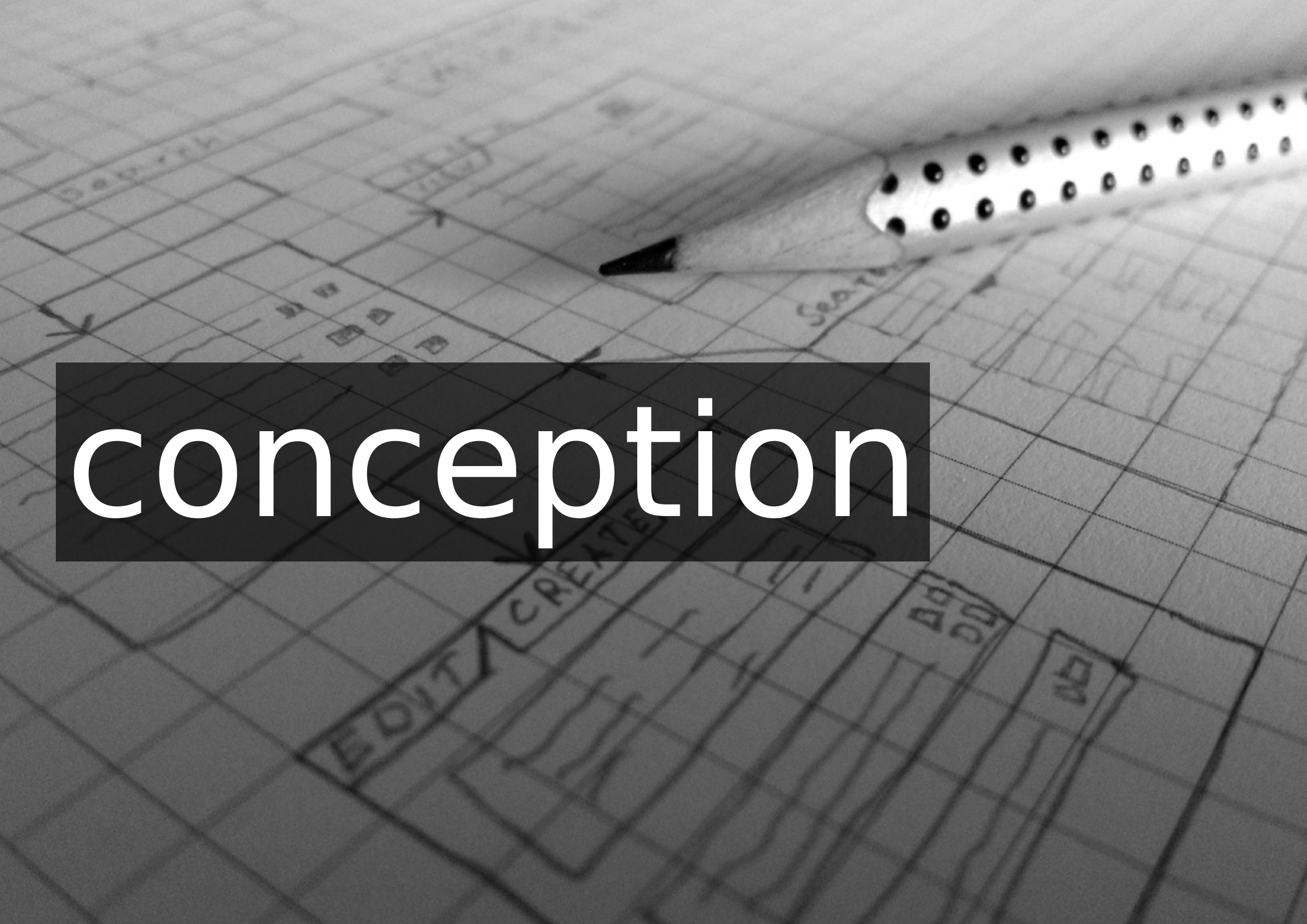
Créer une application:

le guide incomplet...
mais pratique quand même

À la base de chaque application :

idée


conception

implémentation

```
string sInput;
int iLength, iN;
double dblTemp;
bool again = true;

while (again) {
    iN = -1;
    again = false;
    getline(cin, sInput);
    system("cls");
    stringstream(sInput) >> dblTemp;
    iLength = sInput.length();
    if (iLength < 4) {
        again = true;
        continue;
    } else if (sInput[iLength - 3] != '.') {
        again = true;
        continue;
    } while (++iN < iLength) {
        if (isdigit(sInput[iN])) {
            continue;
        } else if (iN == (iLength - 3) ) {
```

```
#include <stdio.h>

int main (int argc, char *argv[])
{
    printf ("Hello, World !");
    return 0;
}
```

```
./hello-world
Hello, World !
```


profit !
#penelope

Simpl(ist)e, et pourtant...

