

---

## Ouvrir les codes sources

Etalab

etalab<sup>gouv.fr</sup>

23/05/2024

## Table des matières

<b>1</b>	<b>Introduction</b>	<b>3</b>
1.1	À qui s'adresse ce guide? . . . . .	3
1.2	À quoi sert-il? . . . . .	3
1.3	Clarifier : quels degrés d'ouverture pour les codes sources? . . . . .	3
1.4	Prioriser : quels logiciels ouvrir à quel degré? . . . . .	4
1.4.1	Exemples de mise en oeuvre . . . . .	4
1.5	Ouvrir : comment encourager les contributions? . . . . .	5
1.6	Communiquer : comment annoncer une ouverture de code source? . . . . .	5
1.6.1	Mettre un lien vers le site web du projet . . . . .	5
1.6.2	Dire où trouver les dépôts de code source . . . . .	6
1.6.3	Indiquer qui contribue déjà au code source . . . . .	6
1.6.4	Indiquer si des contributions sont attendues . . . . .	6
1.6.5	Prévenir les équipes qui développent le logiciel . . . . .	6
1.6.6	Rappeler pourquoi le code source est libre . . . . .	6
1.6.7	Montrer comment le logiciel dépend d'un écosystème . . . . .	6
1.6.8	Si le logiciel est sensible question sécurité, dire ce qui a été fait et va être fait . . . . .	7
1.7	Maintenance de ce document et contributions . . . . .	7
<b>2</b>	<b>Ce document n'est pas...</b>	<b>7</b>
2.1	Pas un guide des bonnes pratiques « open source » . . . . .	7
2.2	Pas un guide sur la transparence algorithmique . . . . .	7
2.3	Pas un guide sur la gouvernance des projets libres . . . . .	8
2.4	Finalité de ce guide . . . . .	8
<b>3</b>	<b>Cadre juridique</b>	<b>8</b>
3.1	Pour qu'un code source soit communicable . . . . .	8
3.2	Licences applicables à la publication d'un code source . . . . .	9
3.3	Guide juridique interactif . . . . .	9

# 1 Introduction

## 1.1 À qui s'adresse ce guide ?

Ce guide s'adresse aux organismes publics, et plus particulièrement aux personnes chargées de l'ouverture des codes sources logiciels dans ces organismes. Accessoirement, il intéressera les prestataires à qui les organismes publics demandent d'aider à l'ouverture des codes sources qu'ils livrent dans le cadre d'un marché public.

Ce guide suppose que vous avez bien compris ce que ce document n'est pas, que vous avez pris connaissance de la [politique de contribution aux logiciels libres de l'État](#) et du cadre juridique dans lequel s'applique la communicabilité des codes sources logiciels produits par des organismes publics.

## 1.2 À quoi sert-il ?

Ce guide sert à aider les organismes publics à définir une politique d'ouverture des logiciels libres qu'ils produisent, dans le cadre de la mise en oeuvre de la loi pour une République numérique du 7 octobre 2016.




Il vient en complément de la Politique de contribution aux logiciels libres de l'État, [publiée](#) en mai 2018, laquelle est doublement limitée : (1) elle ne s'adresse pas aux collectivités territoriales ; (2) elle acte le principe selon lequel les agents publics peuvent publier du code source et contribuer à des logiciels libres, mais elle n'aide pas les organismes publics à répondre à la question « Quels logiciels ouvrir en priorité ? »

Ce guide a pour vocation de répondre aux questions :

- Quels logiciels ouvrir en priorité ?
- Comment bien communiquer autour de la publication d'un logiciel libre ?

## 1.3 Clarifier : quels degrés d'ouverture pour les codes sources ?

Nous proposons de distinguer les quatre degrés d'ouverture suivants :

-  **Niveau A - contributif** : Le code source est publié, les contributions extérieures sont activement recherchées et traitées.
-  **Niveau B - ouvert** : Le code source est publié, les contributions extérieures sont traitées mais non activement recherchées.
-  **Niveau C - publié** : Le code source est publié mais les contributions extérieures ne sont pas traitées.

-  **Niveau D - non-communicable** : Le code source n'est pas communicable au public.

## 1.4 Prioriser : quels logiciels ouvrir à quel degré ?

Tous les logiciels développés par un organisme public n'ont pas vocation à être ouverts au même degré.

Nous proposons trois critères :

1. Le logiciel est-il **un module utile à d'autres logiciels libres** (ou un logiciel « monolithique » sans utilité pour d'autres logiciels libres) ?
2. Le logiciel répond-il à un **besoin générique** (ou à un besoin spécifique à l'organisme qui le produit) ?
3. L'**utilisateur final** du logiciel a-t-il un profil technique (développeur, datascientiste ou designer) ?

**Le niveau A** est recommandé pour les logiciels répondant à au moins deux critères; le niveau B est recommandé pour ceux répondant à au moins un critère; le niveau C pour ceux ne répondant à aucun de ces trois critères (par ex. un logiciel métier très spécifique, dont aucune partie ne peut être réutilisée ailleurs et dont les utilisateurs ne sont pas du tout des contributeurs potentiels.)

