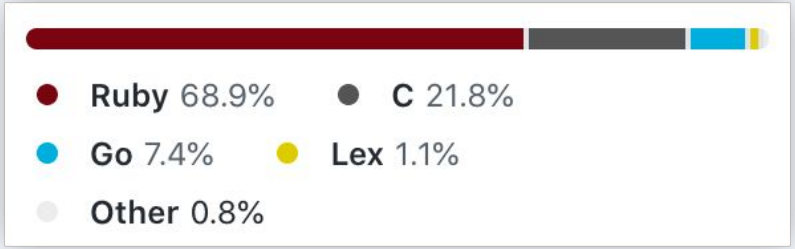
Linguist : Détecter les langages de 400+ millions de dépôts Git

Paul Chaignon | @pchaigno

Ex-mainteneur de Linguist

Linguist : Introduction

- Librairie Ruby, open source
- Utilisée par GitHub et GitLab
- Détecte 730 langages
- Détecte le code généré et le code de tiers-parties (deps)
- Utilise seulement le nom de fichier et son contenu



Who Am I?

Paul Chaignon
Software Engineer @ Cisco
Développeur eBPF pour un plugin réseau Kubernetes

Who Am I?

Paul Chaignon
Software Engineer @ Cisco
Développeur eBPF pour un plugin réseau Kubernetes

Paul Chaignon v2020
Co-mainteneur Linguist - 4 ans
Contributeur LInguist - 6 ans
Pas employé par GitHub

tmp / ??? / **rot13.bf**

Quizz 1/6

Code | Blame | 30 lines (29 loc) · 2.16 KB

```
# ROT13 cipher

-,+[                         Read first character a
    -[                       Skip forward if charac
        >>++++[>++++++++<-]  Set up divisor (32) fo
                               (MEMORY LAYOUT: divi
        <+<-[                Set up dividend (x min
            >+>+>-[>>>]      Increase copy and rema
            <[[>+<-]>>+>]    Special case: move rem
            <<<<<-           Decrement dividend
        ]                    End division loop
    ]>>>[-]+                 End skip loop; zero fo
    >--[-[<->+++[-]]]<[         Zero that flag unle
        ++++++++++++<[       If flag then set up di
                               (MEMORY LAYOUT: zero
            >-[>+>>]         Reduce divisor; Normal
            >[+[<+>-]>+>>]   Special case: increase
            <<<<<-           Decrease dividend
        ]                    End division loop
```

Linguist

- Cas d'usages
- Contraintes
- Alternatives
- Fonctionnement
- Linguist et vous
- Conclusion

tmp / ??? / rot13.bf

Quizz 1/6

Code B .B

pchaigno.github.io/linguist/

```
first character a
forward if charac
up divisor (32) fo
EMORY LAYOUT: divi
up dividend (x mir
ease copy and rema
ial case: move rem
ement dividend
                                    loop
                                  ); zero fo
                                  zero that flag unle
++++++++++++<[                    If flag then set up di
                                    (MEMORY LAYOUT: zero
    >-[>+>>]                      Reduce divisor; Normal
    >[+[<+>-]>+>>]                Special case: increase
    <<<<<-                        Decrease dividend
]                                 End division loop
```

Linguist

- Cas d'usages
- Contraintes
- Alternatives
- Fonctionnement
- Linguist et vous
- Conclusion