Pour créer, il faut des outils

Même pour une application triviale, vous aurez utilisé...


• éditeur de texte
• langage de programmation
• compilateur/interpréteur
• système d'exploitation

Quelques questions à se poser avant de commencer

Quelle plate-forme(s) logicielle(s) ?

Windows ? GNU/Linux ? macOS ?
iOS ? Android ? …
Navigateur Web ?

Quelle version(s) ?

Windows 10 ? Fedora 25 ? ...

Quelle plate-forme(s) matérielle(s) ?

x86 ? ARM ? ...

Quel langage... pour quel usage ?

contraintes différentes → choix différents

portabilité
performance
maintenance
écosystème

Quel langage... compilé ?

C, C++, Java, ...

Quel langage... interprété ?

javascript, python, perl, ...

Cible différente de l'hôte ?

→ compilation croisée
(si utilisation d'un langage compilé)

Quelle version de standard ?

C++: C++98, C+03, C++11

C: C89, C99, C11

Python: 2.7, 3.x


etc.

« Il est tentant, quand on n'a pour seul outil un marteau, de tout traiter comme un clou »

— Abraham Maslow
The Psychology of Science:
A Reconnaissance
(1966)


Trouvez *vos* outils


30 langages = 30 marteaux


L'éditeur de texte

Indépendant

Vim
Emacs
SublimeText
Notepad++
etc.

Intégré dans un IDE

Eclipse
Visual Studio
Code::Blocks
GNOME Builder
etc.

Ce qu'on attend d'un éditeur

- coloration syntaxique
- marque-pages
- indentation automatique
- macros
- remplacement de texte
- multi-documents
- sélection rectangulaire
- extensible


+1

Bonnes pratiques d'édition :

Respectez les conventions
du document que vous éditez...

...ou adoptez les conventions du langage

Python PEP8: https://www.python.org/dev/peps/pep-0008/

Respectez l'indentation

→ configurez votre éditeur !

