*Formation CQP Bloc 2 - U5 - S21*

# Visualisation de données géospatiales avec Kano

https://kalisio.com

Kalisio

# Un cas d'usage concret

*Créer une cartographie liée à la pandémie COVID-19*

# Objectifs du fil rouge

➜ Calculer le taux d'hospitalisation par département et le représenter sur une carte avec une symbologie adaptée

a. Recenser les données disponibles
   - nombre d'hospitalisations
   - limites administratives
   - population

b. Définir les traitements appropriés
   - extraction des données
   - transformation des données

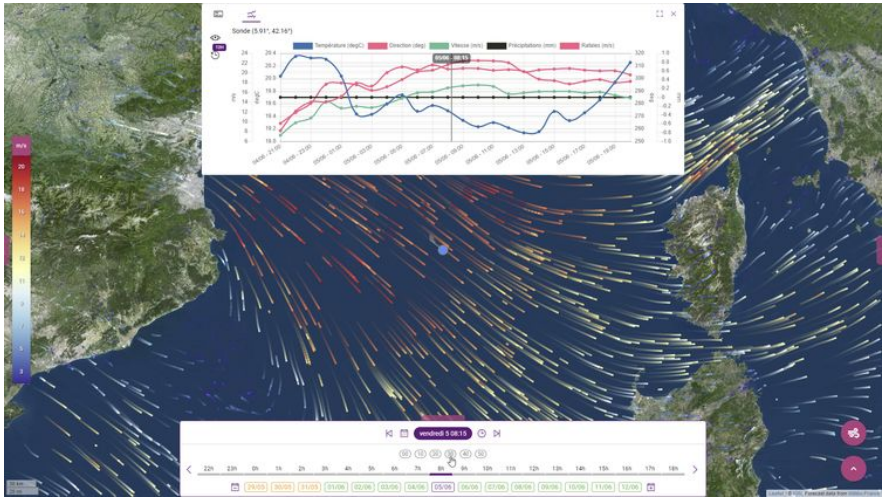
c. Définir la représentation
   - symbologie
   - configuration de l'outil de visualisation

# Kano: capabilities

https://kalisio.github.io/kano

- ➜ Provides advanced visualization capabilities in 2D and 3D using WebGL
- ➜ Enables real-time tracking of geospatial assets
- ➜ Designed to be integrated in third-party applications and customized by developers

KANO



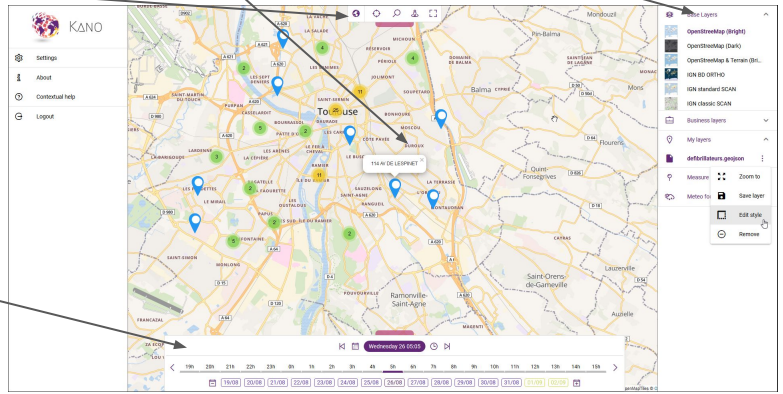
KALISIO

# Kano: capabilities

- Composed of two main activities (i.e. entry point for interacting with the user)
  - **2D activity** to visualise and interact with data on a *virtual map* in 2D
  - **3D activity** to visualise data and interact with on a *virtual globe* in 3D

- The view of each activity can contain
  - a **basemap** (i.e. a background context)
  - a set of **data layers** (i.e. business specific data)
  - an **extent** (i.e. the part of the Earth currently on the screen)
  - **navigation tools** to pan and zoom, search an address, etc.

