

Plan de Formation

Claude Code 2.1.45 + Claude-Craft 7.13.0 — Programme
détaillé

The Bearded Bear

Février 2026



Table des matières

Plan de Formation : Claude Code 2.1.45 + Claude-Craft 7.13.0	2
Contexte de Formation	2
Versions couvertes	2
Format Recommande : 2 jours (14h)	2
Programme Detaille	2
JOUR 1 : Fondamentaux & Nouveaux Projets (7h)	2
JOUR 2 : Projets Existants & Pratique Avancee (7h)	7
Options Additionnelles	11
Option A : Demi-journee de suivi (3h) - 2 semaines apres	11
Option B : Coaching continu (forfait mensuel)	12
Prerequis Participants	12
Techniques	12
Comptes	12
Livrables Inclus	12
Evaluation	13
Quiz de validation des acquis (15min)	13
Auto-evaluation des competences	13
Criteres de reussite	13
Adaptation	13

Plan de Formation : Claude Code 2.1.45 + Claude-Craft 7.13.0

Contexte de Formation

- **Public** : Equipe de developpeurs
- **Objectif** : Maitrise de Claude Code 2.1.45 avec le framework Claude-Craft 7.13.0
- **Cas d'usage** : Nouveaux projets ET projets existants
- **Stack technologique** : 10 stacks supportees (Symfony, React, Flutter, etc.)
- **Format** : Hybride (presentiel + distanciel)

Versions couvertes

Composant	Version	Nouveautes cles
Claude Code	2.1.45	Extended Thinking, MCP, Sub-agents, Permissions 3-tier, Sonnet 4.5/Opus 4.6, Agent Teams, Fast Mode
Claude-Craft	7.13.0	BMAD v6, Ralph, QA Recette, 33 agents, 160 commandes across 20 namespaces
Formation	3.0.0	Mise a jour complete

Format Recommande : 2 jours (14h)

Journee	Focus	Duree
Jour 1	Fondamentaux + Nouveaux projets	7h
Jour 2	Projets existants + Pratique avancee	7h

Programme Detaille

JOUR 1 : Fondamentaux & Nouveaux Projets (7h)

Module 1 : Introduction a Claude Code 2.1.45 (1h30) Objectifs pedagogiques : - Comprendre ce qu'est Claude Code 2.1.45 et ses nouvelles capacites - Savoir installer et configurer Claude Code - Maitriser les commandes de base et avancees (Extended Thinking, MCP, Permissions)

Contenu :

1. **Presentation de Claude Code 2.1.45** (20min)
 - Qu'est-ce que Claude Code? (CLI officiel Anthropic)
 - Difference avec ChatGPT, GitHub Copilot
 - Nouveautes 2.1.45 : Extended Thinking, MCP, Sub-agents, Permissions 3-tier, Fast Mode, Agent Teams
 - Modeles disponibles : Sonnet 4.5 (rapide), Opus 4.6 (flagship), Opus 4.5 (puissant), Haiku (leger)
2. **Installation et configuration** (20min)
 - Installation via npm: `npm install -g @anthropic-ai/claude-code`
 - Configuration de la cle API
 - Verification: `claude --version (2.1.45+)`
 - Configuration du modele par default
3. **Interface et commandes** (20min)
 - Commandes de base: `/help`, `/clear`, `/model`, `/exit`, `/cost`
 - Nouvelles commandes : `/compact`, `/doctor`, `/mcp`, `/permissions`, `/skills`, `/keybindings`, `/tasks`
 - Task Management System pour gestion multi-taches
 - Extended Thinking pour raisonnement approfondi
 - Permissions 3-tier : Allow, Deny, Ask
4. **Gestion du contexte et tokens** (20min)
 - Limites de contexte par modele (~200K)
 - TCL avec Claude-Craft (~95% economie)
 - MCP (Model Context Protocol) pour outils externes
 - Couts et facturation
5. **Exercice pratique** (10min)
 - Installation de Claude Code 2.1.45
 - Activation de l'Extended Thinking
 - Generation d'un script simple

Support: modules/jour1/01-introduction-claude-code.md **Exercice:** exercices/exercice-01-premier-projet.md