Pour les logiciels ne répondant à aucun des trois critères, le niveau D est admissible, tant qu'aucun citoyen n'exige la communication du code source en question, selon le cadre juridique défini dans la loi pour République numérique.

Bien sûr, ces critères sont *relatifs* : la modularité, la généricité et le potentiel de contribution des utilisateurs ne s'évaluent pas dans l'absolu. Ces notions aident seulement à **prioriser les ouvertures logicielles**. Le but est de **canaliser son énergie** sur les logiciels qui ont un bon potentiel contributif et **de communiquer clairement** sur la posture de l'administration dans le cas des publications simples.

### 1.4.1 Exemples de mise en oeuvre

- Une collectivité territoriale développe un outil de correction grammaticale pour LibreOffice. Ce logiciel est un module d'un logiciel libre existant et il répond à un besoin générique : il est pertinent d'en faire un **logiciel libre « contributif »** (niveau A).
- Une administration développe un outil pour organiser la collecte de données sur le web (*scraping*). C'est un outil web « monolithique » mais qui répond à un besoin rencontré hors de l'administration : il peut être publié comme **logiciel libre « ouvert »** (niveau B).

- Une administration centrale développe un thème pour les sites qu'elle publie à l'aide de Jekyll. Ce thème est un module d'un logiciel libre existant mais il répond à un besoin spécifique de l'organisme public : son code source peut être publié, mais sans recherche active de contributeurs ni maintenance particulière à l'égard des contributions extérieures (niveau C).

Chaque organisme peut tenter de prioriser les logiciels à ouvrir en fonction de ces critères.

## 1.5 Ouvrir : comment encourager les contributions ?

Lorsque vous souhaitez encourager les contributions sur les logiciels libres que vous publiez, quelles bonnes pratiques mettre en oeuvre ? Ci-dessous une liste non-exhaustives d'idées :

- Ajoutez ces sections dans votre README :
  - **Auteur** : qui est l'auteur ? Comment le contacter ?
  - **Licence** : quelle est la licence ? Avec un lien vers votre fichier LICENSE .md dans le dépôt.
  - **Contributions** : souhaitez-vous des contributions ? Si oui, sur quels aspects de votre projet ? En fonction des profils de contributeurs, par où peuvent-ils commencer ? Éventuellement, vous pouvez préciser ici quelle est la *gouvernance* du projet (qui décide et comment).
- Utiliser des mots-clés pour votre dépôt :
  - [good-first-issue](#)
  - [beginner-friendly](#)
- Utiliser des mots-clés pour vos issues :
  - [good-first-issue](#)
  - [first-timers-only](#)

Vous trouverez d'autres conseils sur [www.firsttimersonly.com](http://www.firsttimersonly.com).

Dans tous les cas : **expérimentez** et **communiquez** !

## 1.6 Communiquer : comment annoncer une ouverture de code source ?

Voici quelques recommandations lorsqu'une administration communique sur la mise à disposition d'un logiciel libre.

### 1.6.1 Mettre un lien vers le site web du projet

Les projets libres ont souvent une page web dédiée. C'est le point d'entrée pour les utilisateurs et les contributeurs potentiels. À défaut d'un site web, la page de README .md du logiciel suffira.

### **1.6.2 Dire où trouver les dépôts de code source**

Lorsqu'on annonce un logiciel libre, le premier réflexe d'un développeur sera d'aller voir le code source : pour comprendre le problème que le logiciel aide à résoudre, pour connaître la licence et les conditions de contribution au logiciel.

### **1.6.3 Indiquer qui contribue déjà au code source**

Lorsqu'une administration publie du code source libre, elle a peut-être développé le code elle-même, ou bien l'a financé. Elle a peut-être reçu de l'aide d'autres agents publics ou de citoyens. Savoir qui est en charge de la gouvernance du projet et qui sont les auteurs est une information importante.

### **1.6.4 Indiquer si des contributions sont attendues**

En général, on ouvre le code source d'un logiciel parce qu'on espère des contributions extérieures. Ce n'est pas systématiquement le cas pour un organisme public, qui peut simplement souhaiter rendre son code source public, sans vouloir gérer des contributions. Dans les deux cas, il est important d'anticiper les attentes en étant très explicite à ce sujet.

### **1.6.5 Prévenir les équipes qui développent le logiciel**

Dès qu'on annonce un logiciel libre, il faut s'attendre à ce qu'il soit testé et à ce que des questions soient posées ou des retours de bugs envoyés. Le mieux est de prévenir les équipes qui développent le logiciel pour que celles-ci puissent se montrer réactives. La première impression qu'on donne à la communauté des utilisateurs et des contributeurs potentiels est importante.

### **1.6.6 Rappeler pourquoi le code source est libre**

Une administration peut avoir plusieurs raisons de publier le code source des logiciels qu'elle développe ou fait développer.

En général, on peut se référer à l'un des trois piliers évoqués par la loi pour une République numérique pour la gestion des systèmes d'information : maîtrise, pérennité, indépendance.

