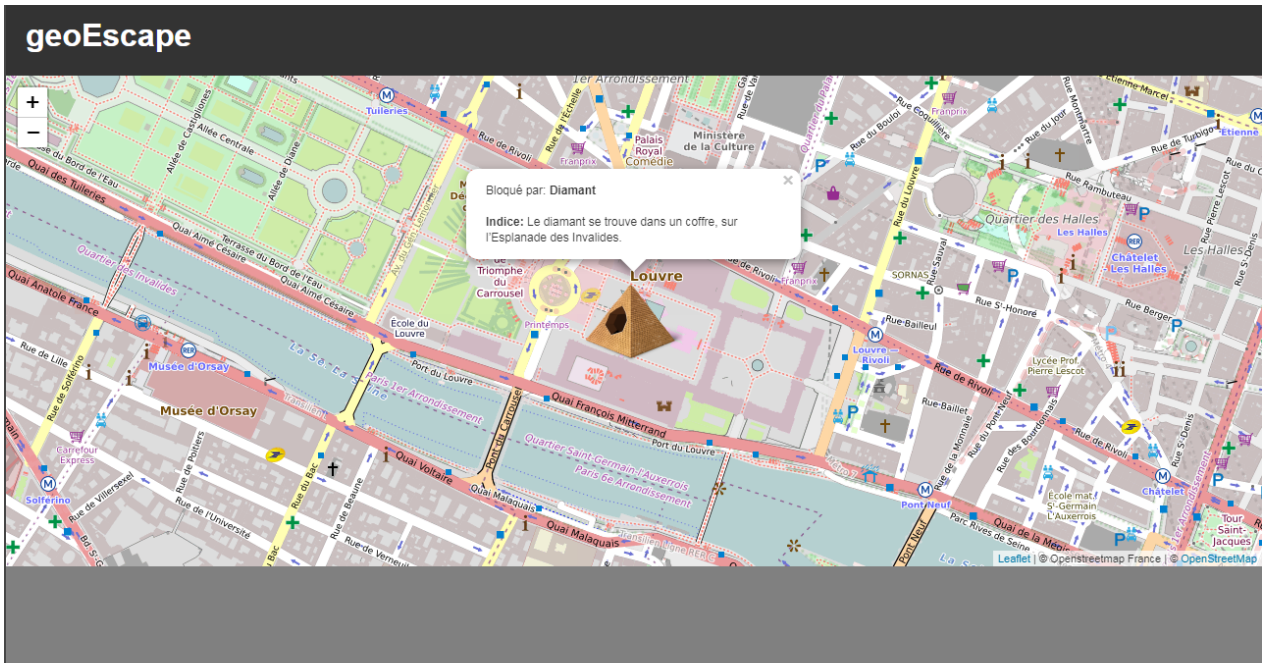
« Escape game » géographique !

L'objectif de ce TP est de créer un « escape game », notamment en résolvant des énigmes et en trouvant des objets. Le tout, sur une carte web.



Le design[1] de l'application est totalement libre. À vous donc de faire les bons choix en matière d'ergonomie, de rendu graphique, d'interaction, etc. Pensez cette application pour le « grand public », elle ne s'adresse pas à des professionnels. Demandez-vous comment vous aimeriez qu'elle fonctionne. Trouvez lui un nom.

Vous n'êtes pas contraints pour la technologie serveur (PHP, Node.js, etc.) ni par le système de base de données (MySQL, PostgreSQL, etc.).

Rendu attendu

Date de retour prévue : **jeudi 17 décembre 2020** à **23h59**

- Tous les fichiers client et serveur
  - HTML/CSS/JavaScript
  - PHP/Node.js
  - Images, Vidéos, Sons, etc.
- Export SQL de la base de données
- README pour les consignes « d'installation » des différents composants, leurs versions, etc.
- Solutions des énigmes

Soit par mail (attention, pas de format .zip ni .js sur des adresses ENSG/IGN), sur clé USB en L308 (après déconfinement) ou sur un dépôt Git (GitHub, GitLab, etc.)

Plus ce sera simple à re-déployer sur nos machines personnelles, plus on sera contents. ☺

[1] Le design au sens large, tel qu'utilisé en anglais, que l'on pourrait traduire par conception

Note : attention au copiés-collés ! Bien que ce soit un travail en groupe, cela ne signifie pas que vos fichiers doivent être identiques entre les groupes, ni même avec les années précédentes. Il est même difficile que ça puisse être le cas. Et vous devez vous douter qu'il est très facile de faire des recherches de similitudes. À bon entendeur, bon travail ! ;) Si jamais vous êtes bloqués, ou si vous avez des questions, n'hésitez pas à passer en L308 ou à nous envoyer des mails.