Module 2 : Le Framework Claude-Craft 7.13.0 (1h30) Objectifs pedagogiques : - Comprendre le TCL (Tiered Context Loading) et son economie de 95% - Savoir installer Claude-Craft 7.13.0 via npx - Connaitre les differents composants (skills, commands, agents, references, BMAD, Ralph, QA)

Contenu :

1. **TCL : Tiered Context Loading** (20min)
 - Pourquoi TCL ? (~95% economie de tokens)
 - Les 3 niveaux : ALWAYS, ON-DEMAND, REFERENCE
 - CLAUDE.md minimal (~700 bytes) vs v3.x (~15,000)
 - context.yaml pour triggers automatiques
2. **Architecture et 10 stacks supportees** (20min)
 - Structure TCL : CLAUDE.md, INDEX.md, context.yaml, references/, skills/
 - Technologies : symfony, laravel, react, angular, vuejs, flutter, reactnative, python, php, csharp
 - Composants majeurs : BMAD v6, Ralph Wiggum, QA Recette
 - 33 agents, 160 commandes across 20 namespaces
3. **Systeme de Skills** (15min)
 - Invocation : /testing, /security, /git-workflow
 - Chargement automatique via context.yaml
 - Skills vs References (on-demand vs complet)
4. **Commands et Agents** (15min)
 - Commands slash : /symfony:check-compliance, /csharp:generate-feature
 - Agents : @tdd-coach, @symfony-reviewer, @product-owner, @ralph-conductor
 - Difference Skills vs Commands vs Agents
5. **Installation via NPX** (10min)
 - Interactive : npx @the-bearded-bear/claude-craft install
 - Directe : npx @the-bearded-bear/claude-craft install ~/projet --tech=symfony --lang=fr
 - Verification de la structure TCL
6. **Configuration context.yaml** (10min)
 - Triggers automatiques par mots-cles
 - auto_load : true/false
 - Personnalisation pour le projet

Exercice pratique : - Installer Claude-Craft sur un projet de demo - Explorer la structure .claude/ - Lister les commandes et agents disponibles

Support: modules/jour1/02-framework-claude-craft.md **Exercice:** exercices/exercice-02-installation-claude-craft.md

Module 3 : Workflow de Developpement et BMAD v6 (2h) Objectifs pedagogiques : - Maitriser les 3 tracks BMAD v6 - Comprendre les Quality Gates et le Status Routing - Utiliser les commandes workflow et BMAD

Contenu :

1. **BMAD v6 : Les 3 tracks de developpement** (30min)

Track	Setup	Phases	Usage
Quick Flow	< 5 min	Implement only	Bugfix, hotfix
Standard	< 15 min	Plan -> Design -> Implement	Feature moyenne
Enterprise	< 30 min	Analyze -> Plan -> Design -> Implement	Plateforme, refactoring majeur

2. **Quality Gates** (20min)

Gate	Seuil	Quand
PRD Gate	>=80%	Vision -> PRD
Tech Spec Gate	>=90%	PRD -> Tech Spec
Backlog Gate	INVEST 6/6	Tech Spec -> Backlog
Sprint Ready	100%	Backlog -> Sprint
Story DoD	100%	Dev -> Done

3. **Status-based Routing** (15min)

- Cycle: backlog -> ready-for-dev -> in-progress -> review -> done
- Gestion du statut blocked
- TDD Phases : Red -> Green -> Refactor

4. **Commandes workflow et BMAD** (25min)

- `/workflow:init` - Analyse et recommandation du track
- `/workflow:analyze` - Phase d'exploration du code
- `/workflow:plan` - Generation PRD et backlog
- `/workflow:design` - Specifications techniques
- `/workflow:implement` - Developpement guide
- `/workflow:init`, `/workflow:status` - Initialisation et suivi workflow
- `/sprint:next-story`, `/sprint:transition` - Gestion des stories
- `/gate:validate-prd`, `/gate:validate-story` - Validation quality gates

5. **Atelier : Workflow Standard avec BMAD** (30min)

- Partir d'une user story
- Executer `/workflow:init`
- Suivre les etapes du workflow
- Passer les quality gates
- Implementer avec guidance Claude

Support: modules/jour1/03-workflow-developpement.md **Exercice:** exercices/exercice-03-workflow-standard.md

Module 4 : Demarrer un Nouveau Projet Symfony (2h) **Objectifs pedagogiques :** - Configurer Claude-Craft pour un projet Symfony 8.0 / PHP 8.5 - Generer des features completes - Respecter l'architecture Clean/Hexagonale

