

Claude Code

Marketplace & Plugins

Créer et Distribuer vos Propres Extensions

Présentation Claude Code France - Octobre 2025
Para CC-France



Au Programme

1. **Introduction** - Pourquoi des plugins ?
2. **Architecture** - Comment ça marche ?
3. **Marketplace** - Système de distribution
4. **Création** - Développer votre plugin
5. **Publication** - Partager avec la communauté
6. **Exemple Réel** - WD Framework
7. **Best Practices** - Conseils et astuces
8. **Démonstration** - Live coding



Pourquoi des Plugins

?

Le Problème



Sans Plugins

- Configuration manuelle répétitive
- Commandes custom dispersées
- Pas de partage communautaire



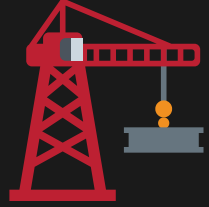
Cas d'Usage Plugins

Exemples Concrets



Développement

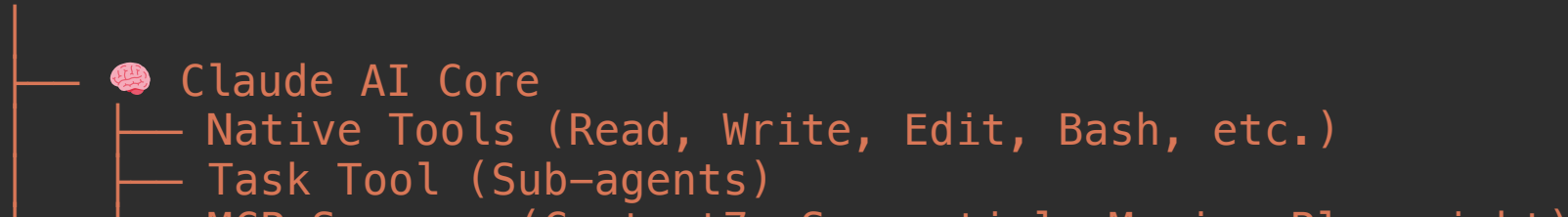
- **Frameworks spécifiques** - React, Vue, Angular workflows
- **Langages** - Python, Rust, Go best practices
- **Outils** - Docker, Kubernetes, CI/CD automation



Architecture Claude Code

Vue d'Ensemble du Système

Claude Code Architecture



Qu'est-ce qu'un Plugin ?

Définition

Un **plugin Claude Code** est un ensemble de fichiers structurés qui étend les capacités de Claude avec :

- **Slash commands** personnalisées



Composants



Structure d'un Plugin

Anatomie Complète

```
mon-plugin/  
├── .claude-plugin/  
│   ├── plugin.json  
│   └── marketplace.json  
└── commands/  
    ├── analyze.md  
    ├── build.md  
    └── deploy.md
```

Métadonnées plugin
Configuration principale
Info marketplace (optionnel)

Slash commands
/mon-plugin:analyze
/mon-plugin:build
/mon-plugin:deploy

plugin.json - Configuration

Fichier Principal du Plugin

```
{  
  "name": "mon-plugin",  
  "version": "1.0.0",  
  "description": "Description courte du plugin",  
  "author": {
```

marketplace.json - Distribution

Configuration Marketplace (Optionnel)

```
{  
  "name": "mon-marketplace",
```



Slash Commands


Créer une Commande Personnalisée

Fichier : `commands/analyze.md`

```
---  
description: Analyze codebase for issues and improvements  
arguments:  
  name: target
```

Slash Commands - Syntaxe

Structure d'une Commande

 Front Matter
(YAML)

```
---  
description: "Command description"  
arguments:  
  - name: target  
    description: "Argument desc"
```



Agents Spécialisés

Délégation via Task Tool

Fichier : `agents/frontend.md`

```
---  
description: Frontend development specialist  
subagent_type: frontend-specialist  
---
```

You are a frontend development specialist with expertise in:

- Modern UI frameworks (React, Vue, Angular)*



Types d'Agents Disponibles

Subagent Types Claude Code



Agents Disponibles

- **general-purpose**

Multi-domain, recherche complexe

frontend specialist



System Marketplace

Comment ça Fonctionne ?



