

A man with a beard is seen from the side, looking at a computer monitor. The monitor displays a dashboard with a table of data. The table has columns for 'Date', 'Statut', 'Type de tâche', 'Statut', 'Statut', 'Statut', 'Statut', and 'Statut'. The 'Statut' column contains green and red indicators. The man is resting his chin on his hand, looking thoughtful.

CHECKLIST IMPRIMABLE

Le lancement maîtrisé par feature flag

Utilisez cette checklist pour préparer, activer et contrôler une fonctionnalité déjà livrée en production sans redéployer. Elle aide à limiter l'exposition, tout en gardant un retour rapide vers le comportement précédent.

Au sommaire

- 1 Le principe à garder en tête
- 2 Avant le déploiement du code
- 3 Activer progressivement en production
- 4 Contrôles pendant l'activation
- 5 Après le lancement : retirer la dette du flag

Le principe à garder en tête

Le code est déployé en production, mais une configuration décide si la fonctionnalité est exécutée.

La livraison technique et l'activation fonctionnelle sont deux décisions distinctes.

Selon l'architecture, la configuration peut être rechargée à chaud ; l'activation sans redémarrage n'est donc pas automatique.

Un feature flag ne corrige pas un défaut de code : il réduit surtout l'exposition et permet une désactivation rapide.

LE FEATURE FLAG

Activer une fonctionnalité sans redéployer

Un interrupteur logiciel qui permet d'activer ou de désactiver une fonctionnalité à distance et en temps réel.



DES CONTEXTES DE CIBLAGE PUISSANTS

Les feature flags peuvent être activés selon différents contextes pour atteindre précisément les bons utilisateurs.



Combinez plusieurs contextes pour un ciblage encore plus précis et adapté à vos objectifs.

Les feature flags ne remplacent pas de bonnes pratiques de développement et de tests, ils les complètent.

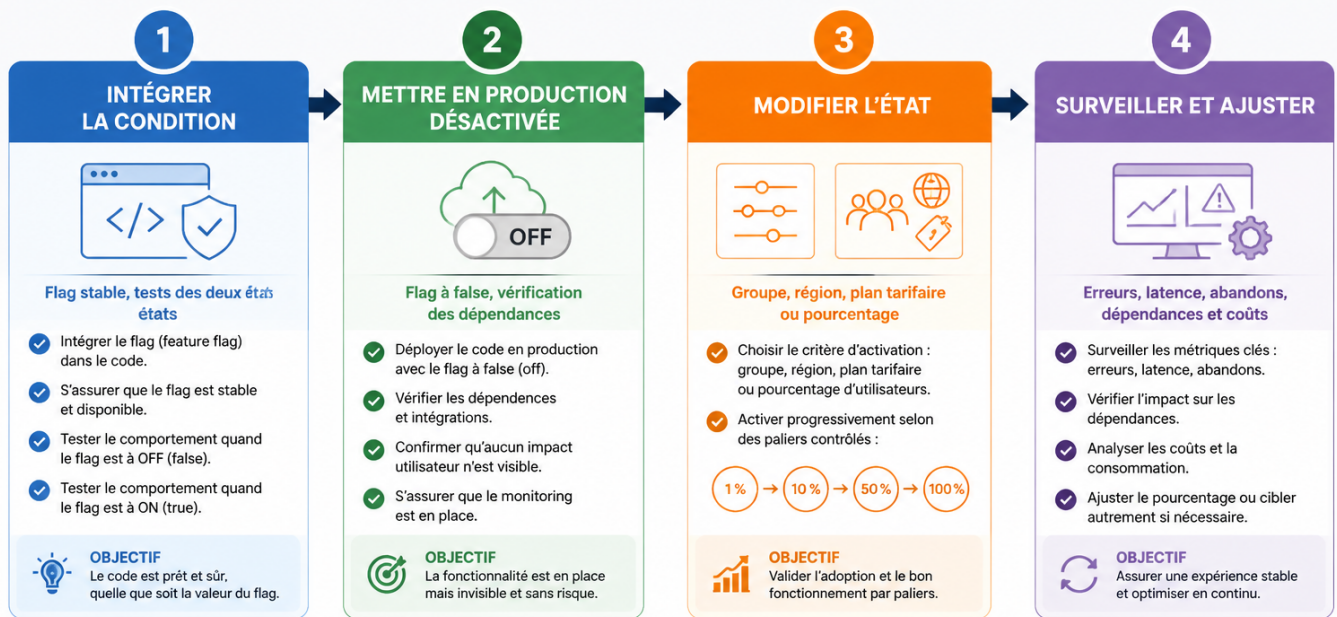
POURQUOI UTILISER LES FEATURE FLAGS ?