- facilite la lecture des diffs
- évite l'attribution
- sémantiquement important

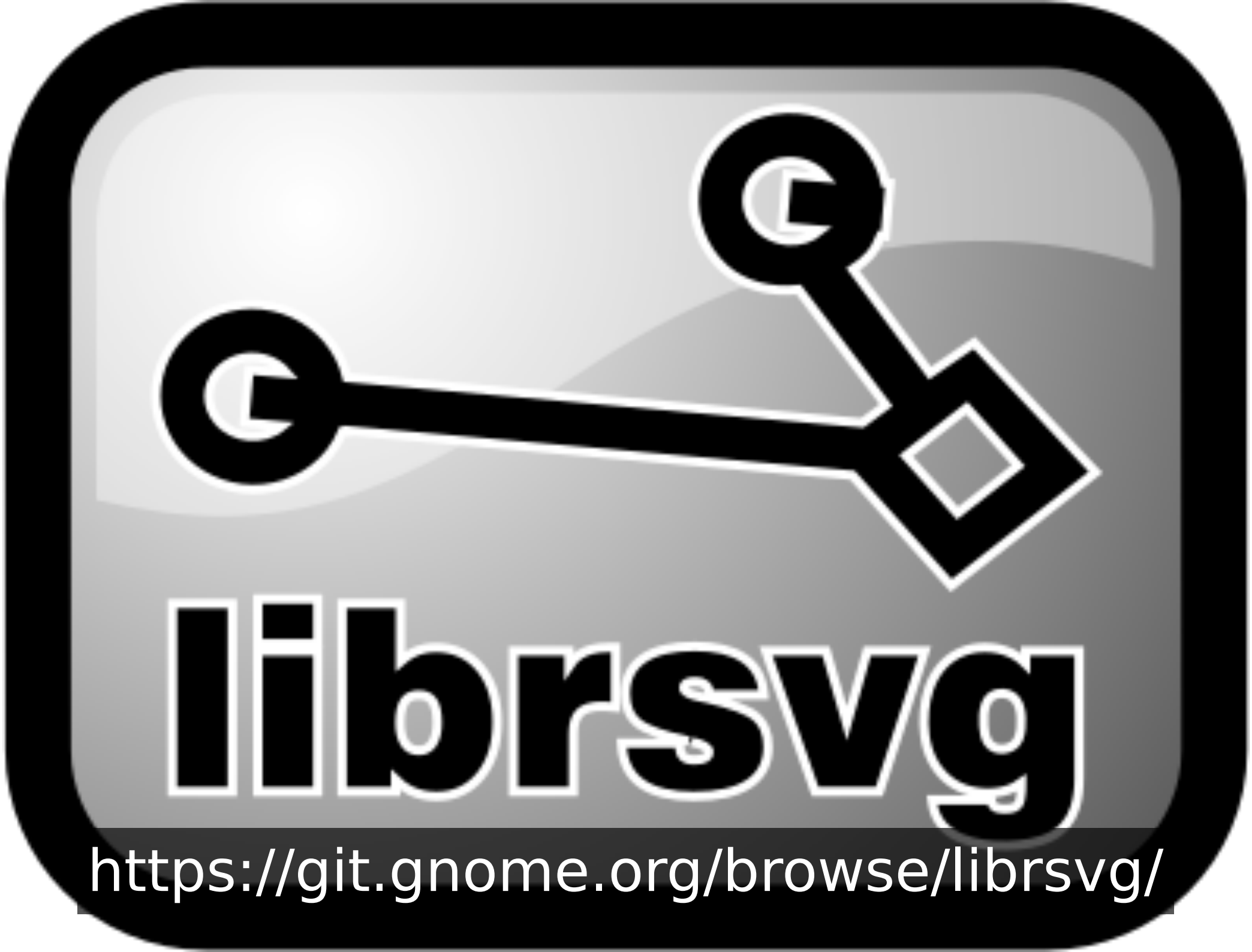
Python, Makefile


Languages
SANGIORGI & WALKER
The π-calculus
A Theory of Mobile Processes
CAMBRIDG
Guzdial
Rose
Squeak
Open Personal Computing
and Multimedia
DYBVIG
THE SCHEME PROGRAMMING LANGUAGE
ANSI SCHEME
SECOND EDITION
PTR
Nelson
Systems Programming with
Modula-3
Learning Python
Lutz & Ascher
THIRD EDITION
Programming Perl
Wall,
Christiansen
& Orwant
Miranda™
The Craft of
Functional
Programming
Thompson
ULLMAN
ELEMENTS OF ML PROGRAMMING
ML97 EDITION
The Little MLer
Felleisen and Friedman
The Java™ Programming Language
Second Edition
Arnold
Gosling
JAVA
Apple
PRESS
The Dylan
Reference Manual
Shalit
Addison
Wesley
STROUSTRUP
C++
THE C++ PROGRAMMING LANGUAGE
THIRD EDITION
KERNIGHAN · RITCHIE
THE C PROGRAMMING LANGUAGE
SECOND EDITION
PTR
Software Construction
Ada 95
ADDISON
WESLEY

Choix du langage

le plus adapté... parmi ceux que maîtrisez

Occasion d'apprendre un nouveau langage ?


librsvg
https://git.gnome.org/browse/librsvg/


C
2001
Federico
XML → bitmap

Migration C → Rust

oxydation

https://people.gnome.org/~federico/news-2016-10.html#25

C

langage difficile
gestion mémoire
sécurité (buffer overflow)

C : risques

fichiers mal formés :

→ crash

→ faille de sécurité

Critique en production

Rust

soutenu par Mozilla et Samsung
utilisé dans Firefox (Servo, Quantum)

https://www.rust-lang.org/fr/

Mozilla Servo (moteur de rendu, projet de recherche)
Quantum (merge de Servo dans Gecko)
https://wiki.mozilla.org/Quantum

Objectifs de Rust

parallélisme
gestion mémoire
sécurité

pas de null, pas de ramasse miettes

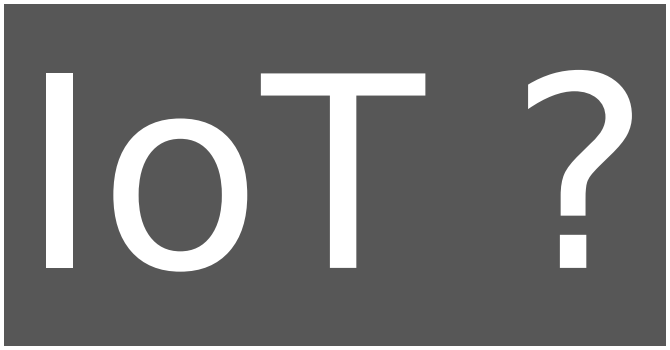
```
fn main() {
    println!("Hello, Rust!");
}
```

Migration → Rust

intégrable dans du code C
migration par morceaux

« Vous n'avez pas besoin de tout réécrire en une fois. Je ne connais pas d'autre langage qui permette cela. »

— Federico Mena Quintero, 2016


IoT ?