tmp / ??? / **rot13.bf**

Quizz 1/6

Code | Blame | 30 lines (29 loc) · 2.16 KB

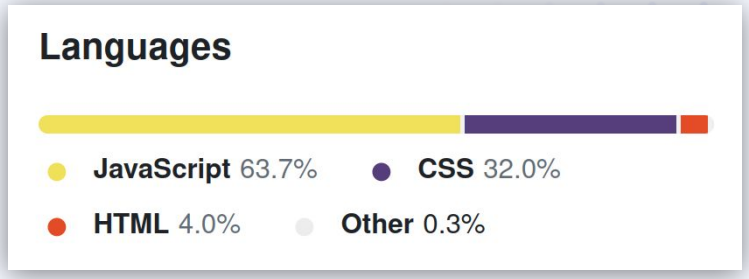
```
# ROT13 cipher

-,+[                         Read first character a
    -[                       Skip forward if charac
        >>++++[>++++++++<-]  Set up divisor (32) fo
                               (MEMORY LAYOUT: divi
        <+<-[                Set up dividend (x min
            >+>+>-[>>>]      Increase copy and rema
            <[[>+<-]>>+>]    Special case: move rem
            <<<<<-           Decrement dividend
        ]                    End division loop
    ]>>>[-]+                 End skip loop; zero fo
    >--[-[<->+++[-]]]<[         Zero that flag unle
        ++++++++++++<[       If flag then set up di
                               (MEMORY LAYOUT: zero
            >-[>+>>]         Reduce divisor; Normal
            >[+[<+>-]>+>>]   Special case: increase
            <<<<<-           Decrease dividend
        ]                    End division loop
```

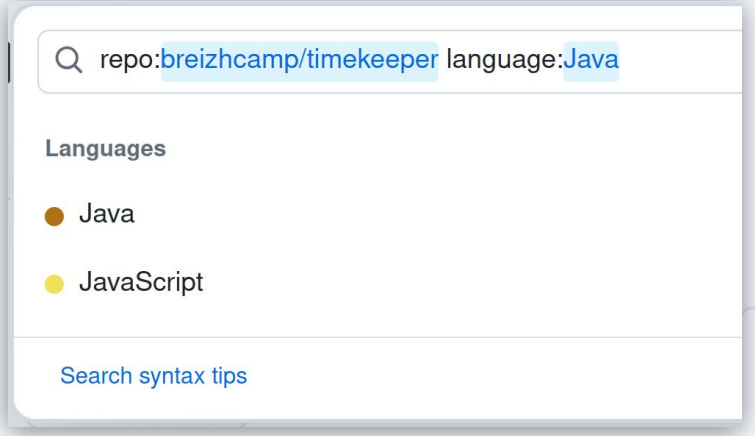
Linguist

- Cas d'usages
- Contraintes
- Alternatives
- Fonctionnement
- Linguist et vous
- Conclusion

Recherches et statistiques

- Everyone loves stats!
- Note : Le langage du dépôt est simplement le premier langage dans les statistiques.





Coloration syntaxique

```
/* first order of business is to set up memory allocation routines */
if (afs != NULL) {
    if (afs->malloc == NULL || afs->realloc == NULL || afs->free == NULL)
    {
        return NULL;
    }
} else {
    yajl_set_default_alloc_funcs(&afsBuffer);
    afs = &afsBuffer;
}

hand = (yajl_handle) YA_MALLOC(afs, sizeof(struct yajl_handle_t));

/* copy in pointers to allocation routines */
memcpy((void *) &(hand->alloc), (void *) afs, sizeof(yajl_alloc_funcs));

if (config != NULL) {
    allowComments = config->allowComments;
    validateUTF8 = config->checkUTF8;
}

hand->callbacks = callbacks;
hand->ctx = ctx;
hand->lexer = yajl_lex_alloc(&(hand->alloc), allowComments, validateUTF8);
hand->bytesConsumed = 0;
hand->decodeBuf = yajl_buf_alloc(&(hand->alloc));
yajl_bs_init(hand->stateStack, &(hand->alloc));

yajl_bs_push(hand->stateStack, yajl_state_start);
```

