CHECKLIST IMPRIMABLE

La checklist Unity mobile : du prototype au build prêt à tester

Utilisez cette feuille de contrôle pour cadrer les performances de votre jeu Unity sur mobile avant que les ralentissements ne deviennent une dette technique. Cochez chaque point sur des builds testés sur de vrais appareils.

Au sommaire

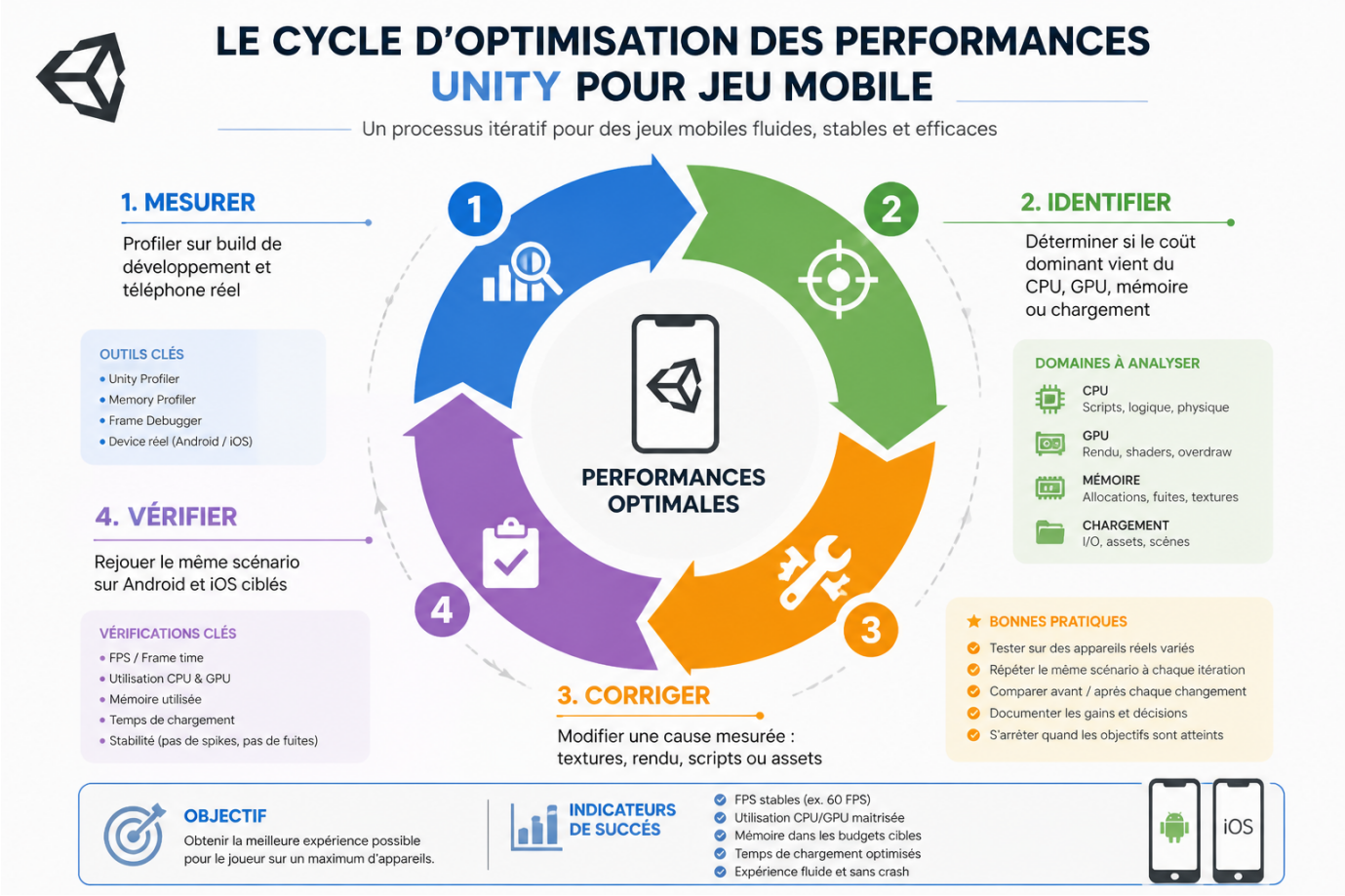
1. Cadrer le budget mobile dès le prototype

Définir les appareils de référence Android et iOS
Prévoyez notamment un Android proche du minimum visé, un modèle plus récent et un appareil Apple compatible avec votre cible.

