# Hackathon Mastères : Sustain-A-Game

## Objectif

Développer un jeu vidéo mobile *low tech* en Java pour Android en 31 heures continues. Les trois équipes, composées de développeurs, designers et marketeurs, doit collaborer simultanément pour concevoir un jeu engageant, performant et respectueux des principes d’écoconception.

### Attendus

* Jeu jouable sur Android avec une boucle de *gameplay* simple et efficace.
* Une histoire cohérente divisée en 3 à 5 niveaux, définie par les marketeurs.
* Personnages, décors et éléments visuels créés par les designers, en intégrant des principes d’écoconception.
* Projet Java complet, bien structuré et commenté.
* Présentation finale du projet, de son déroulé, documenté.

## Rôles et Responsabilités

#### Marketeurs :

**Gestion de Projet** :

* + - Planifier les différentes phases du projet en collaboration avec les développeurs et les designers.
    - S’assurer que chaque membre de l’équipe comprend ses responsabilités et les délais à respecter.
    - Gérer les ressources et veiller à la bonne progression du projet.
  + **Management** :
    - Faciliter la communication entre les différents pôles pour assurer une bonne collaboration.
    - Animer des points de suivi réguliers pour ajuster les objectifs et résoudre les problèmes éventuels.
    - Motiver l’équipe et veiller à maintenir un environnement de travail positif et productif.
  + Définir un arc narratif évolutif dès le début du projet.
  + Collaborer avec les designers pour l'intégration narrative et visuelle.
  + Soutenir les développeurs en veillant à ce que l'histoire soit reflétée dans les mécaniques de jeu.

#### Designers :

* **Écoconception** :
  + Utiliser des assets graphiques légers pour minimiser la consommation de ressources.
  + Réduire les animations complexes qui consomment beaucoup d’énergie.
  + Travailler avec des éco-couleurs
  + Utiliser un style graphique « éco-responsable » pour limiter l’impact sur les performances
  + Limiter les plans de travail et les calques dans Illustrator
  + Utiliser des formats d’exports adaptés
* Créer des esquisses rapides pour aligner les concepts avec le narratif.
* Travailler sur les assets en parallèle du développement pour une intégration continue.
* Ajuster les éléments visuels en fonction des retours des développeurs et des marketeurs.

#### Développeurs :

* Mettre en place une architecture de base dès le début pour intégrer rapidement les éléments narratifs et visuels.
* Tester et intégrer en continu les assets fournis par les designers.
* Collaborer avec les marketeurs pour s'assurer que le gameplay soutient bien l'histoire.

##### # Conception et Architecture

* Minimiser les fonctionnalités non essentielles : Chaque fonctionnalité, chaque écran, chaque ligne de code a un coût énergétique. Concentrez-vous sur l'essentiel et évitez le "feature creep". Une application plus légère est plus rapide et moins énergivore.
* Optimiser les interfaces utilisateur (UI) et l'expérience utilisateur (UX) :
* Design épuré : Privilégiez les interfaces simples, avec moins d'animations complexes et de couleurs vives (notamment sur les écrans OLED où les pixels noirs ne consomment pas d'énergie).
* Limiter les requêtes réseau : Concevez l'application pour réduire le nombre et la taille des requêtes vers les serveurs. Le mode hors ligne, la mise en cache (caching) des données et la synchronisation intelligente sont vos alliés.
* Fluidité : Une UX bien pensée réduit le temps passé par l'utilisateur sur l'application, et donc sa consommation d'énergie.
* Langages et frameworks efficients : Certains langages et frameworks sont intrinsèquement plus gourmands en ressources que d'autres. Faites des choix éclairés en privilégiant la performance et l'optimisation. Attention, si un framework simplifie l'écriture du code, il peut rapidement augmenter le nombre d'instructions à exécuter par la machine.
* Éviter la sur-ingénierie : N'ajoutez pas de dépendances ou de bibliothèques inutiles qui alourdissent l'application.

##### # Développement et Optimisation du Code

* Optimiser les performances : Évitez les fuites de mémoire (memory leaks) et libérez les ressources inutilisées dès que possible.
* Optimisation des algorithmes : Privilégiez les algorithmes efficaces (complexité algorithmique faible) pour les traitements gourmands en ressources.
* Réduire les calculs inutiles : Évitez les boucles infinies ou les calculs répétés pour des valeurs constantes.
* Gestion de la batterie : N'activez la géolocalisation que lorsque c'est strictement nécessaire et désactivez-la dès que possible. Utilisez les modes de localisation moins précis quand cela suffit. Limitez l'utilisation des capteurs (accéléromètre, gyroscope, etc.) au strict minimum.
* Optimisation des données : Compressez les images, les vidéos et les données textuelles avant de les envoyer ou de les recevoir.
* Requêtes ciblées : Ne téléchargez que les données nécessaires. Utilisez des API RESTful pour récupérer uniquement les informations requises.
* Mise en cache intelligente : Mettez en cache les données fréquemment utilisées pour éviter de les re-télécharger.
* Gestion des erreurs : Une bonne gestion des erreurs évite les plantages qui entraînent une consommation excessive de ressources et une mauvaise expérience utilisateur.

# Tests et Déploiement

* Intégrez des tests de performance et de consommation énergétique dans votre processus de QA. Identifiez les goulots d'étranglement et les fonctionnalités les plus gourmandes.
* Serveurs : Si votre application communique avec un backend, l'éco-conception s'étend aux serveurs : choisissez des hébergeurs écologiques, optimisez les requêtes et les traitements côté serveur.

## Contraintes Techniques

* Développement en Java.
* Fonctionnement fluide sur des appareils Android à ressources limitées.
* Taille finale du jeu ne dépassant pas 50 Mo.
* Intégration continue pour permettre des tests réguliers.
* Dimension de travail finale 360 x 800

## Contraintes Design

* Créer une palette de 16 éco couleurs web (voir présentation)
* Interdiction d’usage de l’IA
* Format d’export : Bitmap ou gif non animés
* Poids du fichier Ai à minimiser : Le moins de calques possibles (le moins lourd dispose de points bonus)
* Création en style graphique Pixel Art
* Dimension de travail finale 360 x 800

## Livrables

* Fichier APK du jeu.
* Documentation du code et des choix de conception.
* Présentation finale du concept, des défis et des solutions.

## Évaluation

* Originalité de l’histoire et du gameplay.
* Qualité, cohérence, et optimisation écologique des éléments graphiques.
* Fluidité et optimisation du jeu sur Android.
* Efficacité de la gestion de projet et de la communication au sein de l’équipe.
* + 1 points par tranche de 2Mo en moins.