Distribution via GitHub

1.  Développer Plugin
 - └─ Structure + commandes + documentation



Types de Distribution



Plugin Standalone

```
# Repository unique  
github.com/user/mon-plugin  
  
# Installation directe  
/plugin install \  
github.com/user/mon-plugin
```



Marketplace Multi-Plugins

```
# Repository marketplace  
github.com/org/marketplace  
  
# Ajout marketplace
```



Commandes Plugin Manager

Gestion des Plugins

```
# 📄 MARKETPLACE
```

```
/plugin marketplace list
```

```
/plugin marketplace add <url>
```

```
/plugin marketplace update <name>
```

```
/plugin marketplace remove <name>
```

```
# Lister marketplaces
```

```
# Ajouter marketplace
```

```
# Update marketplace
```

```
# Supprimer marketplace
```



Tutoriel : Créer Votre Plugin

Étape 1 : Initialiser Structure

```
# Créer structure de base
mkdir mon-plugin && cd mon-plugin
mkdir -p .claude-plugin commands agents .claude

# Créer plugin.json
```



Tutoriel : Créer Votre Plugin

Étape 2 : Créer Première Commande

```
# Créer commande /mon-plugin:hello  
cat > commands/hello.md <<'EOF'
```



Tutoriel : Créer Votre Plugin

Étape 3 : Documentation

```
# Créer README.md
cat > README.md <<'EOF'
# Mon Plugin

Description de votre plugin.

## Installation
```



Tutoriel : Créer Votre Plugin

Étape 4 : Git & GitHub

```
# Initialiser git  
git init  
git add .  
git commit -m "Initial commit: Mon Plugin v1.0.0"
```



Tutoriel : Créer Votre Plugin

Étape 5 : Tester l'Installation

```
# Dans Claude Code
```

```
# Installer directement depuis GitHub
```

```
/plugin install github.com/votre-compte/mon-plugin
```



Exemple Réel : WD Framework

Cas d'Usage Complet



Objectif

Framework intelligent pour développement web avec :



WD Framework : Structure

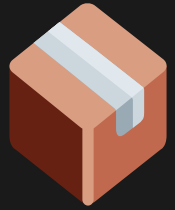
Architecture Complète

```
wd-framework/  
├── .claude-plugin/  
│   ├── plugin.json          # v2.0.1  
│   └── marketplace.json     # wd-marketplace
```



WD Framework : plugin.json

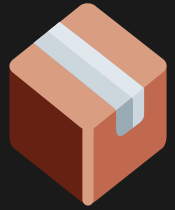
```
{
  "name": "wd",
  "version": "2.0.1",
  "description": "Intelligent web development framework with 22 specialized commands, 5 expert agents, 11 AI personas, and complete orchestration system",
  "author": {
    "name": "Para CC-France",
    "email": "contact@cc-france.org",
    "url": "https://github.com/Para-FR"
  },
  "homepage": "https://github.com/Para-FR/wd-framework",
  "repository": "https://github.com/Para-FR/wd-framework",
  "license": "MIT",
  "keywords": [
    "webdev", "web-development", "react", "nextjs",
    "vue", "typescript", "frontend", "backend",
    "fullstack", "claude-code"
  ]
}
```



WD Framework : Commande Exemple

commands/implement.md (Simplifié)

```
---  
description: Feature and code implementation with intelligent persona activation  
arguments:  
  - name: feature-description  
    description: Description of feature to implement  
    required: true  
flags:  
  - name: type  
    description: Implementation type (component|api|service|feature)  
  - name: framework  
    description: Target framework (react|vue|angular|nextjs)
```



WD Framework : Agent Exemple

`agents/frontend.md` (Simplifié)

```
---  
description: Frontend development specialist with Magic MCP integration  
subagent_type: frontend-specialist  
---
```

You are a frontend development specialist.