Bonnes pratiques de codage :

Respectez la casse

snake_case
CamelCase
mixedCase

Respectez le codage (charset)

```
printf ("©o®ação = ♥");
```

UTF-8 vs ~~ISO 8859-1~~

Écrivez du code portable

(essayez)

chemins du système de fichiers

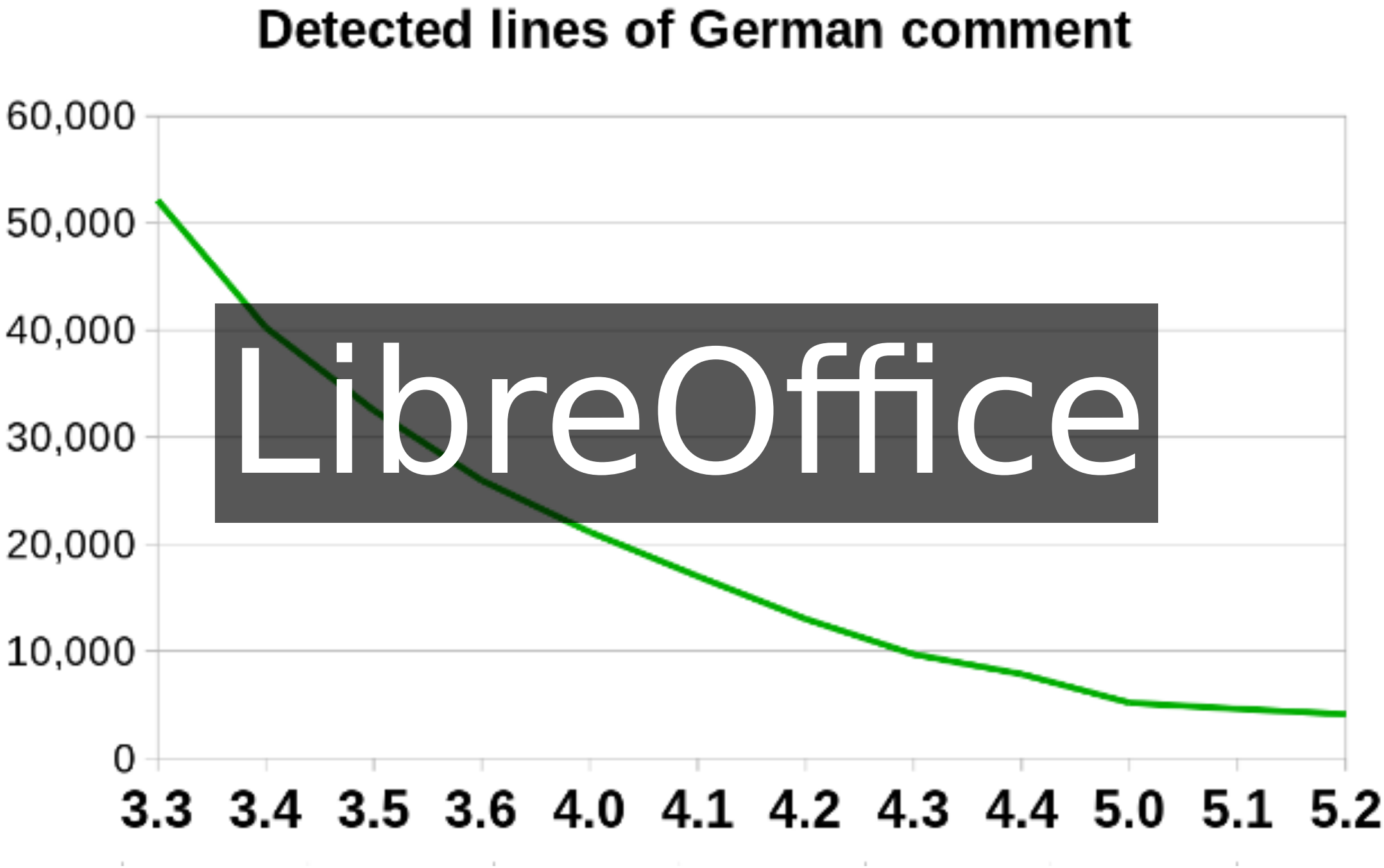
os.path.join()
g_build_path() via glib

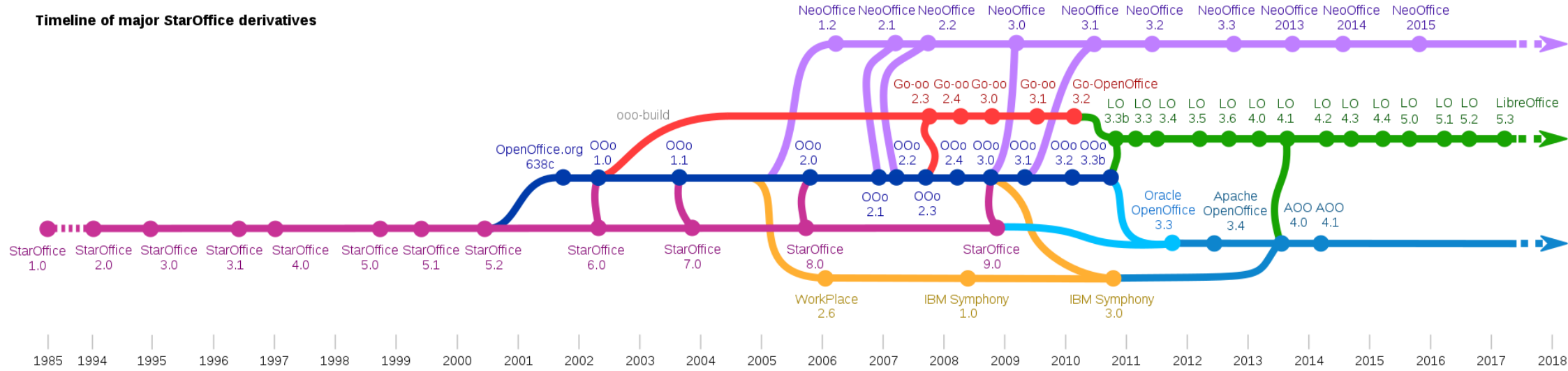
Respectez la langue

fonctions
variables
commentaires

en anglais !

→ plus court
→ plus facile à diffuser (S.O ☺)
→ maintenable


Detected lines of German comment
60,000
50,000
40,000
30,000
20,000
10,000
0
LibreOffice
3.3
3.4
3.5
3.6
4.0
4.1
4.2
4.3
4.4
5.0
5.1
5.2


Timeline of major StarOffice derivatives
StarOffice 1.0
StarOffice 2.0
StarOffice 3.0
StarOffice 3.1
StarOffice 4.0
StarOffice 5.0