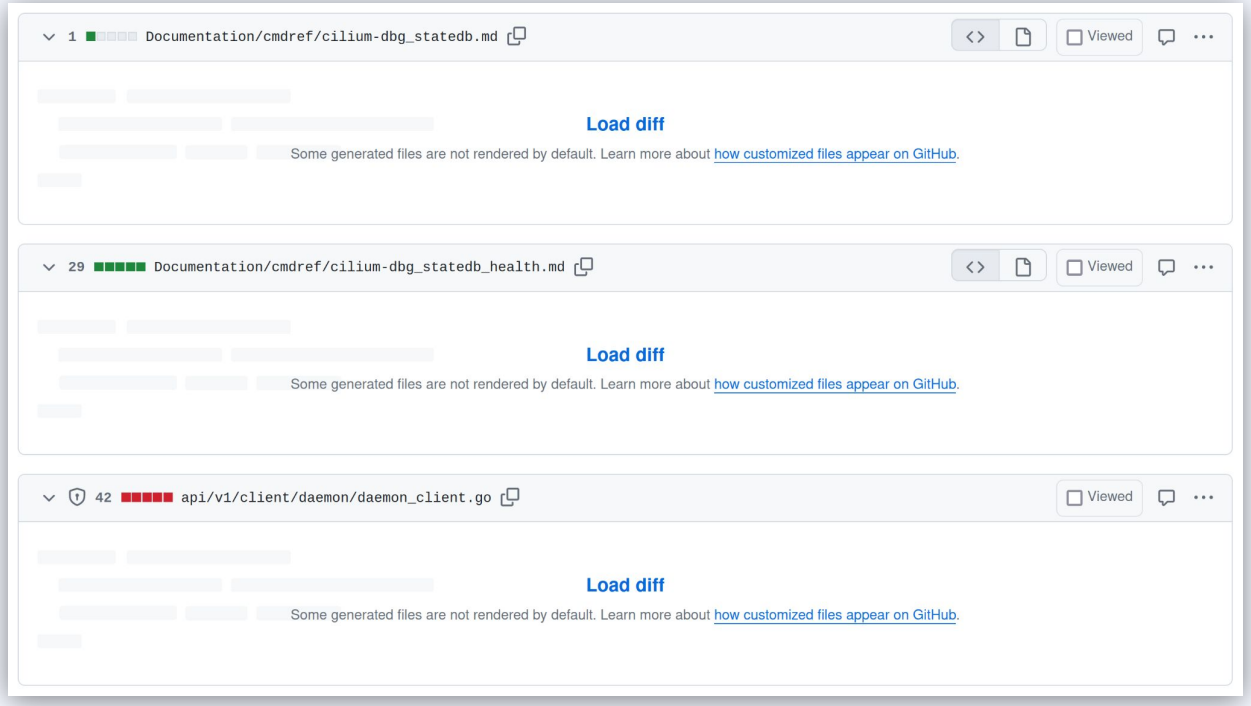

```
/* first order of business is to set up memory allocation routines */
if (afs != NULL) {
    if (afs->malloc == NULL || afs->realloc == NULL || afs->free == NULL)
    {
        return NULL;
    }
} else {
    yajl_set_default_alloc_funcs(&afsBuffer);
    afs = &afsBuffer;
}

hand = (yajl_handle) YA_MALLOC(afs, sizeof(struct yajl_handle_t));

/* copy in pointers to allocation routines */
memcpy((void *) &(hand->alloc), (void *) afs, sizeof(yajl_alloc_funcs));

if (config != NULL) {
    allowComments = config->allowComments;
    validateUTF8 = config->checkUTF8;
}

hand->callbacks = callbacks;
hand->ctx = ctx;
hand->lexer = yajl_lex_alloc(&(hand->alloc), allowComments, validateUTF8);
hand->bytesConsumed = 0;
hand->decodeBuf = yajl_buf_alloc(&(hand->alloc));
yajl_bs_init(hand->stateStack, &(hand->alloc));

yajl_bs_push(hand->stateStack, yajl_state_start);
```