- Déploiement plus sûr**
Réduisez les risques en testant avant d'exposer à tous.
- Plus d'agilité**
Expérimentez, apprenez et itérez rapidement.
- Meilleure expérience utilisateur**
Livrez les bonnes fonctionnalités aux bons utilisateurs.
- Contrôle total**
Activez ou désactivez à tout moment, sans redéploiement.

INFOGRAPHIE Schéma du fonctionnement d'un feature flag pour activer une fonctionnalité sans redéploiement

ACTIVATION D'UNE FONCTIONNALITÉ SANS REDÉPLOYER

— Déployez en toute sécurité. Activez progressivement. Apprenez en continu. —



POURQUOI UTILISER LES FEATURE FLAGS ?



Réduire les risques en production



Livrer plus vite et plus souvent



Cibler et tester sans impacter tous les utilisateurs



Apprendre et décider sur la base de données réelles



Optimiser les coûts et les ressources



À RETENIR :

Les feature flags permettent de dissocier le déploiement du code de son activation. Vous gardez le contrôle, réduisez les risques et améliorez en continu.



SCHÉMA Les étapes d'activation progressive d'une fonctionnalité contrôlée par feature flag.

Avant le déploiement du code

Définir un état désactivé réellement fonctionnel

Le parcours précédent doit rester utilisable, et pas seulement être masqué dans l'interface.

Recenser tous les points d'entrée

Vérifier l'interface, les API, les événements, les tâches en arrière-plan et les droits d'accès.

Choisir une règle d'activation compréhensible

Préciser si le ciblage concerne un groupe, un pourcentage d'utilisateurs, une région, un environnement ou un plan tarifaire.

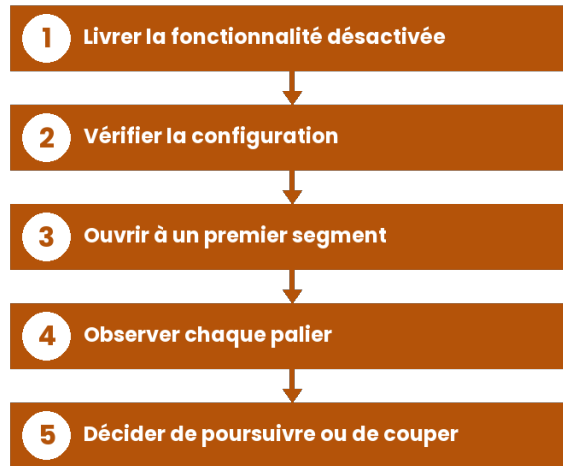
Préparer le scénario de retour arrière

Savoir quelle action désactive le flag et restaure le comportement précédent.

Tester en préproduction

Conserver les vérifications isolées avant la mise en ligne, notamment pour les migrations, permissions et services externes.

Activer progressivement en production



1 Livrer la fonctionnalité désactivée

Déployer le code avec le flag positionné sur l'état qui conserve le comportement précédent.

2 Vérifier la configuration

Confirmer que le mécanisme de configuration est séparé des données critiques et que sa modification est maîtrisée.

3 Ouvrir à un premier segment

Commencer par une exposition limitée, par exemple 1 %, puis augmenter progressivement à 10 %, 50 % et 100 % selon les résultats observés.

4 Observer chaque palier

Contrôler les résultats et les éventuelles régressions avant de poursuivre l'extension.

5 Décider de poursuivre ou de couper

Si un défaut apparaît, désactiver rapidement la fonctionnalité et revenir au comportement précédent.

Contrôles pendant l'activation

☐ Vérifier que la règle cible bien les utilisateurs prévus

Contrôler le segment, le contexte et l'environnement concernés.

☐ Surveiller le comportement de la fonctionnalité

Comparer les résultats observés à ceux attendus avant chaque nouveau palier.

☐ Confirmer que l'ancien parcours reste disponible

Le retour arrière doit pouvoir être effectué sans modifier le code.

☐ Vérifier les dépendances

Porter une attention particulière aux appels vers des services externes, aux permissions et aux traitements asynchrones.

☐ Tracer la décision d'activation

Documenter qui active, sur quel périmètre et dans quel objectif.

! Ne confondez pas interrupteur et correctif

Désactiver un feature flag limite l'exposition d'une fonctionnalité, mais ne répare ni le code, ni les données, ni une dépendance défectueuse. Les tests, la surveillance et la préparation du retour arrière restent indispensables.

Après le lancement : retirer la dette du flag

Une fois la fonctionnalité stabilisée et son activation généralisée, planifiez le nettoyage du dispositif : identifier le code devenu inutile, retirer l'ancien parcours lorsqu'il n'est plus nécessaire et supprimer le flag selon votre processus de maintenance. Le découplage entre déploiement et activation ne dispense pas de retirer les éléments temporaires.

À adapter à votre architecture et à vos procédures de mise en production. L'activation sans redéploiement dépend notamment de la capacité de l'application à recharger sa configuration.