

# Linux embarqué avec Yocto Project

**Christophe BLAESS**

[christophe.blaess@logilin.fr](mailto:christophe.blaess@logilin.fr)  
<https://www.blaess.fr/christophe/>



*Ingénierie et formations sur Linux et les logiciels libres*  
<https://www.logilin.fr>

## Avant-propos

Ce support de formation est distribué sous licence **Creative Commons 4.0**



*(Attribution - Partage dans les mêmes conditions).*

Vous êtes libres de copier et partager ce document, en mentionnant son origine. Si vous l'intégrez dans un contenu plus vaste, ce dernier devra être distribué avec les mêmes droits.

Ce cours a été rédigé en utilisant des logiciels libres sur système d'exploitation Linux :

- *LibreOffice Writer* pour le support et la mise en page
- *LibreOffice Draw* pour les dessins vectoriels
- *Gimp* pour les images bitmap
- *Excalidraw* (en ligne) pour certains schémas

« Linux » est un nom déposé par Linus Torvalds et administré par le « Linux Mark Institute ».

Le logo du pingouin « Tux » a été créé par Larry Ewing.

« Yocto Project » est une marque déposée par la Linux Foundation

Linux embarqué avec Yocto Project

ILY v. 8.7

<https://www.blaess.fr/christophe/>

<https://www.logilin.fr>

# Introduction

Ce cours est une prise en main de l'**outil Yocto Project** pour produire un système Linux embarqué et développer le code applicatif.

Les travaux pratiques nécessitent une certaine familiarité avec la ligne de commande du système Linux.

Les démonstrations et **travaux pratiques** se dérouleront sur PC x86, sur émulateur Qemu, et sur cartes *Raspberry Pi* ou *Beaglebone Black* (processeurs ARM).

Le **support de cours** en version PDF est disponible ici :

<https://www.logilin.fr/files/support-ILY.zip>



# Plan de la formation

## I – Créer un système Linux embarqué

- Environnement embarqué : concepts, composants, outils, *build systems*.
- Production d'une image standard : environnement, configuration, compilation, test.
- Composition d'un système Linux embarqué : matériel, *bootloader*, noyau, *init*...

## II – Personnalisation du système

- Découverte du système : connexion, système de fichiers, arborescence.
- Personnalisation de l'image : création d'une image, administration, recettes.
- Ajout de *packages* : standards Poky, OpenEmbedded, configuration de Busybox.

## III – Configuration avancée du système

- Extensions de recettes : surcharge de fichiers, configuration réseau.
- Création et application de *patches* : fichiers de données, fichier compilé, devtool.
- Noyau Linux et *device tree* : configuration, patch du *kernel*, patch du *device tree*.

## IV – Développement du code métier

- Ajout de scripts personnalisés : recettes, fichiers de configuration.
- Compilation du code métier : cross-compilation, *toolchain* Yocto.
- Débogage distant : *gdbserver*, outils d'aide au débogage.
- Intégration du code métier : recettes, lancement au démarrage.

## Annexe A – Bibliographie

- Livres.
- Articles en ligne.

## Annexe B – Aides mémoire

- Aide mémoire des commandes Linux.
- Aide mémoire de la programmation shell.