Projet programmation web

Xavier Barbeau, Chloé Decoust,
Leslie Planet, Hélène Vizoso

François Garnier

2024 - 2025

Sommaire

Introduction

Langage utilisé :

Démonstration de l’outil

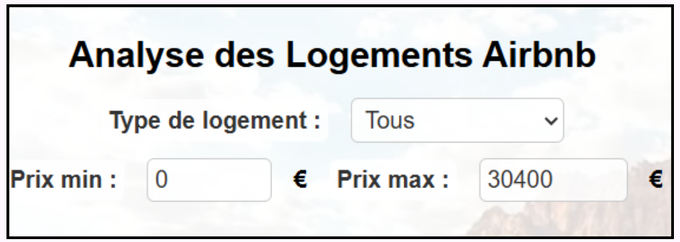
Commentaires du code

Onglet : Analyse des logements

```
// Met à jour les graphiques en camembert
function updateRoomTypePieCharts(data) {
    const roomTypeCount = countRoomTypes(data);
    const sortedRoomTypes = Object.entries(roomTypeCount).sort((a, b) => b[1] - a[1]);

    if (sortedRoomTypes.length === 0) {
        console.warn("Aucun logement ne correspond aux filtres.");
        return;
    }

    const mainCategory = sortedRoomTypes[0];
    const otherCategories = sortedRoomTypes.slice(1);
    //graph1
    const trace1 = {
        labels: sortedRoomTypes.map(item => item[0]),
        values: sortedRoomTypes.map(item => item[1]),
        type: 'pie',
        textinfo: 'label+percent',
        hoverinfo: 'label+value+percent',
    };
```

```
// Création de la carte
let map = L.map('map').setView([48.8566, 2.3522], 12);
L.tileLayer('https://{s}.tile.openstreetmap.org/{z}/{x}/{y}.png', {
    attribution: '&copy; OpenStreetMap contributors'
}).addTo(map);

let markersCluster = L.markerClusterGroup();


function updateMap(data) {
    markersCluster.clearLayers();
    data.forEach(item => {
        if (item.latitude && item.longitude) {
            let logementNom = item.name ? item.name : "Nom inconnu";  // Si pas de nom, afficher "Nom inconnu"
            let logementType = item.room_type ? item.room_type : "Type inconnu";  // Si pas de type, afficher "Type inconn

            let marker = L.marker([item.latitude, item.longitude])
                .bindPopup(`<b>${logementNom}</b><br><i>Type:</i> ${logementType}<br><i>Prix:</i> ${item.price}€`);

            markersCluster.addLayer(marker);
        } else {
            console.warn("Coordonnées manquantes pour le logement:", item);
        }
    });
    map.addLayer(markersCluster);
}
```

```
// Remplir le filtre de type de logement
function RoomTypeFilter() {
    const roomTypeFilter = document.getElementById('room_type_filter');
    const roomTypes = [...new Set(listingsData.map(item => item.room_type))].sort();

    roomTypeFilter.innerHTML = `<option value="">Tous</option>`; // Option "Tous"
    roomTypes.forEach(type => {
        roomTypeFilter.innerHTML += `<option value="${type}">${type}</option>`;
    });
}
```

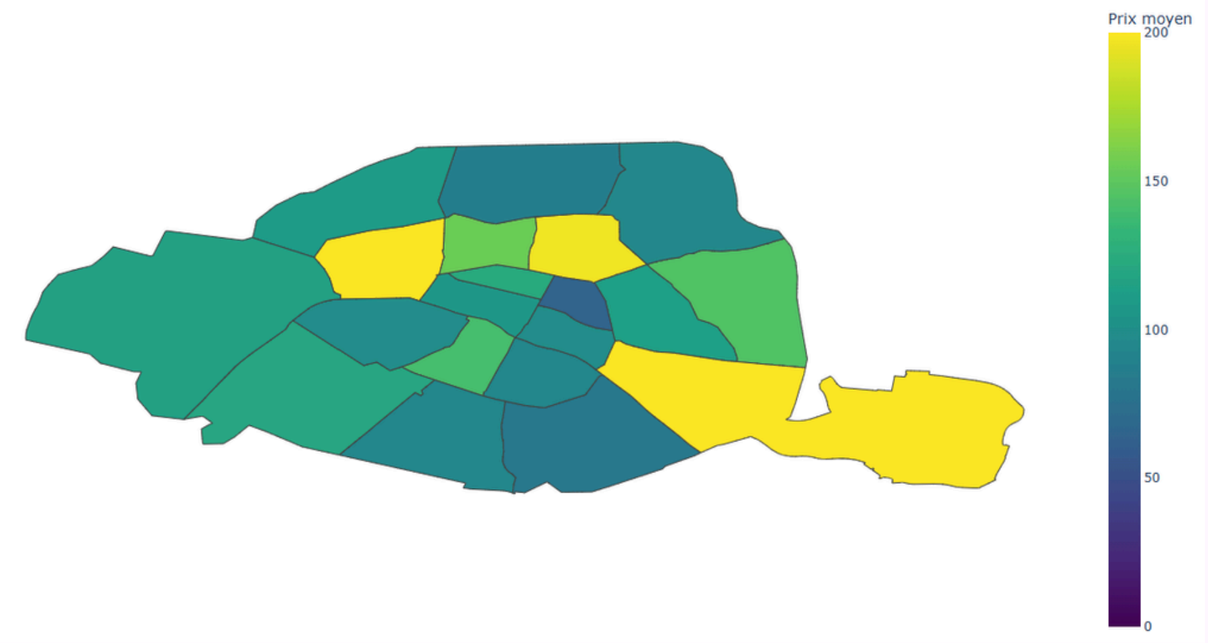



```
// récupère le prix maximum
function getMaxPrice(data) {
    return Math.max(...data.filter(item => item.price !== null).map(item => item.price));
}
```