Contenu :

1. **Configuration initiale Symfony** (20min)
 - Creation projet : `symfony new mon-api --webapp`
 - Installation Claude-Craft : `make install-symfony TARGET=./mon-api LANG=fr`
 - Configuration du contexte : `00-project-context.md`
 - Personnalisation pour le projet
2. **Generation de features** (30min)
 - `/symfony:generate-feature` - Scaffold complet
 - `/symfony:generate-entity` - Entites Doctrine
 - `/symfony:generate-crud` - CRUD complet
 - Structure generee : Domain -> Application -> Infrastructure -> API
3. **Architecture Clean + Hexagonale** (20min)
 - Structure des repertoires
 - Couches et responsabilites
 - Inversion de dependances
 - Ports et Adapters
4. **API Platform et endpoints REST** (20min)
 - Configuration API Platform
 - Generation d'endpoints
 - Documentation OpenAPI automatique
 - Validation avec Symfony Validator
5. **Respect automatique des patterns** (10min)
 - SOLID (SRP, OCP, LSP, ISP, DIP)
 - DRY, KISS, YAGNI
 - PSR-12 et coding standards
 - Analyse automatique
6. **Atelier pratique : Micro-projet Symfony** (20min)
 - Creer un projet "Gestion de Taches"
 - Generer l'entite Task avec Doctrine
 - Creer le use case `CreateTaskHandler`
 - Ajouter l'endpoint API Platform
 - Ecrire les tests PHPUnit

Support: `modules/jour1/04-nouveau-projet-symfony.md` **Exercice:** `exercices/exercice-04-projet-symfony-scratch.md`

JOUR 2 : Projets Existants & Pratique Avancee (7h)

Module 5 : Integrer Claude-Craft sur un Projet Existant (1h30) **Objectifs pedagogiques :** - Auditer un projet Symfony existant - Elaborer une strategie d'adoption progressive - Gerer la dette technique avec Claude

Contenu :

1. **Audit initial du projet** (30min)
 - `/symfony:check-architecture` - Validation architecture hexagonale
 - `/symfony:check-code-quality` - PHPStan, PHP-CS-Fixer
 - `/symfony:check-security` - Audit securite OWASP
 - `/symfony:check-compliance` - Conformite globale
 - `/symfony:check-testing` - Couverture de tests
2. **Lecture et interpretation des rapports** (15min)
 - Niveaux de severite
 - Priorisation des issues
 - Quick wins vs refactoring profond
3. **Strategie d'adoption progressive** (15min)
 - Pas de "big bang"!
 - Commencer par les nouvelles features
 - Refactoring opportuniste
 - Strangler Fig Pattern
4. **Migration progressive vers Clean Architecture** (15min)
 - Identifier les bounded contexts
 - Extraire le domaine du legacy
 - Creer les interfaces (ports)
 - Implementer les adapters
5. **Atelier : Audit projet reel** (15min)
 - Executer un audit complet sur un projet de l'equipe
 - Analyser les findings
 - Elaborer un plan de remediation
 - Prioriser les actions

Support : modules/jour2/05-projet-existant.md **Exercice :** exercices/exercice-05-audit-projet-existant.md

Module 6 : Qualite et Securite (1h) **Objectifs pedagogiques :** - Maitriser les outils de qualite - Appliquer les principes de securite OWASP - Integrer la qualite dans le workflow Git

Contenu :

1. **Pre-commit checklist** (15min)
 - 13 sections de validation
 - Automatisation avec hooks Git
 - Integration CI/CD
2. **Tests TDD/BDD avec Claude** (15min)
 - Cycle Red-Green-Refactor
 - Generation de tests avec Claude
 - PHPUnit pour les tests unitaires
 - Behat pour les tests BDD
 - Couverture cible : 80%+
3. **Securite OWASP Top 10** (20min)
 - Injection (SQL, Command)
 - Broken Authentication
 - XSS (Cross-Site Scripting)
 - CSRF (Cross-Site Request Forgery)
 - Audit avec `/symfony:check-security`
4. **Git workflow et Conventional Commits** (10min)
 - GitHub Flow
 - Conventional Commits: `feat:`, `fix:`, `refactor:`
 - Pull Request et Code Review
 - Hooks de validation