StarOffice 5.1
StarOffice 5.2
StarOffice 6.0
StarOffice 7.0
StarOffice 8.0
StarOffice 9.0
OpenOffice.org 638c
OOo 1.0
OOo 1.1
OOo 2.0
OOo 2.1
OOo 2.2
OOo 2.3
OOo 2.4
OOo 3.0
OOo 3.1
OOo 3.2
OOo 3.3b
ooo-build
Go-oo 2.3
Go-oo 2.4
Go-oo 3.0
Go-oo 3.1
Go-OpenOffice 3.2
NeoOffice 1.2
NeoOffice 2.1
NeoOffice 2.2
NeoOffice 3.0
NeoOffice 3.1
NeoOffice 3.2
NeoOffice 3.3
NeoOffice 2013
NeoOffice 2014
NeoOffice 2015
LO 3.3b
LO 3.3
LO 3.4
LO 3.5
LO 3.6
LO 4.0
LO 4.1
LO 4.2
LO 4.3
LO 4.4
LO 5.0
LO 5.1
LO 5.2
LibreOffice 5.3
Oracle OpenOffice 3.3
Apache OpenOffice 3.4
AOO 4.0
AOO 4.1
WorkPlace 2.6
IBM Symphony 1.0
IBM Symphony 3.0
1985
1994
1995
1996
1997
1998
1999
2000
2001
2002
2003
2004
2005
2006
2007
2008
2009
2010
2011
2012
2013
2014
2015
2016
2017
2018