- Handles both space and time information to visualize **time-varying** geospatial phenomena
  - user navigates back and forward in time using a **timeline**




https://kalisio.github.io/kano/guides/getting-started.html

# Exercices

➜ Déployer l'application en mode container

a. Définir un fichier docker-compose pour lancer une instance de MongoDB
   - image `mongo:3.6.5`
   - créer un volume pour conserver les données
   - ajouter un réseau pour la rendre accessible depuis le container de l'application

b. Définir un fichier docker-compose pour lancer une instance de Kano
   - image `kalisio/kano:dev`
   - définir les variables d'environnement requises
     - `APP_SECRET`
     - `LAYERS_FILTER`
     - `CESIUM_TOKEN`
   - ouvrir un port sur l'hôte, e.g. `3000`, associé au port `8081` du container
   - ajouter le même réseau que le container MongoDB

c. Se connecter à l'application sur l'hôte, e.g. `http://localhost:3000`

# Exercices

➜ Solution

mongodb.yml

```yaml
version: '3.3'

services:
  mongodb:
    image: mongo:3.6.5
    volumes:
      - mongodb_app:/data/db
    networks:
      - app

volumes:
  mongodb_app:

networks:
  app:
```

app.yml

```yaml
version: '3.3'

services:
  app:
    image: kalisio/kano:dev
    environment:
      - APP_SECRET=xxx
      - LAYERS_FILTER=OSM_BRIGHT OSM_DARK IMAGERY CESIUM_ELLIPSOID K2
      - CESIUM_TOKEN=yyy
    ports:
      - "3000:8081"
    networks:
      - app

networks:
  app:
```

Lancer => `docker-compose -f .\mongodb.yml -f .\app.yml up -d`

Stopper => `docker-compose -f .\mongodb.yml -f .\app.yml down -v`

# Exercices

➜ Configurer une couche permettant d'afficher les données COVID-19

a. Lancer un serveur web permettant d'accéder aux données COVID-19
   - utilisation du module NodeJS http-server (`npm install -g http-server`)
   - utiliser un port différent de celui de Kano sur l'hôte, e.g. `4000`
   - activer le CORS

b. Définir un fichier `covid19-layers.js` contenant la définition de la couche
   - `featureId` = identifiant unique des features
   - définir les traductions fr/en pour le nom et la description
   - ajouter une icône, l'attribution et un tag
   - définir les options pour Leaflet
     - type de couche = `geoJson`
     - `source` = URL des données COVID-19
   - voir exemples des couches par défaut de Kano

c. Modifier le fichier docker-compose lançant Kano afin d'y injecter la nouvelle couche
   - mettre à jour `LAYERS_FILTER` en conséquence

d. Se connecter à l'application sur l'hôte, e.g. `http://localhost:3000`

# Exercices

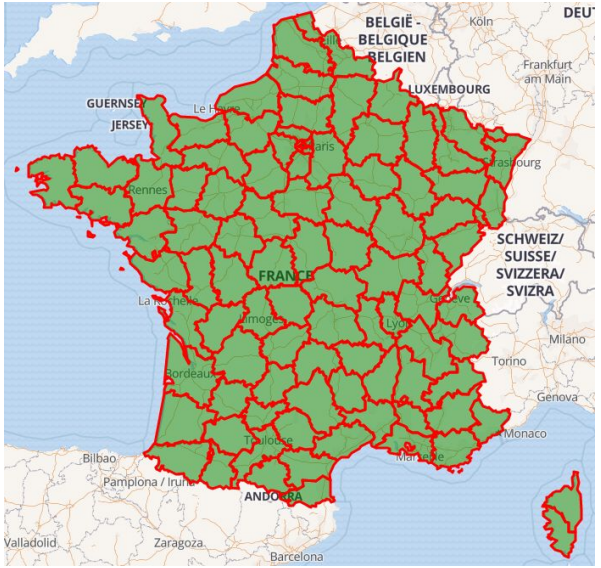
➜ Solution `http-server --port 4000 --cors`

`covid19-layers.js`

```
module.exports = [
  {
    name: 'Layers.U5_S21',
    description: 'Layers.U5_S21_DESCRIPTION',
    i18n: {
      fr: {
        Layers: {
          U5_S21: 'U5 - S21',
          U5_S21_DESCRIPTION: 'Hospitalisations par départements'
        }
      },
      en: {
        Layers: {
          U5_S21: 'U5 - S21',
          U5_S21_DESCRIPTION: 'Hospitalizations by departments'
        }
      }
    },
    tags: [ 'business' ],
    icon: 'fas fa-atlas',
    attribution: 'Santé Publique Fance / IGN / INSEE',
    type: 'OverlayLayer',
    featureId: 'code',
    leaflet: {
      type: 'geoJson',
      source: 'http://127.0.0.1:4000/hospitalisations-departements-2020-05-01.json'
    }
  }
]
```