Création de l'énigme

Tout d'abord, créez le déroulement de votre jeu (objets cachés, objets bloqués, code, etc.). Il peut être assez simple au départ, il sera ensuite facile de le modifier. Pour ce TP, nous partons du principe qu'un objet peut conditionnellement :

- Être récupérable
- Être un code
- Être bloqué par un autre objet
- Être bloqué par un code
- Libérer un objet bloqué
- Être chargé au démarrage du jeu

Et ce n'est pas exclusif. Par exemple, un objet 1 peut être bloqué par un objet 2, et également libérer un objet 3. Par contre, certains choix peuvent s'exclure. Par exemple, un objet bloqué par un code ne l'est pas aussi par un objet.

> Il est tout à fait possible de créer d'autre type d'objets, d'autres façons de bloquer, d'autres méthodes de déblocage. Comme par exemple : déplacer un objet sur un autre objet, jeu type puzzle pour débloquer un objet, etc. Demandez-nous conseil avant de vous lancer dans de nouvelles interactions.
>
> Mais surtout, faites d'abord ce qui est demandé. Attention au hors-sujet !

Base de données

- Créez une base de données pour vos objets
  - En fonction du type d'objets et de ses capacités
- Un objet doit également avoir un nom, une position géographique, un niveau de zoom minimum à partir duquel il est visible, une icône (image, taille, position de l'ancre)
- Un objet bloqué peut également avoir un indice pour aider le joueur

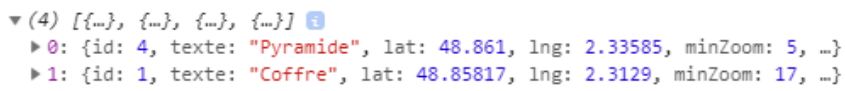
Partie serveur

- Créez votre service web. Par exemple :
  - `objets.php` renvoie tous les objets qui permettent de commencer le jeu
  - `objets.php?id=2` renvoie l'objet dont l'identifiant est 2
  - Mieux, si vous utilisez le framework FlightPHP (tutoriel sur le site du cours), vous pouvez créer votre propre API REST[2] et gérer des URLs de type
    - `GET api/objets`
    - `GET api/objets/2`
- Votre service web/API doit renvoyer les résultats en JSON
  - Vérifier l'encodage des nombres

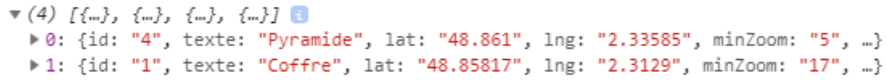
[2] Une API REST simpliste, pas une architecture complète. Pour en savoir plus : https://fr.wikipedia.org/wiki/Representational_state_transfer

- Si vous avez plusieurs tables, faites les bonnes jointures et faites attention aux noms de champs identiques

Bon retour JSON



Moins bon retour JSON



Partie client

Tout d'abord, intégrez une simple carte (Leaflet, Google Maps, etc.), éventuellement avec le provider de tuiles que vous souhaitez : OpenStreetMap, Mapbox (avec clé), etc.

- Créez un bouton « Jouer » pour lancer le jeu
- Lors du début du jeu, appelez votre API en AJAX pour charger les objets utiles au départ du jeu (retour JSON). Pour chaque objet, créez un marqueur sur la carte (à la bonne position, avec son icone, et seulement visible en fonction du zoom)
    - Si vous avez choisi Leaflet, il est possible de créer un `L.featureGroup` pour chaque marqueur. À chaque fois que l'on interagit avec la carte (évènement `zoomend` par exemple), on parcourt tous les `featureGroup`, et pour chaque groupe, on le vide de ces calques et on ajoute conditionnellement le marqueur en fonction du zoom actuel
- Pour chaque objet présent, il nous faut gérer l'évènement `click` et exécuter les bonnes actions en fonction du type de l'objet :
    - L'objet est récupérable :
        - On le déplace dans les objets récupérés
        - L'objet n'est plus visible sur la carte
    - L'objet est un code :
        - On affiche le code
    - L'objet est bloqué par un autre objet :
        - On a déjà l'objet qui débloque
            - On vérifie si l'objet qui débloque est bien sélectionné dans mes objets récupérés
            - L'objet n'est plus bloqué, et est ouvert
        - On n'a pas encore l'objet qui débloque
            - On appelle l'API en AJAX pour connaitre l'objet bloquant et l'indice
    - L'objet est bloqué par un code
        - On appelle l'API en AJAX pour connaitre l'objet code et affichez l'indice
        - On crée un formulaire pour taper le code
        - On vérifie les données envoyées lors de la validation
        - On valide ou non
    - L'objet libère un nouvel objet :
        - On appelle l'API en AJAX pour récupérer les données de ce nouvel objet
        - On crée le nouvel objet (marqueur) comme précédemment

        - L’objet n’a pas (plus) de rôle
            - Objet déjà ouvert par exemple
- Détection de la fin du jeu

Scores

- À la fin du jeu, sauvegardez le score du joueur dans la base de données. Ça peut être le temps qu’il a mis pour résoudre l’énigme ou tout autre système de notation. Vous aurez besoin de son nom/pseudo
- Ajoutez sur votre page d’accueil le « Hall of fame » (les meilleurs scores) généré dès le chargement de la page

Bon travail !