Coloration syntaxique

```
/* first order of business is to set up memory allocation routines */
if (afs != NULL) {
    if (afs->malloc == NULL || afs->realloc == NULL || afs->free == NULL)
    {
        return NULL;
    }
} else {
    yajl_set_default_alloc_funcs(&afsBuffer);
    afs = &afsBuffer;
}

hand = (yajl_handle) YA_MALLOC(afs, sizeof(struct yajl_handle_t));

/* copy in pointers to allocation routines */
memcpy((void *) &(hand->alloc), (void *) afs, sizeof(yajl_alloc_funcs));

if (config != NULL) {
    allowComments = config->allowComments;
    validateUTF8 = config->checkUTF8;
}

hand->callbacks = callbacks;
hand->ctx = ctx;
hand->lexer = yajl_lex_alloc(&(hand->alloc), allowComments, validateUTF8);
hand->bytesConsumed = 0;
hand->decodeBuf = yajl_buf_alloc(&(hand->alloc));
yajl_bs_init(hand->stateStack, &(hand->alloc));
```

```
/* first order of business is to set up memory allocation routines */
if (afs != NULL) {
    if (afs->malloc == NULL || afs->realloc == NULL || afs->free == NULL)
    {
        return NULL;
    }
} else {
    yajl_set_default_alloc_funcs(&afsBuffer);
    afs = &afsBuffer;
}

hand = (yajl_handle) YA_MALLOC(afs, sizeof(struct yajl_handle_t));

/* copy in pointers to allocation routines */
memcpy((void *) &(hand->alloc), (void *) afs, sizeof(yajl_alloc_funcs));

if (config != NULL) {
    allowComments = config->allowComments;
    validateUTF8 = config->checkUTF8;
}

hand->callbacks = callbacks;
hand->ctx = ctx;
hand->lexer = yajl_lex_alloc(&(hand->alloc), allowComments, validateUTF8);
hand->bytesConsumed = 0;
hand->decodeBuf = yajl_buf_alloc(&(hand->alloc));
yajl_bs_init(hand->stateStack, &(hand->alloc));
```

Diffs intelligibles



Les fichiers générés ou tiers-parties ne sont pas chargés dans les diffs.

tmp / ??? / funs.php

Quizz 2/6

Code | Blame | 24 lines (20 loc) · 670 Bytes

```
1   <?hh
2   /**
3    * Copyright (c) 2014, Facebook, Inc.
4    * All rights reserved.
5    * This source code is licensed under the BSD-style
6    * LICENSE file in the root directory of this sourc
7    * of patent rights can be found in the PATENTS fil
8    */
9
10  class InvariantViolationException extends Exceptior
11
12  function invariant(mixed $test, string $message): v
13    if (!$test) {
14      invariant_violation($message);
15    }
16  }
17
18  function invariant_violation(string $message): voic
19    throw new InvariantViolationException($message);
```

Linguist

- Cas d'usages
- Contraintes
- Alternatives
- Fonctionnement
- Linguist et vous
- Conclusion

Une librairie “pragmatique”

- GitHub et Gitlab sont peu personnalisables
- Force à faire des choix par défaut consensuels
- Doit correspondre aux habitudes, plutôt qu’aux standards :
  - Extensions de Prolog sont .pl et .pro, mais certains utilisent .prolog
  - Pas de commentaires en JSON mais utilisés quand même

Maintenue par des profanes

- 730+ langages supportés
- Une majorité de langages spécialisés ou peu connus :
  - SuperCollider : synthèse et traitement audio
  - Zeek : politiques de monitoring pour NIDS
  - Luau : scripts pour la plateforme de jeu Roblox
  - SmPL : patches sémantiques pour code C
- Doit être maintenable même sans expertise sur les langages

Code Compatible MIT

- GitHub et GitLab vendent une version entreprise donc les licenses copyleft (ex. GPL) sont exclues
- Code propriétaire aussi exclu (duh)
- Le code sans license doit être traité comme propriétaire

Économe et rapide

- Linguist traite tous les fichiers de chaque commit
- Pour 420+ millions de dépôts sur GitHub (mais maj. inactifs)
- Tâche en background, avec priorité faible par rapport aux PRs ou recherches
- Ressources limitées, moins important que l'indexage ou git

tmp / ??? / **JSONKit.h**

Quizz 3/6

Code | Blame | 189 lines (156 loc) · 9.11 KB

```
1   #include <stddef.h>
2   #include <stdint.h>
3   #include <limits.h>
4   #include <TargetConditionals.h>
5   #include <AvailabilityMacros.h>
6
7   #import <Foundation/NSArray.h>
8   #import <Foundation/NSData.h>
9   #import <Foundation/NSDictionary.h>
10  #import <Foundation/NSError.h>
11  #import <Foundation/NSObjCRuntime.h>
12  #import <Foundation/NSString.h>
13
14  // For Mac OS X < 10.5.
15  #ifndef   NSINTEGER_DEFINED
16  #define   NSINTEGER_DEFINED
17  #if       defined(__LP64__) || defined(NS_BUILD_32_
18  typedef long           NSInteger;
19  typedef unsigned long  NSUInteger;
20  #define NSIntegerMin   LONG_MIN
```