`app.yml`

```
volumes:
  - type: bind
    source: ./covid19-layers.js
    target: /opt/kalisio/kano/api/config/layers/covid19-layers.js
```



# Exercices

➜ Modifier la configuration de la couche pour définir un style personnalisé.

a. Modifier la configuration de la couche
   - `stroke-color, stroke-width, stroke-opacity` = tracé des linéaires
   - `fill-color, fill-opacity` = remplissage (i.e. polygones)

b. Relancer Kano avec la nouvelle configuration de couche

c. Se connecter à l'application sur l'hôte, e.g. `http://localhost:3000`

# Exercices

➜ Solution

```
stroke: '#fee8c8',
'stroke-width': 2,
'stroke-opacity': 0.5,
'fill-opacity': 0.5,
'fill-color': '#e34a33'
```

# Exercices

➜ Modifier la configuration de la couche pour afficher une bulle d'information avec le nom du département et la nombre d'hospitalisations associées.

a. Modifier la configuration de la couche
   - `tooltip` = options de la bulle d'information
     - `template` = template du contenu

b. Relancer Kano avec la nouvelle configuration de couche

c. Se connecter à l'application sur l'hôte, e.g. `http://localhost:3000`

d. Naviguer dans le temps en utilisant la timeline

# Exercices

➜ Solution


```
tooltip: {
  template: '<b><%= properties.nom %>: <%= properties.hospitalisations %> hospitalisations</b>',
  options: {
    opacity: 0.8,
    direction: 'top'
  }
}
```

KALISIO

# Exercices

➜ Modifier la configuration de la couche pour permettre de requêter les données en fonction de la date: pour ce faire utiliser le système de templating.

a. Modifier la configuration de la couche
   - `sourceTemplate` = URL template des données COVID-19 basée sur la date
   - utiliser la variable `time` pour accéder au temps courant dans le contexte

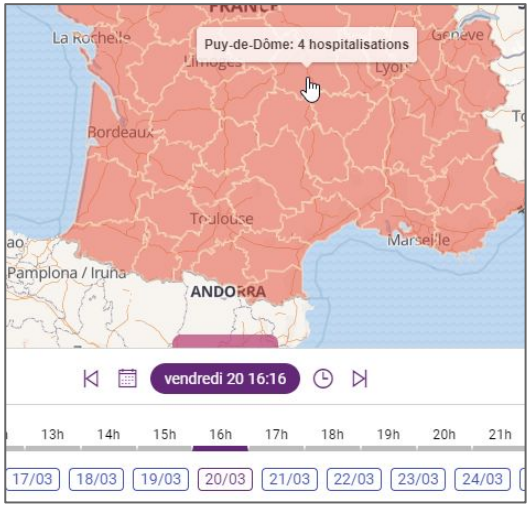
b. Relancer Kano avec la nouvelle configuration de couche

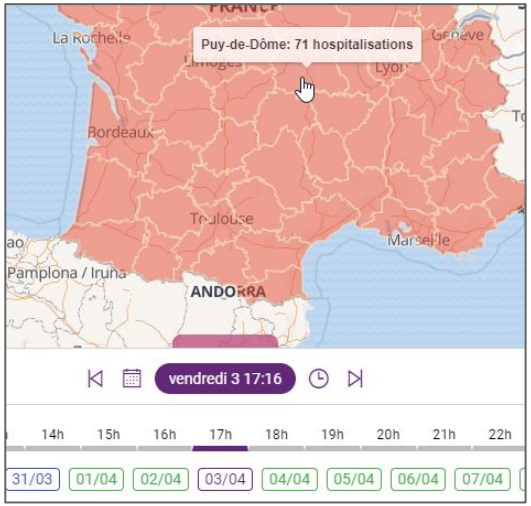
c. Se connecter à l'application sur l'hôte, e.g. `http://localhost:3000`

d. Naviguer dans le temps en utilisant la timeline

# Exercices

➜ Solution




```
realtime: true,
sourceTemplate: `http://127.0.0.1:4000/hospitalisations-departements-<%= time.format('YYYY-MM-DD') %>.json`,
```