### **1.6.7 Montrer comment le logiciel dépend d'un écosystème**

Les logiciels libres sont souvent construits à partir d'autres logiciels libres et peuvent parfois servir de briques pour d'autres solutions. C'est important d'en avoir conscience en communiquant sur le

logiciel, car une critique émise (ou un retour de bug) pourra en fait porter sur un logiciel qui n'est pas développé par l'équipe.

### 1.6.8 Si le logiciel est sensible question sécurité, dire ce qui a été fait et va être fait

Pour la communication autour de forts enjeux liés à leur sécurité, il est important de souligner ce point dans la communication, en indiquant ce qui a été fait et ce qui sera fait.

Par exemple, si le logiciel a fait l'objet d'un audit de sécurité par l'ANSSI ou si le logiciel a déjà été testé auprès d'agents qui s'y connaissent bien en sécurité, dire quand et quels ont été les résultats. Si une opération de "*bug bounty*" (chasse aux bugs) est prévue, dire quand et quelles sont les attentes.

## 1.7 Maintenance de ce document et contributions

Ce document est maintenu par Bastien Guerry à Etalab.

Pour toute question, vous pouvez écrire à [opensource@data.gouv.fr](mailto:opensource@data.gouv.fr) ou directement à [bastien.guerry@data.gouv.fr](mailto:bastien.guerry@data.gouv.fr).

## 2 Ce document n'est pas...

### 2.1 Pas un guide des bonnes pratiques « open source »

Il ne détaille pas les **bonnes pratiques** d'un projet libre, que ce soit sur le plan des pratiques techniques ou de la communication au sein et autour des projets. Des éléments de ces bonnes pratiques peuvent être trouvés dans la [Politique de contribution de l'État aux logiciels libres](#).

### 2.2 Pas un guide sur la transparence algorithmique



Il ne porte pas sur la question de la **transparence algorithmique** : la possibilité pour un citoyen de demander des comptes sur les décisions individuelles le concernant et mobilisant un traitement algorithmique ne préjuge pas de l'existence de ce traitement sous forme de logiciel ni de la communicabilité du logiciel, si celui-ci existe.

Sur cette question de la transparence algorithmique, des **degrés d'appropriabilité** peuvent être conçus : le « degré 0 », où seul le code source du logiciel est publié; le « degré 1 », où figurent des explications lisibles par des non-techniciens sur les algorithmes mis en oeuvre dans le logiciel (documentation statique); le « degré 2 », où l'utilisateur peut non seulement lire les explications au

sujet des algorithmes, mais tester la mise en oeuvre de ces algorithmes via un simulateur les utilisant ou un autre dispositif interactif.

Voir le « [guide des algorithmes publics](#) » publié par Etalab.

## 2.3 Pas un guide sur la gouvernance des projets libres

Ce n'est pas un guide pour la **gouvernance des projets de logiciels libres** : ce sujet est un sujet à part entière. Nous distinguons la maintenance *a minima* (niveau B ) et la recherche active de contributeurs (niveau A ) mais ces aspects ne couvrent qu'une petite partie de la question de la gouvernance.

## 2.4 Finalité de ce guide

**Sa finalité** est d'aider les organismes publics à prioriser les codes sources pouvant être communiqués afin de construire une stratégie proactive d'ouverture logicielle.

# 3 Cadre juridique

Toute entité chargée d'une mission de service public doit publier tout document produit ou reçu dans le cadre de cette mission, quelle qu'en soit la date, le lieu de conservation et le support. Les codes sources, en tant que documents administratifs, relèvent de cette obligation (voir l'avis CADA du 8 janvier 2015 n°[20144578](#)).

Les codes sources concernés sont, au même titre que n'importe quelle autre donnée administrative publiable en open data, celles « dont la publication présente un intérêt économique, social, sanitaire ou environnemental. »

Pour les licences, voir les articles [L323-2](#) et [D323-2-1](#) du Code des relations entre le public et les administrations.

## 3.1 Pour qu'un code source soit communicable

- L'obligation de communicabilité porte sur les collectivités de plus de 3500 habitants et les organismes publics de plus de 50 agents.
- L'organisme public ouvrant le code source doit en avoir la propriété intellectuelle.



- Le code source doit être « achevé » : dès lors qu’une version du code est mise en oeuvre dans l’administration, cette version est considérée comme « achevée ». Notamment une version dite bêta ou inférieure à 1.0, si elle est effectivement utilisée, est bien achevée et communicable.
- Sa communication ne doit pas porter atteinte :
  - au secret commercial et industriel;
  - à la sûreté de l’État, à la sécurité publique, à la sécurité des personnes ou à la sûreté des systèmes d’information des administrations;
  - à la recherche et à la prévention, par les services compétents, d’infractions de toute nature.

En dehors de ces limites, toute personne ou toute administration peut demander la communication d’un code source.

### **3.2 Licences applicables à la publication d’un code source**

La liste des licences sous lesquelles un code source peut être placé au moment de sa diffusion est [accessible ici](#).

### **3.3 Guide juridique interactif**

Pour savoir si le code source d’un logiciel développé et utilisé par votre organisme public est communicable, nous vous invitons à tester ce [guide juridique interactif](#).