Linguist

- Cas d'usages
- Contraintes
- Alternatives
- Fonctionnement
- Linguist et vous
- Conclusion

Extensions de fichier

- S’il fallait coder une détection en 10min, vous utiliseriez… les extensions!
- Approches prise par de nombreux éditeurs de code
- Requiert souvent des réglages manuels

Réseaux de neurones

- Approche de ex. BigCode's StarEncoder
  - Encoder-only bi-directionally self-attentive transformer
  - ~80 langages reconnus
- Non-existant in 2011
- Problème légal à l'époque :
  - Difficile de trouver assez de samples compatibles MIT
  - Les juristes de GitHub le demandaient même pour les données d'entraînement
  - C'était avant Copilot ;)
- Aujourd'hui, toujours difficile de trouver des samples annotés

Grammaires de coloration

- Utiliser les grammaires de coloration syntaxiques
  - Approche de ex. Pygments
- Délègue le problème d'expertise aux grammaires tiers-parties
- De nombreux langages n'ont pas de grammaire compatible et disponible
- Faire le parsing *n* fois tend à être lent

tmp / ??? / Adding.NCL.Language.s01e01.srt

Code | Blame | 240 lines (183 loc) · 5.75

Quizz 4/6

```
1
00:00:01,250 --> 00:00:03,740
Adding NCL language.

2
00:00:04,600 --> 00:00:08,730
Thanks for the pull request! Do you know if these f

3
00:00:09,800 --> 00:00:13,700
Those are poorly-named documentation files for NCL

4
00:00:14,560 --> 00:00:17,200
- What's better?
- This is better.

5
00:00:18,500 --> 00:00:23,000
```

Linguist

- Cas d'usages
- Contraintes
- Alternatives
- Fonctionnement
- Linguist et vous
- Conclusion

Combinaison de stratégies

- Linguist combine différentes stratégies
- Commence avec tous les langages en candidats
- Filtre au fur et à mesure

```
STRATEGIES = [
  Linguist::Strategy::Modeline,
  Linguist::Strategy::Filename,
  Linguist::Shebang,
  Linguist::Strategy::Extension,
  Linguist::Strategy::XML,
  Linguist::Strategy::Manpage,
  Linguist::Heuristics,
  Linguist::Classifier
]
```

Premier filtre : extensions

- Première détection sur les extensions…
  - .java, .js, .cpp, etc.
- …et les noms de fichiers
  - Makefile, Jenkinsfile, .bashrc, etc.
- Pas suffisant pour beaucoup d’extensions
  - .h: C, C++, Objective-C
  - .v: Coq, Verilog, V
  - + environ 130 autres extensions avec conflits

Classifieur bayésien naïf

- Classifieur probabiliste
- Très simple à saisir et implémenter
- Requiert peu de samples pour avoir des résultats corrects

Classifieur bayésien naïf

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)}$$

en.wikipedia.org/wiki/Naive_Bayes_classifier

Classifieur bayésien naïf

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)}$$

⚠️ Si `extern` et `elif` sont indépendants conditionnellement à `Zig`

$$P(Zig|extern, elif) = \frac{P(Zig)P(extern|Zig)P(elif|Zig)}{P(extern, elif)}$$

en.wikipedia.org/wiki/Naive_Bayes_classifier

Classifieur bayésien naïf

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)}$$

$$P(Zig|extern, elif) = \frac{P(Zig)P(extern|Zig)P(elif|Zig)}{P(extern, elif)}$$

$$P(Java|extern, elif) = \frac{P(Java)P(extern|Java)P(elif|Java)}{P(extern, elif)}$$

en.wikipedia.org/wiki/Naive_Bayes_classifier

Classifieur bayésien naïf

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)}$$

$$P(Zig|extern, elif) = \frac{P(Zig)P(extern|Zig)P(elif|Zig)}{P(extern, elif)}$$

$$P(Java|extern, elif) = \frac{P(Java)P(extern|Java)P(elif|Java)}{P(extern, elif)}$$

$$max(P(Zig|extern, elif), P(Java|extern, elif))$$

en.wikipedia.org/wiki/Naive_Bayes_classifier

Classifieur bayésien naïf

- Reste loin d’être parfait
- Surtout que Linguist manque parfois de samples divers
  - Dur de s’en rendre compte sans être un “expert” du langage
- Cherche des moyens fiables de gérer les cas plus complexes