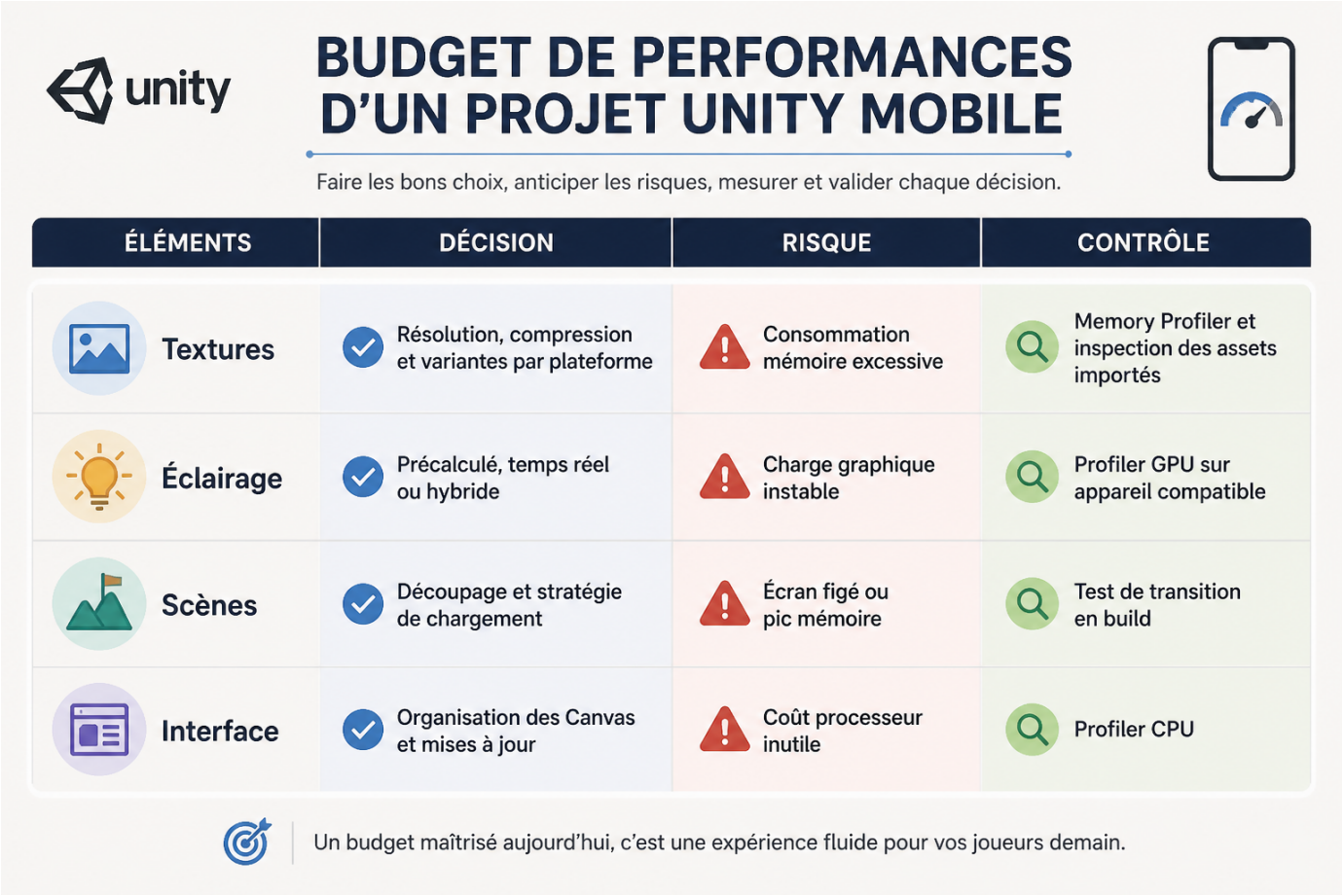
Fixer des objectifs mesurables de fluidité, mémoire, taille de build et chargement
Adaptez-les au genre du jeu, à la direction artistique et au parc d'appareils ciblé.