Exercice : - Executer un audit qualite complet - Corriger les 3 findings les plus critiques - Ecrire les tests de non-regression

Support : `modules/jour2/06-qualite-securite.md` **Exercice :** `exercices/exercice-06-audit-qualite.md`

Module 7 : Agents Specialises, BMAD et Docker (1h30) **Objectifs pedagogiques :** - Connaitre les 33 agents repartis en 4 categories - Savoir invoquer le bon agent selon le contexte - Personnaliser les agents

Contenu :

1. **Panorama des 33 agents en 4 categories** (30min)

Common Agents (12) :

Agent	Specialite	Quand l'utiliser
<code>@api-designer</code>	Conception API REST/GraphQL	Design d'endpoints
<code>@database-architect</code>	Optimisation BDD	Schema BDD

Agent	Specialite	Quand l'utiliser
@devops-engineer	CI/CD, Docker	Infrastructure
@performance-auditor	Analyse performance	Optimisation
@refactoring-specialist	Refactoring	Restructuration
@tdd-coach	Coaching tests	Ecriture de tests
@uiux-orchestrator	Coordination UI/UX	Design global
@ui-designer	Design systems	Tokens, composants
@ux-ergonome	Flux utilisateur	Ergonomie cognitive
@accessibility-expert	WCAG 2.2 AAA	Accessibilite
@research-assistant	Recherche technique	Documentation externe
@ralph-conductor	Orchestration loop	Automatisation

Technology Reviewers (10) : @symfony-reviewer, @flutter-reviewer, @react-reviewer, @python-reviewer, @angular-reviewer, @laravel-reviewer, @vuejs-reviewer, @reactnative-reviewer, @csharp-reviewer, @php-reviewer

Docker Agents (5) : @docker-dockerfile, @docker-compose, @docker-debug, @docker-cicd, @docker-architect

Project Agents (2) : @product-owner, @tech-lead

Note : Les roles BMAD (pm, ba, architect, po, sm, dev, qa, ux) sont integres dans les commandes workflow et sprint, pas en tant qu'agents standalone.

2. Invocation et contexte (20min)

- Syntaxe : @agent-name [instruction]
- Passage de contexte
- Combinaison d'agents
- Output et formatage

3. Personnalisation (20min)

- Structure d'un agent : AGENT.md
- Prompts et personas
- Contraintes et guidelines
- Creation d'agents custom

4. Exercice : Code review assistee (20min)

- Selectionner du code a reviewer
- Invoquer @symfony-reviewer
- Analyser les suggestions
- Appliquer les corrections

Support : modules/jour2/07-agents-specialises.md **Exercice :** exercices/exercice-07-code-review-agent.md

Module 8 : Outils Avances, Ralph et Autonomie (1h30) Objectifs pedagogiques : - Maitriser les Hooks (13 evenements) - Utiliser Ralph Wiggum pour l'autonomie - Exploiter Extended Thinking, MCP, QA Recette - Configurer les Permissions et Plugins

Contenu :

1. **Système de Hooks (13 evenements)** (15min)
 - 13 evenements : PreToolUse, PostToolUse, PostToolUseFailure, SubagentStart, etc.
 - Configuration dans .claude/settings.json
 - Variables : \$FILE_PATH, \$TOOL_NAME, \$SESSION_COST
 - Exemples : lint auto, notification Slack, securite
2. **Ralph Wiggum** (20min)
 - Boucle continue jusqu'à completion
 - DoD Validators : command, output_contains, file_changed, hook, human
 - Circuit breaker et gestion d'erreurs
 - Invocation : /common:ralph-run "..."
3. **Extended Thinking et MCP** (15min)
 - Extended Thinking pour raisonnement complexe
 - MCP (Model Context Protocol) : integration d'outils externes
 - Configuration et activation
 - Cas d'usage avances
4. **QA Recette** (10min)
 - Tests d'acceptance automatisés via Chrome
 - Golden Rule : un bug corrige ne doit JAMAIS reapparaître
 - Generation automatique de tests de regression
 - Prerequis : Chrome extension v1.0.36+
5. **Permissions et Plugins** (10min)
 - Permissions 3-tier : Allow, Deny, Ask
 - Configuration fine des autorisations
 - Plugins et extensions disponibles
 - Integration IDE (VS Code, JetBrains)