Emacs & Vim Modelines

- Options spécifiques à ces éditeurs
- Via des commandes dans les commentaires d'en-tête
- Dont le langage !

Vim

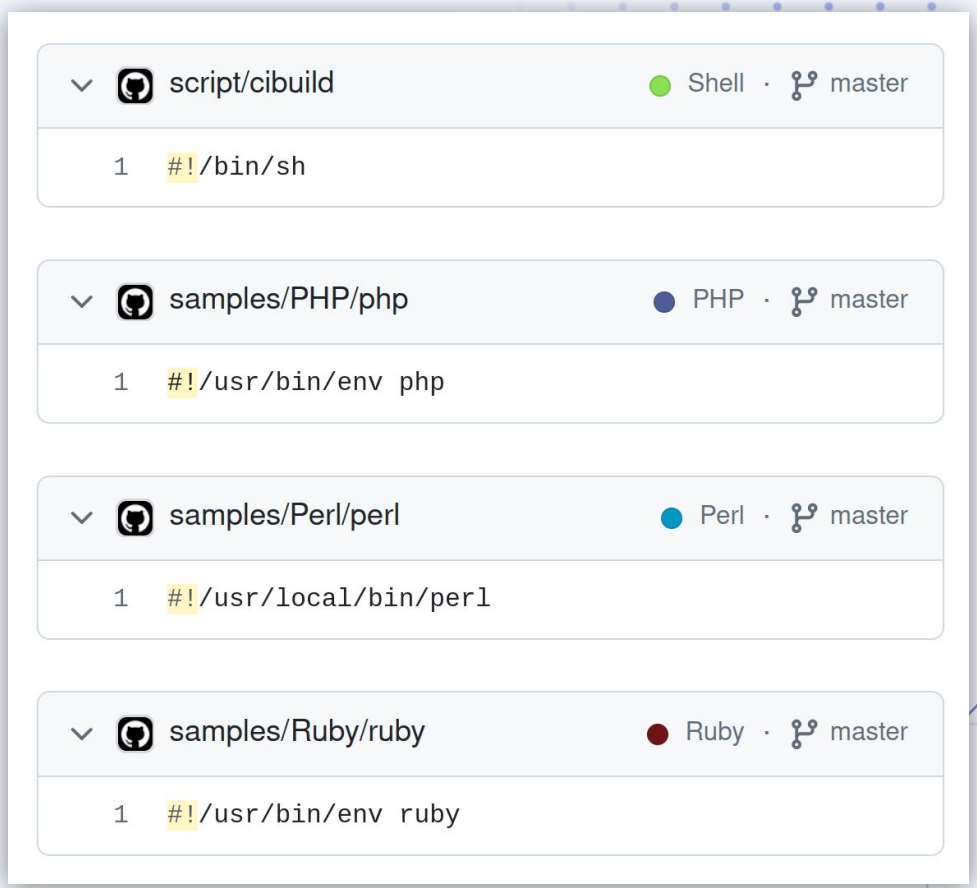
```
# Some examples of various styles:
vim: syntax=java
vim: set syntax=ruby:
vim: set filetype=prolog:
vim: set ft=cpp:
```

Emacs

```
-*- mode: php; -*-
-*- c++ -*-
```

Shebangs

- Typiquement pour les langages de script
- Indique l'interpréteur à utiliser
- #!, sharp-bang, she-bang



Expert Rules

- On peut généraliser les approches Modelines & Shebang
- Regular expressions ✨
- Limited to specific, known cases

```
- extensions: ['.pl']
  rules:
  - language: Prolog
    pattern: '^[^#]*:-'
  - language: Perl
    and:
      - negative_pattern: '^\s*use\s+v6\b'
      - pattern:
        - '\buse\s+(?:strict\b|v?5\b)'
        - '^\s*use\s+(?:constant|overload)\b'
  - language: Raku
    named_pattern: '^\s*(?:use\s+v6\b|\bmodule\b)'
```