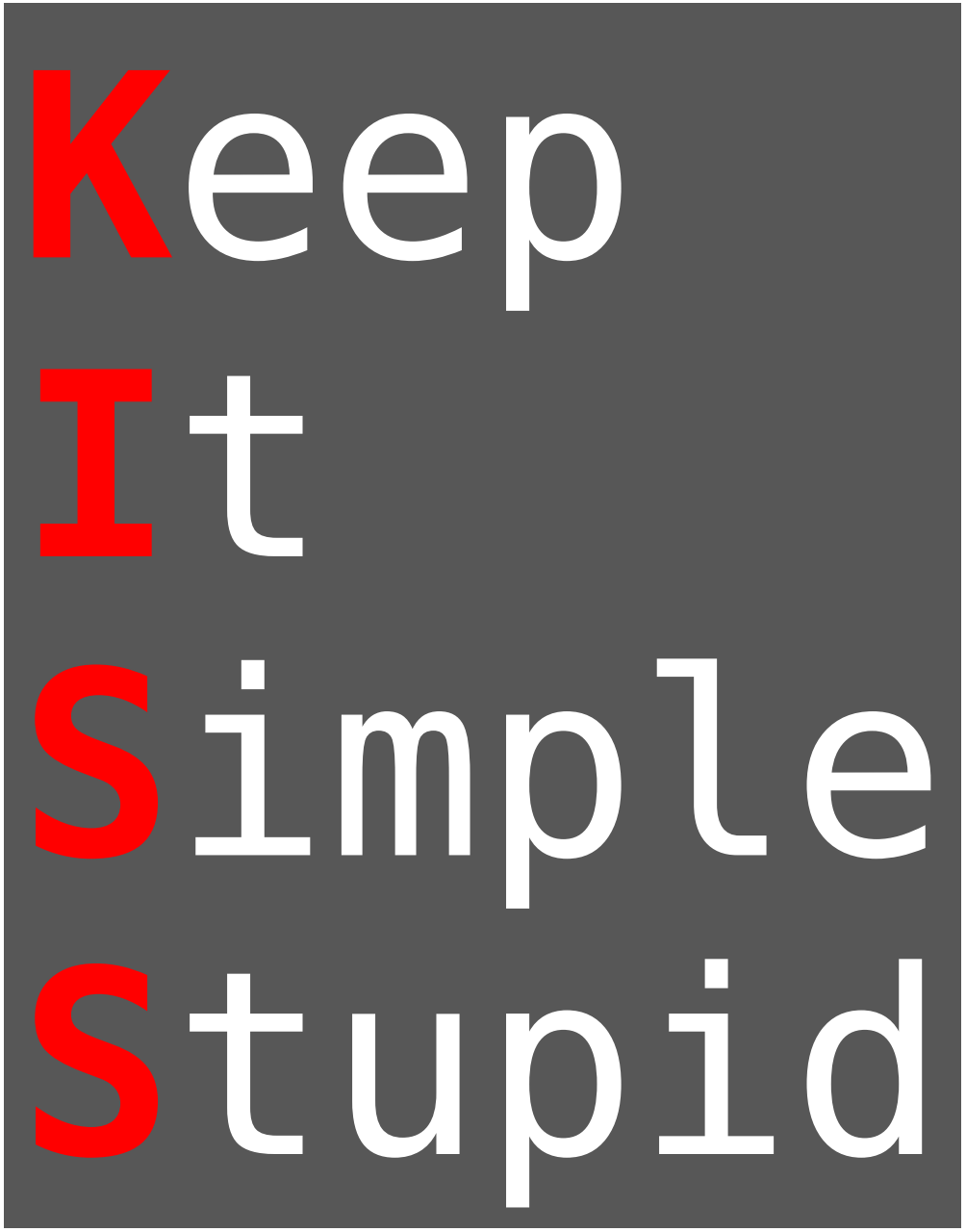
32 ans !


Anglais US
Do we get the leaders we deserve ?