Expertise

- Modern UI frameworks (React, Vue, Angular)
- Accessibility (WCAG 2.1 AA compliance)



WD Framework : Installation

Pour les Utilisateurs

```
# Méthode 1 : Via Marketplace (Recommandé)  
/plugin marketplace add https://github.com/Para-FR/wd-framework  
/plugin install wd
```

```
# Méthode 2 : Installation Directe
```



WD Framework :

Utilisation

Exemples Concrets

```
# Analyse de codebase  
/wd:analyze @src/ --focus security
```

```
# Implémentation composant  
/wd:implement LoginForm --type component --framework react
```



WD Framework :

Métriques

Impact Réel



Performance

- **+30-50%** meilleurs résultats (wave)
- **+40-70%** gain de temps (agents)
- **2-5x** plus rapide (MCP)






Best Practices

Créer un Plugin de Qualité



Structure

-  **Nommage clair** - Descriptif et mémorable
-  **Versioning semantic** - Suivre semver.org
-  **Documentation complète** - README + exemples



Best Practices (suite)

Qualité et Maintenance



Testing

-  **Tester manuellement** - Avant publication






Best Practices (fin)

Communauté et Distribution



Open Source

-  **License claire** - MIT recommandé
-  **Contributing guide** - Accueillir contributions
-  **Code of conduct** - Environnement sain

Pièges à Éviter

Erreurs Communes

Structure

- Paths relatifs dans plugin.json
- Commandes sans description
- Versioning incohérent

Commandes



Debugging Plugins

Résolution Problèmes Courants

```
# Plugin non trouvé après installation
/plugin list                # Vérifier présence
ls ~/.claude/plugins/      # Check filesystem

# Commande non reconnue
/plugin info mon-plugin    # Vérifier metadata
cat ~/.claude/plugins/mon-plugin/.claude-plugin/plugin.json

# Mise à jour qui ne fonctionne pas
/plugin marketplace update mon-marketplace
```



Roadmap Système Plugins

Évolutions Futures



Court Terme

- **Plugin templates** - Générateurs scaffolding



Idées de Plugins

Inspirations pour CC France



Développement

- **Framework-specific** - Django, FastAPI, NestJS workflows
- **Testing** - Pytest, Jest, Cypress automation
- **Database** - PostgreSQL, MongoDB, Redis helpers



Ressources

Documentation Officielle



Claude Code

- **Docs** : docs.claude.com/claude-code
- **GitHub** : github.com/anthropics/claude-code
- **Community** : Discord Claude



Démo Live

Créons un Plugin Ensemble !



Objectif

Créer un plugin simple "python-helper" avec :

- Commande `/python:lint` - Linting Python files
- Commande `/python:test` - Run pytest with coverage



Récapitulatif

Points Clés

1. 🛠️ **Plugins** = Extensions Claude Code réutilisables
2. 📦 **Marketplace** = Distribution GitHub-based
3. 📁 **Structure** = .claude-plugin/ + commands/ + agents/
4. 💬 **Commandes** = Markdown avec front matter YAML
5. 🤖 **Agents** = Task tool avec subagent_type

Votre Tour !

Challenge pour CC France

Créez Votre Premier Plugin

Mission : D'ici 2 semaines

1. Identifier un besoin récurrent dans vos workflows
2. Créer un plugin avec 2-3 commandes
3. Le publier sur GitHub

 **Merci !**

Questions & Discussion

Créez, Partagez, Innovez

avec Claude Code Plugins & Marketplace

Ressources

- WD Framework : github.com/Para-FR/wd-framework
- Documentation : docs.claude.com/claude-code
- Community : Discord CC France

Contact

- Para CC-France
- contact@cc-france.org

*Présentation créée avec ❤️ pour Claude Code France
Octobre 2025 - Let's build the ecosystem together! 🚀*