Cours 10

Introduction aux Subagents

Anthropic Academy — Resume detaille

Yoann Boulch | 01/04/2026

1. Qu'est-ce que les subagents

■ Assistants isolés avec leurs propres fenêtres de contexte

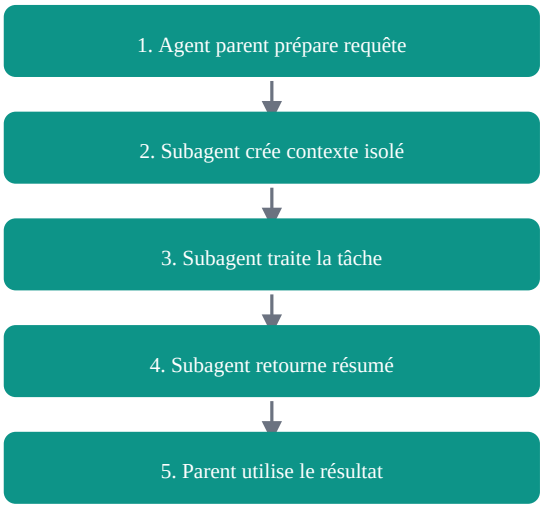
Un subagent est une instance isolée d'agent Claude avec sa propre fenêtre de contexte, créée pour accomplir une tâche spécifique avant de retourner les résultats.

Points clés:

- Contexte séparé et isolé
- Traitement parallèle possible
- Retour structuré au parent
- Pas de pollution du contexte principal

2. Fonctionnement des subagents

Flux d'un subagent



Les subagents créent une nouvelle session avec leur propre contexte, effectuent le travail, puis retournent un résumé au parent.

3. Créer des subagents

■ Utilisez /agents pour créer une configuration de subagent

```
# Configuration subagent (dans CLAUDE.md ou config) subagents: code_reviewer: name: "Code Reviewer" description: "Revue complète de code" instructions: | Tu es un expert en revue de code. Analyse qualité, sécurité, performance. doc_generator: name: "Documentation Generator" description: "Génère documentation technique" instructions: | Tu es expert en documentation. Crée des docs claires et complètes.
```

4. Types de subagents spécialisés

■ Créez des subagents pour différents rôles

Type	Spécialité	Exemple
Code Reviewer	Revue qualité	Tests, sécurité
Doc Generator	Documentation	Specs, guides
Test Writer	Tests unitaires	Coverage, cas
Validator	Validation	Format, règles

5. Conception efficace des subagents

■ Définissez sorties structurées et rapportage d'obstacles

Un bon subagent produit des résultats structurés et rapporte clairement les obstacles rencontrés.

Bonnes pratiques:

- Définir un format de sortie clair (JSON ou Markdown)
- Demander un résumé exécutif en premier
- Inclure les obstacles/limitations trouvées
- Fournir les preuves ou références

6. Limiter l'accès aux outils

■ Principe du moindre privilège pour la sécurité

Les subagents devraient avoir accès uniquement aux outils nécessaires pour leur tâche.

Restriction des outils:

- Code reviewer: read, grep seulement (pas write)
- Doc generator: read, write pour docs/ seulement
- Validator: read seulement, pas de modifications

7. Quand utiliser les subagents

■ Idéal pour tâches complexes multi-fichiers ou parallèles

Situations appropriées:

- Analyser plusieurs fichiers en parallèle
- Déléguer une tâche spécialisée
- Garder le contexte parent propre
- Combiner résultats de plusieurs analyses

8. Quand NE PAS utiliser les subagents

■ Attention au surcoût et à la complexité

À éviter:

- Tâches simples et rapides (trop de surcoût)
- Requête une seule ligne (utiliser le contexte principal)
- Dépendances fortes entre étapes
- Quand la latence est critique

9. Patterns de subagents

■ Modèles architecturaux réutilisables

```
# Pattern Fan-Out / Fan-In Agent Parent: 1. Crée subagents pour chaque fichier 2. Attend tous les résultats 3. Aggrège et synthétise # Pattern Pipeline Agent1 → Agent2 → Agent3 Chaque agent traite sa partie # Pattern Specialist Delegation Parent → Specialist Subagent Parent gère l'orchestration Spécialiste fait l'expertise
```

10. Subagents + Compétences

■ Combinez pour des workflows experts

Les subagents spécialisés bénéficient de compétences dédiées qui leur fournissent des instructions et contexte expert.

Architecture recommandée:

1. Parent agent: orchestration générale
2. Subagent specialist: tâche spécifique
3. Skill attached: expertise détaillée
4. Structured output: résultats standardisés

11. Gestion du contexte en subagents

■ Contexte isolé = pas de pollution du parent

Chaque subagent commence avec un contexte vierge, sans l'historique du parent.

Avantages:

- Parent ne voit que le résumé du subagent
- Pas d'accumulation d'informations inutiles
- Meilleure performance avec contexte court
- Résultats reproductibles et prévisibles

12. Cas d'usage pratiques

■ Exemples concrets d'utilisation

Exemple 1: Revue de code

Parent crée un subagent Code Reviewer pour chaque fichier modifié. Les résultats sont agrégés en rapport complet.

Exemple 2: Génération de docs

Parent délègue la documentation à un subagent spécialisé avec accès en lecture. Subagent génère docs Markdown structurées.

Exemple 3: Audit de sécurité

Parent crée subagent Validator pour chaque service. Chacun analyse la sécurité en isolation, rapportant les risques.

13. Performance et optimisation

■ Optimisez l'utilisation des subagents

Les subagents ont un coût en latence et en contexte. Optimisez leur utilisation.

Stratégies d'optimisation:

- Paralleliser: créer plusieurs subagents simultanément
- Limiter le contexte: ne passer que ce qui est nécessaire
- Cacher les résultats: réutiliser pour requêtes similaires
- Monitorer: suivre les temps d'exécution

14. Troubleshooting et debugging

■ Déboguer les problèmes avec les subagents

Problèmes courants:

- Subagent ne complète pas: contexte insuffisant ou timeout
- Sortie non structurée: clarifier le format attendu
- Coût élevé: évaluer si subagent vraiment nécessaire
- Inconsistance: fournir des instructions plus détaillées

15. Résumé et prochaines étapes

■ Maîtrisez la délégation avec les subagents

Les subagents transforment comment on orchestre le travail complexe. Utilisez-les judicieusement pour des tâches réellement complexes, et vous débloquerez une nouvelle dimension de productivité.