Choisir un niveau graphique minimal réaliste
La lisibilité, la réactivité et la stabilité priment souvent sur un effet visuel ponctuel.

Créer un scénario de test répétable
Utilisez un niveau ou une séquence représentative pour comparer les builds au fil du projet.



INFOGRAPHIE *Schéma des étapes pour optimiser les performances Unity Android et iOS*



INFOGRAPHIE *Les postes à budgéter et à surveiller tout au long du développement mobile.*

2. Contrôler les sources de ralentissement

Mesurer avant d'optimiser
Identifiez si le coût dominant vient du rendu, du processeur, de la mémoire ou des chargements.

Examiner les textures importées
Vérifiez leur résolution, leur compression et leurs variantes selon la plateforme.

Réduire les effets visuels coûteux
Surveillez particulièrement transparences, particules, ombres en temps réel et matériaux complexes.

Limiter le travail de rendu répété
Contrôlez les appels de rendu et les objets visibles dans les scènes représentatives.

Traquer les allocations et les mises à jour inutiles
Inspectez les scripts, l'interface et la fréquence de mise à jour des Canvas.

Découper les niveaux et tester les transitions
Validez les chargements pour éviter écran figé et pic de mémoire.

3. Associer chaque symptôme au bon contrôle

Symptôme observé	Piste à vérifier	Contrôle utile
Baisse de fluidité pendant les scènes chargées	Rendu, effets visuels, ombres, transparences	Profiler GPU sur appareil compatible

Symptôme observé	Piste à vérifier	Contrôle utile
Pic de mémoire ou fermeture du jeu	Textures, assets chargés, transitions de scènes	Memory Profiler et inspection des assets
Ralentissement lié à la logique ou à l'interface	Scripts, allocations, Canvas mis à jour trop souvent	Profiler CPU et débogage de l'interface
Transition longue ou écran figé	Découpage des scènes et stratégie de chargement	Test en build de développement sur appareil réel

4. Préparer un export Android et iOS maîtrisé

Configurer les réglages de build pour chaque plateforme
Un projet commun ne dispense pas de réglages et de validations spécifiques à Android et à iOS.

Vérifier la compatibilité avec la version Unity retenue
Les versions de systèmes et appareils pris en charge évoluent : consultez la documentation correspondante.

Contrôler le contenu final du build
Repérez les packages, assets et dépendances qui augmentent inutilement la taille ou la complexité.

Tester le build exporté, pas seulement l'éditeur
L'ordinateur de développement ne reproduit ni la puce graphique, ni la mémoire, ni la batterie d'un téléphone.

Répéter les parcours clés avant l'envoi en boutique
Testez lancement, gameplay représentatif, transitions, retours en arrière-plan et comportements propres à la plateforme.

> ! **À ne pas confondre**
> La portabilité d'Unity facilite le partage du projet entre Android et iOS, mais ne garantit ni la fluidité ni la compatibilité. Chaque plateforme et chaque appareil de référence doivent être validés avec le build final.

Le réflexe à conserver

- Mesurer sur appareil réel avant de modifier l'architecture ou de remplacer un système entier.
- Préserver une direction artistique compatible avec les contraintes de mémoire, de rendu et de batterie.
- Corriger d'abord la cause dominante observée dans les outils d'analyse.
- Considérer textures, effets, appels de rendu, allocations et chargements comme des postes à surveiller en continu.

Les réglages et compatibilités exacts dépendent de la version d'Unity, du pipeline de rendu et des appareils ciblés. Vérifiez la documentation Unity et les exigences des plateformes avant publication.