Demo : - Ralph Wiggum en action sur une feature complete - Hooks de lint automatique - QA Recette sur un cas simple

Support : modules/jour2/08-outils-avances.md

Module 9 : Mise en Pratique Collective (1h30) Objectifs pedagogiques : - Appliquer les connaissances en situation réelle - Collaborer en equipe avec Claude - Etablir les bonnes pratiques d'equipe

Contenu :

1. **Challenge d'equipe** (30min)
 - Cas reel fourni par l'equipe ou prepare
 - Travail en binomes
 - Utilisation complete du workflow
 - Presentation des solutions
2. **Demo BMAD v6** (15min)
 - Initialisation d'un projet BMAD en live
 - Passage des quality gates
 - Status routing en action
3. **Demo QA Recette** (15min)
 - Lancement d'une session de recette sur un cas prepare
 - Observation du test automatise via Chrome
 - Generation du rapport
4. **Questions/Reponses** (15min)
 - Clarifications sur tous les modules
 - Cas specifiques de l'equipe
 - Problemes rencontres
5. **Bonnes pratiques de collaboration** (10min)
 - Partage du contexte en equipe
 - Conventions d'utilisation
 - Revue de code avec Claude
 - Documentation generee
6. **Pieges a eviter** (5min)
 - Over-reliance sur l'IA
 - Copier-coller aveugle
 - Ignorer les tests
 - Negliger la securite

Support : modules/jour2/09-atelier-pratique.md **Exercice :** exercices/exercice-08-challenge-equipe.md

Options Additionnelles

Option A : Demi-journee de suivi (3h) - 2 semaines apres

Contenu : - Retour d'experience de l'equipe (1h) - Resolution des problemes rencontres (1h) - Approfondissement sur demande (30min) - Optimisation des workflows etablis (30min)

Option B : Coaching continu (forfait mensuel)

Contenu : - 2h de support par semaine - Revue de code assistee - Accompagnement sur projets specifiques - Acces prioritaire au formateur

Prerequis Participants**Techniques**

- Connaissance de base du developpement logiciel
- Familiarite avec Git (commit, branch, merge)
- IDE installe (VS Code recommande)
- Node.js pour l'installation NPX
- Chrome (pour QA Recette)

Comptes

- Compte Claude/Anthropic actif avec credits
 - Acces aux depots de code de l'equipe (si audit)
-

Livrables Inclus

Livrable	Description
Slides de formation	Support de presentation (PDF/PPTX)
Cheat sheets	Aide-memoire Claude Code + Claude-Craft
Exercices et solutions	Tous les ateliers pratiques
Projet de demo	Projet Symfony configure avec Claude-Craft
Certificat	Attestation de participation
Acces Discord/Slack	Support post-formation (1 mois)

Evaluation

Quiz de validation des acquis (15min)

Questions types :

Theme	Exemple de question
Claude Code	Quelle commande active l'Extended Thinking?
Claude-Craft	Combien d'agents sont disponibles dans Claude-Craft 7.13.0?
BMAD v6	Quel est le seuil du PRD Quality Gate?
Ralph	Quels sont les 5 types de DoD Validators?
QA Recette	Qu'est-ce que la Golden Rule?
MCP	A quoi sert le Model Context Protocol?
Thinking	Quand utiliser l'Extended Thinking?
Task Management	Quels sont les 4 outils du Task Management System?
Permissions	Quels sont les 3 niveaux de permissions?

Auto-evaluation des competences

— Feedback formateur

Criteres de reussite

Critere	Objectif
Installation Claude-Craft	Autonome
Utilisation workflow	Choix du bon track BMAD
Generation de code	Feature complete
Audit de projet	Identification des issues
Agents	Invocation appropriée
Ralph	Comprehension de l'autonomie

Adaptation

Ce plan est adaptable selon :

- **Duree** : Condensable sur 1,5 jour ou extensible sur 3 jours
 - **Stack** : Symfony par default, adaptable (React, Python, Flutter)
 - **Niveau** : Junior, confirme, senior
 - **Focus** : Nouveaux projets uniquement, legacy uniquement, ou mixte
-

Version : 3.0.0 **Date** : Janvier 2026 **Auteur** : The Bearded CTO **Claude Code** : 2.1.45 **Claude-Craft** : 7.13.0