Commentaires du code

Onglet : Les cartes

```
fig = px.choropleth(df,
                    geojson=neighbourhood,
                    locations="neighbourhood",
                    featureidkey="properties.neighbourhood",
                    color="price",
                    color_continuous_scale="Viridis",
                    range_color=(0, 200),
                    labels={'price': 'Prix moyen'})


fig.update_geos(fitbounds="locations", visible=False)

fig.update_layout(margin={"r": 0, "t": 0, "l": 0, "b": 0})

fig.write_html(os.path.join(output_dir, "carte_interactive.html"))
```

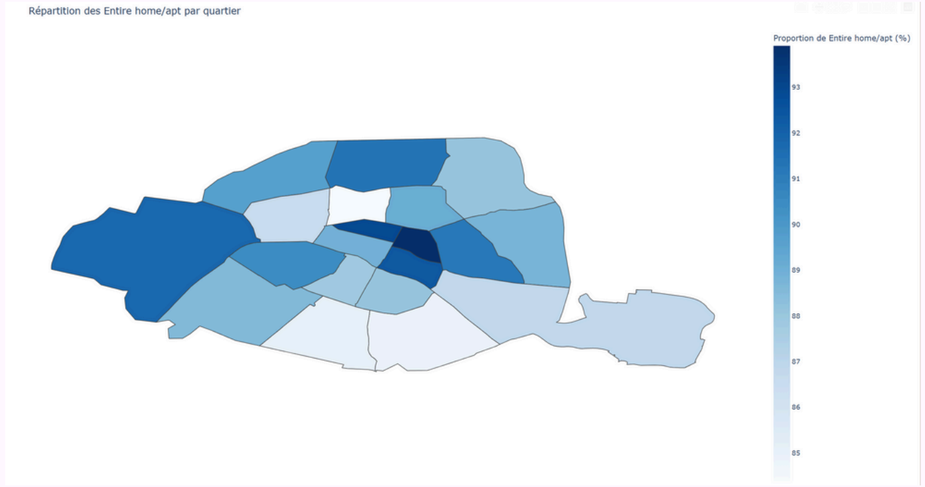


Commentaires du code

Onglet : Arrondissements

```python
for room_type in df_merge["room_type"].unique():
    df_filtered = df_merge[df_merge["room_type"] == room_type]

    # Definir une echelle adaptee
    min_value = df_filtered["proportion"].min()
    max_value = df_filtered["proportion"].max()

    # Utiliser une echelle monochrome
    color_scale = color_map.get(room_type, "Greys")  # Defaut = Gris si inconnu

    fig = px.choropleth(df_filtered,
                        geojson=neighbourhood,
                        locations="neighbourhood",
                        featureidkey="properties.neighbourhood",
                        color="proportion",
                        color_continuous_scale=color_scale,
                        range_color=(min_value, max_value),
                        labels={'proportion': f'Proportion de {room_type} (%)'},
                        title=f"Repartition des {room_type} par quartier")

    fig.update_geos(fitbounds="locations", visible=False)
    fig.update_layout(margin={"r": 0, "t": 50, "l": 0, "b": 50})

    # Nettoyer le nom du fichier
    clean_room_type = re.sub(r'[^\w\-_]', '_', room_type)
    filename = f"./C_L/cartes/carte_{clean_room_type.lower()}.html"

    fig.write_html(filename)
    print(f"Carte sauvegardee sous '{filename}' (echelle : {min_value} - {max_value}, couleur : {color_scale})")
```

```html
<div class="card">
    <h3>Carte des Prix</h3>
    <iframe src="./cartes/carte_entire_home_apt.html" title="Carte des Prix"></iframe>
</div>
```



Conclusion

Ce que l’on peut améliorer