Mantras du développeur


Don't
Repeat
Yourself

générateur de release notes


Keep
It
Simple
Stupid

héritage multiple RATP

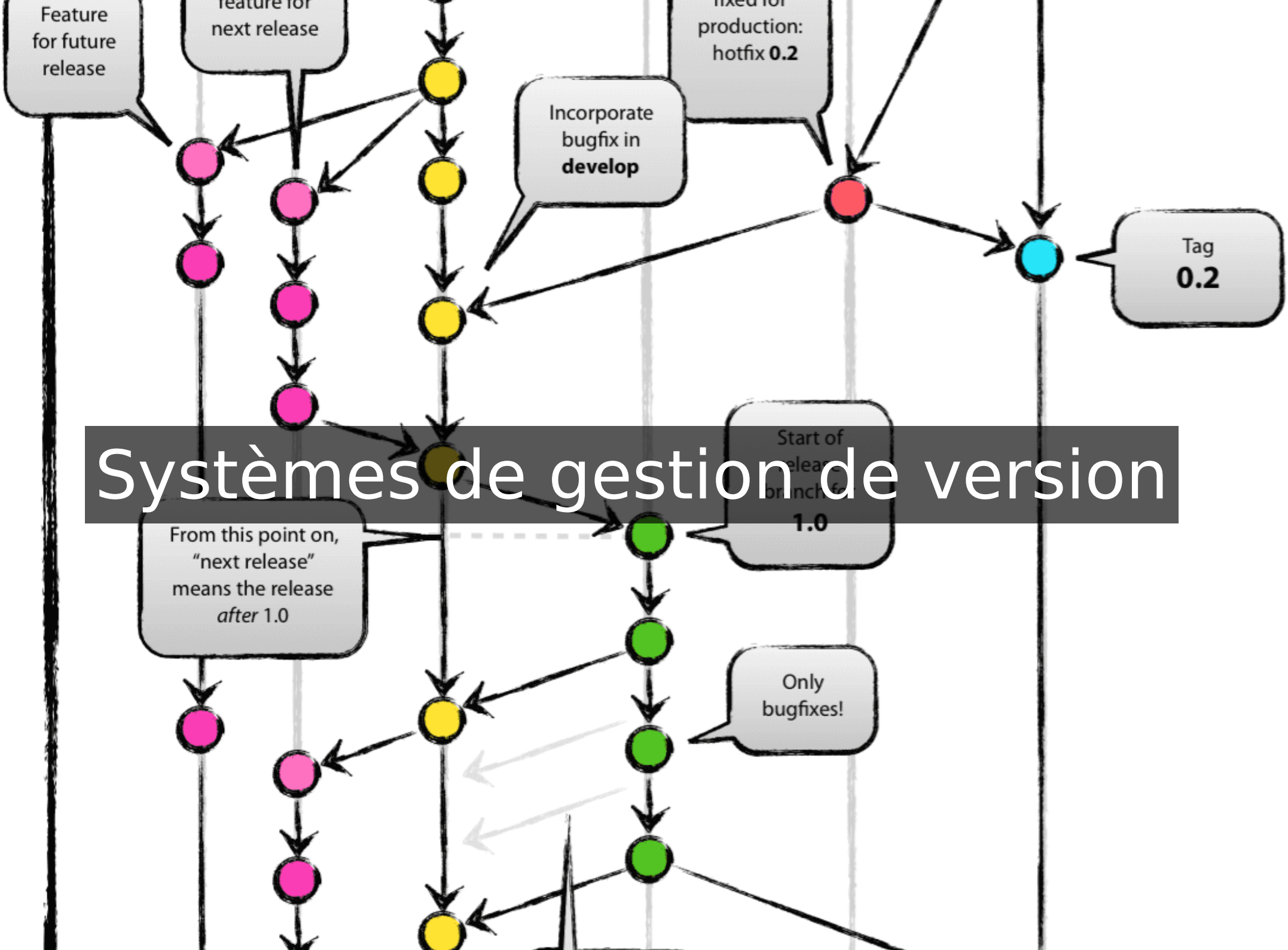

Not
Invented
Here

réimplémenter ses formats de date
listes chaînées

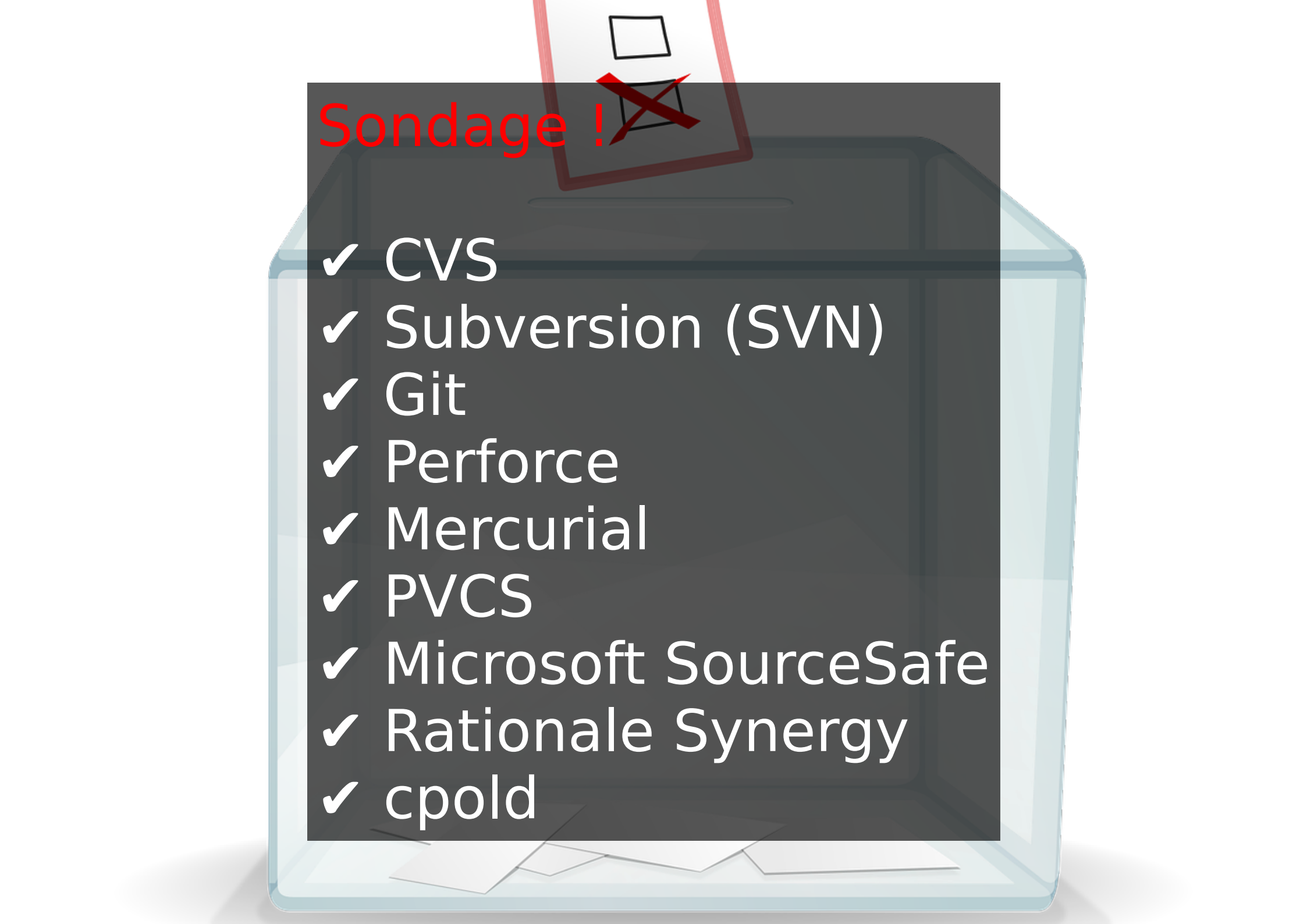
Zen of Python

https://www.python.org/dev/peps/pep-0020/


+1



Systèmes de gestion de version


Sondage !
✔ CVS
✔ Subversion (SVN)
✔ Git
✔ Perforce
✔ Mercurial
✔ PVCS
✔ Microsoft SourceSafe
✔ Rationale Synergy
✔ cpold

Le test de Joël

http://french.joelonsoftware.com/Articles/TheJoelTest.html

1. Utilisez-vous un système de gestion de code source ?
2. Pouvez-vous faire un build en une seule étape ?
3. Faites-vous des builds quotidiens ?
4. Avez-vous une base de données de bugs ?
5. Corrigez-vous vos bugs avant d'écrire du nouveau code ?
6. Avez-vous un planning à jour ?
7. Avez-vous une spec ?
8. Les programmeurs bénéficient-ils d'un environnement de travail calme ?
9. Utilisez-vous les meilleurs outils que vous puissiez vous payer ?
10. Avez-vous des testeurs ?
11. Les candidats écrivent-ils du code pendant leur entretien d'embauche ?
12. Faites-vous des tests d'utilisabilité de couloir ?

Définitions

checkout/clone
commit
branch
merge

Récupération des données du serveur
Enregistre des modifications
Crée une nouvelle version indépendante