# Exercices

- ➜ Modifier la configuration de la couche pour colorer les features en fonction du taux d'hospitalisation: pour ce faire utiliser le système de templating.

  a. Modifier la configuration de la couche
     - `fill-color` = template de la couleur de remplissage
     - Utiliser chromajs pour définir la gradient de couleur
       - utiliser la variable `chroma` pour y accéder dans le contexte

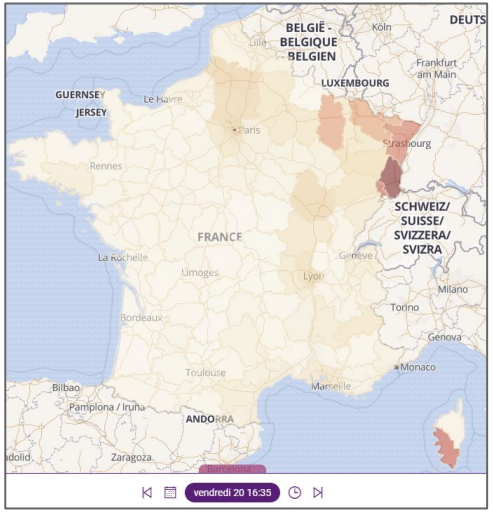
  b. Relancer Kano avec la nouvelle configuration de couche

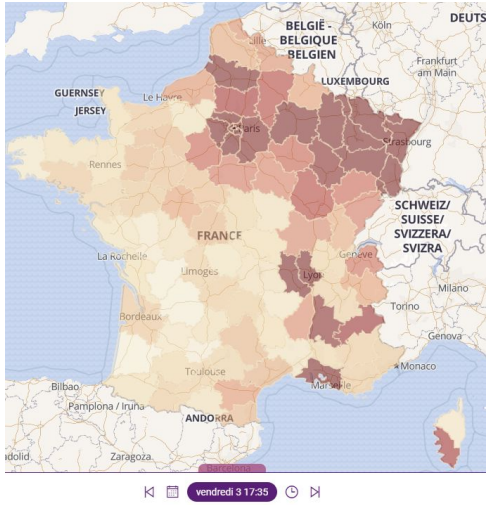
  c. Se connecter à l'application sur l'hôte, e.g. `http://localhost:3000`

  d. Naviguer dans le temps en utilisant la timeline

# Exercices

➜ Solution




```
'fill-color': '<%= chroma.scale(\'OrRd\').domain([0,50])(properties.taux).hex() %>',
template: ['fill-color'],
```

# Exercices

➜ Modifier la configuration de la couche pour la rendre fonctionnelle en mode 3D.

a. Modifier la configuration de la couche
   - définir les options pour Cesium de façon similaire à celles de Leaflet
     - `type`, `sourceTemplate`, `realtime`, `tooltip`
   - définir un style d'entité (`entityStyle`) de type `polygon` dont la hauteur d'extrusion (`extrudedHeight`) dépend du taux d'hospitalisation

b. Relancer Kano avec la nouvelle configuration de couche

c. Se connecter à l'application sur l'hôte, e.g. `http://localhost:3000`

d. Basculer en mode 3D

e. Naviguer dans le temps en utilisant la timeline

# Exercices

➜ Solution

```
cesium: {
  type: 'geoJson',
  realtime: true,
  sourceTemplate: `http://127.0.0.1:4000/hospitalisations-departements-<%= time.format('YYYY-MM-DD') %>.json`,
  entityStyle: {
    polygon: {
      outline: false,
      extrudedHeight: '<%= 1000 * properties.taux %>',
    },
    template: ['polygon.extrudedHeight']
  },
  tooltip: {
    template: '<%= properties.nom %>: <%= properties.hospitalisations %> hospitalisations'
  }
}
```

# Exercices

➜ Déployer l'application en mode développement

a. Récupérer le code de la dernière version via la KLI

b. Procéder à l'installation et au link des différents modules

c. Lancer la base MongoDB

d. Lancer la partie backend de kano

e. Lancer la partie frontend de kano

f. Se connecter à l'application

# Exercices

➜ Solution

```
git clone https://github.com/kalisio/kano.git
yarn install
yarn link
kli kano.js --clone
kli kano.js --install
kli kano.js --link
cd kano/api
yarn dev
cd kano
yarn dev
```

*Spatial information that powers up your business*

KALISIO