if available), description, and website link (if available).

5. The code also includes error handling for the AJAX request.

6. There are two buttons (`satelliteButton` and `streetButton`) that allow the user to switch between satellite and street view on the map. Clicking on either button updates the map's mapTypeId accordingly and adds/removes the "active" class to/from the buttons for styling purposes.

7. The code uses jQuery to retrieve product data from a JSON file (`/data/items.json`) and dynamically generate HTML content based on the retrieved data. The generated content includes product images, names, and prices.

8. The generated product content is then inserted into a swiper container (`#swiper\_sliders`) using jQuery.

9. Finally, an instance of Swiper is created to initialize the swiper container, providing optional parameters like loop, pagination, slidesPerView, and spaceBetween.

Overall, this code sets up a Google Map with markers and info windows, allows the user to switch between satellite and street view, retrieves product data and displays it in a swiper container.

Swiper

Parameters explanation

1. `loop`: This parameter enables the loop mode, allowing the swiper to continuously slide from the last slide to the first slide and vice versa. It creates an infinite loop effect. In this case, it is set to `true`.

2. `pagination`: This parameter enables the pagination feature, which adds clickable pagination bullets at the bottom of the swiper container. The `el` property specifies the HTML element or CSS selector where the pagination should be rendered. In this case, it is set to `.swiper-pagination`, indicating that the pagination bullets will be placed inside an element with the class name "swiper-pagination".

3. `slidesPerView`: This parameter determines the number of slides that should be visible in the swiper container at the same time. It can be a fixed number or a fractional value. In this case, it is set to `3`, indicating that 3 slides will be visible simultaneously.

4. `spaceBetween`: This parameter sets the space (in pixels) between each slide in the swiper container. It controls the gap between adjacent slides. In this case, it is set to `20`, indicating that there will be a 20-pixel gap between slides.

5. `centeredSlides`: This parameter determines whether the slides should be centered in the swiper container. If set to `true`, the active slide will be centered in the container. If set to `false`, the active slide will be aligned to the left of the container. In this case, it is set to `false`.

Overall, these parameters configure the behavior and appearance of the Swiper instance, allowing for a looped carousel with pagination, 3 visible slides at a time, a gap of 20 pixels between slides, and no centering of slides.