Fusionne les modifications de 2 branches

Systèmes centralisés (SVN)

serveur et réseau critiques
distant = lent (latence, débit)
1 → n

Single Point Of Failure
latence -> opérations lentes
téléchargements lents -> mirroir SVN
merge impossible
commit impossible

Systèmes décentralisés (Git)

https://git-scm.com/


Linus
1991: Linux
2005: git

Git : avantages

agilité
local = rapide
branches multiples
merge facilité
1 → n ou n → n

Git : bonus

interopérabilité SVN (git-svn)
chaque clone est une sauvegarde
workflow de petits commits

☹ Git

plus grande complexité
développeur → intégrateur
gros binaires

utilisez plutôt git-annex

Git: créer un dépôt

```
git init
git add .
git commit
```

au début de votre prochain entretien technique

♥ Git

```
git rebase --interactive
```


♥ Git
git commit --amend


♥ Git
git add --patch


♥ Git
git stash


♥ Git
git bissect


♥ Git
git format-patch

Bonnes pratiques gestion de version :

diff avant commit !


+1

Build systems

Analyseurs statiques

Coala

(*lint, deblint, pylint, coala, shellcheck)

Licences

ne pas copier/coller du code sans connaitre exactement la licence et les droits
libre ne veut pas dire compatible
dwheeler
logiciel libre et licences libres (évoquer SPDX, Creative Commons pour les resources non-code)


Trouvez vos outils


Trouvez votre workflow


FIN