Putting it all together

- Linguist combine les stratégies précédentes
- Filtrent les langages candidats
- Plus précis / moins coûteux en premier

```
STRATEGIES = [
  Linguist::Strategy::Modeline,
  Linguist::Strategy::Filename,
  Linguist::Shebang,
  Linguist::Strategy::Extension,
  Linguist::Strategy::XML,
  Linguist::Strategy::Manpage,
  Linguist::Heuristics,
  Linguist::Classifier
]
```

Types de fichiers

- *Detectable* : Inclus dans les stats (langages programming et markup)
- *Generated* : Exclus des stats, caché dans les diffs
- *Vendored* : Exclus des stats
- *Documentation* : Exclus des stats

- Basé sur des règles assez simples :
  - `node_modules/` => vendored
  - `jquery-3.7.1.js` => vendored
  - `.css`, single line, 1k characters => minified, considered generated
  - `.js`, first line is `"// Generated by "` => generated (CoffeeScript)

tmp / ??? / **htmlify.pl**

Quizz 5/6

Code | Blame | 629 lines (574 loc) · 18.8 KB

```
1   use v6;
2
3   # This script isn't in bin/ because it's not meant
4
5   BEGIN say 'Initializing ...';
6
7   use Pod::To::HTML;
8   use URI::Escape;
9   use lib 'lib';
10  use Perl6::TypeGraph;
11  use Perl6::TypeGraph::Viz;
12  use Perl6::Documentable::Registry;
13
14  my $*DEBUG = False;
15
16  my $tg;
17  my %methods-by-type;
18  my $footer = footer-html;
19  my $head   = q[
```

Linguist

- Cas d'usages
- Contraintes
- Alternatives
- Fonctionnement
- Linguist et vous
- Conclusion

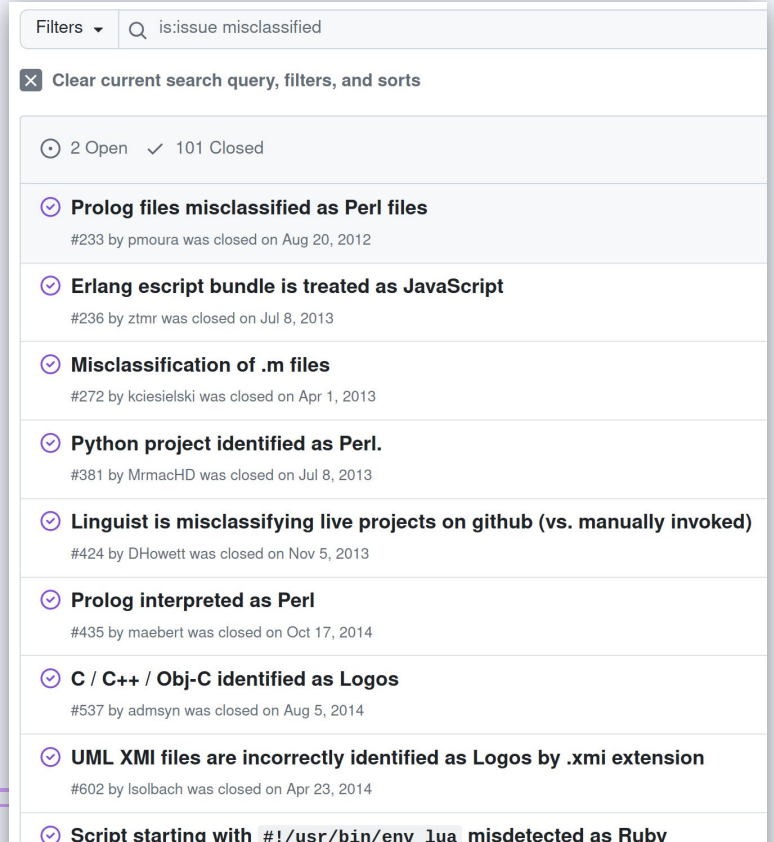
Ajouter un langage

- Prérequis pour chaque extension :
  - Utilisée dans des centaines de dépôts GitHub publics
  - Au moins un exemple réaliste par extension

