
1 Modélisation des données (cœur du projet)

Classe **Date**

Rôle :

- Représenter une date d'événement
- Vérifier si une date est passée ou non

Fonctionnalités clés :

- Stockage jour / mois / année
- Affichage formaté (`__str__`)
- Méthode `est_passee()` utilisant `datetime`

Axe : **gestion temporelle des événements**

Classe **Evenement**

Rôle :

- Représenter un événement avec ses informations principales

Attributs principaux :

- nom
- lieu
- date (objet `Date`)
- contact
- gratuit / payant (`bool`)

Fonctionnalités :

- Méthode `__str__` pour affichage
- Méthodes d'accès / affichage (nom, date, lieu, etc.)

Axe : **structure objet d'un événement**

2 Gestion des événements

Création d'événements

Fonctions :

- `CreerEvenement`

- CreerID

Principe :

- Les événements sont stockés dans un dictionnaire {id: Evenement}
- L'ID est généré automatiquement (récursif ou via la taille du dictionnaire)

Axe : **ajout dynamique d'événements**

Modification d'événements

Fonction :

- modifier_evenement(...)

Fonctionnalités :

- Modification ciblée d'un événement existant
- Mise à jour partielle (nom, date, lieu, contact, prix)

Axe : **édition d'événements existants**

Recherche d'événements

Classe :

- Recherche

Fonctionnalités :

- Recherche par :
 - nom
 - date
 - lieu
 - contact
- Filtrage spécifique des événements gratuits

Axe : **recherche et filtrage intelligent**

3 Persistance des données (sauvegarde / chargement)

Sauvegarde dans un fichier : base-evenement.txt

Fonctions :

- Sauver
- SauverEvent

Principe :

- Les événements sont stockés dans un fichier texte
- Format CSV simplifié séparateur virgule
- Évite les doublons d'ID

Axe : **stockage permanent des données**

Chargement des événements**Fonction :**

- Charger

Fonctionnalités :

- Lecture du fichier
- Reconstruction des objets Date et Evenement
- Remplissage du dictionnaire principal

Axe : **reconstruction des données au lancement**

4 Interface graphique et terminal (Tkinter)

Fenêtres principales**Fonctionnalités GUI :**

- Fenêtre principale (menu)
- Ajout d'un événement
- Modification d'un événement
- Choix gratuit / payant (radio buttons)
- Champs de saisie (nom, date, lieu, contact)

Une version TUI existe et est plus stable : TuiPils()

Axe : **interaction utilisateur via interface graphique**

Séparation logique

- Interface \neq logique métier
- Les fenêtres récupèrent les données
- Les traitements sont faits dans les fonctions/classes
- L'interface est connecté au programme avec des lambdas

5 Objectif global du projet

Créer une application de gestion d'événements permettant :

- D'avoir des événements de base préenregistrés
- D'ajouter de nouveaux événements
- De modifier les événements existants
- De sauvegarder les données
- De rechercher et filtrer les événements
- D'interagir via une interface graphique