- Optionnels :
  - Grammaires
  - Interpréteur
  - Couleur

```
Asymptote:
  type: programming
  color: "#ff0000"
  extensions:
  - ".asy"
  interpreters:
  - asy
  tm_scope: source.c++
  ace_mode: c_cpp
  codemirror_mode: clike
  codemirror_mime_type: text/x-kotlin
  language_id: 591605007
```

Corriger la détection



- Deux options :
  - Ajouter des samples (`samples/LANG/`)
  - Ajouter ou corriger une expert rule (`heuristics.yml`)

Corriger la coloration

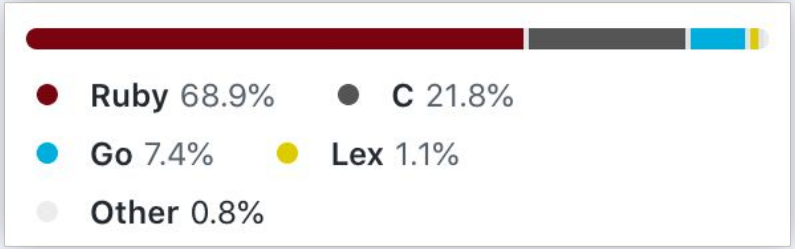
- GitHub utilise les grammaires tiers-parties TextMate (Sublime Text, Atom)

- La plupart des grammaires sont inactives
  - Patient, you must be!
  - Forking, you must avoid!

```
# Comment lines
comment:
        name:  "comment.line.number-sign.ssh-config"
        begin: "#"
        end:   "$"
        beginCaptures:
                0: name: "punctuation.definition.comment.ssh-config"
```

Changer la couleur d’un langage

- Une des demandes les plus courantes
- Les discussions les plus “animées” 🙄
- Prérequis :
  - Forme de consensus dans la communauté
  - Doit être discernable



Everyone wants blue!

lildude.github.io/linguist-free-colours

Everyone wants blue!

lildude.github.io/linguist-free-colours

Contourner Linguist

master

tmp / .gitattributes

14 lines (9 loc) · 388 Bytes

```
1   # Reconnaitre les fichiers .rb comme Java
2   *.rb linguist-language=Java
3
4   # Marquer les fichiers d'un dossier comme vendored
5   special-vendored-path/* linguist-vendored
6
7   # Ne pas marquer un dossier et ses sous-dossiers comme étant de la doc
8   docs/** linguist-documentation=false
9
10  # Compter les fichiers .sql dans les statistiques
11  *.sql linguist-detectable=true
12
```

tmp / ??? / **Smallstep.v**

Quizz 6/6

Code | Blame | 637 lines (526 loc) · 13.5 KB

```
1   Require Export Imp.
2   Require Export Relations.
3
4   Fixpoint eval (t : tm) : nat :=
5      match t with
6         | tm_const n => n
7         | tm_plus a1 a2 => eval a1 + eval a2
8      end.
9
10  Example test_step_1 :
11        tm_plus
12          (tm_plus (tm_const 0) (tm_const 3))
13          (tm_plus (tm_const 2) (tm_const 4))
14        =>
15        tm_plus
16          (tm_const (plus 0 3))
17          (tm_plus (tm_const 2) (tm_const 4)).
18  Proof.
19    apply ST_Plus1. apply ST_PlusConstConst. Qed.
```

Linguist

- Cas d'usages
- Contraintes
- Alternatives
- Fonctionnement
- Linguist et vous
- Conclusion

Conclusion

- Combinaison d’approches simples de détection
- Beaucoup de langages existants donc beaucoup à améliorer
- Simple (Ruby aussi) => facile pour contribuer

Merci !

linkedin.com/in/pchaigno

twitter.com/pchaigno

CREDITS: This presentation template was created by **Slidesgo**, and includes icons by **Flaticon**, and infographics & images by **Freepik**