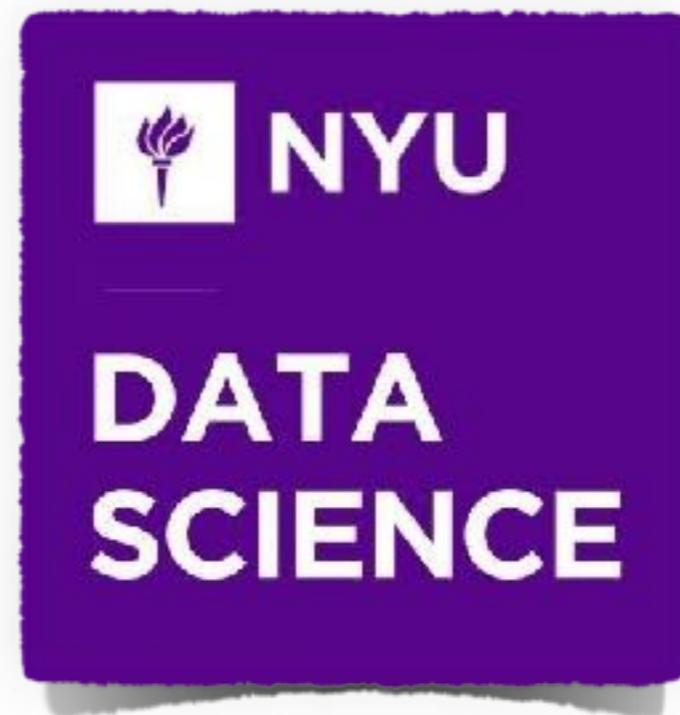


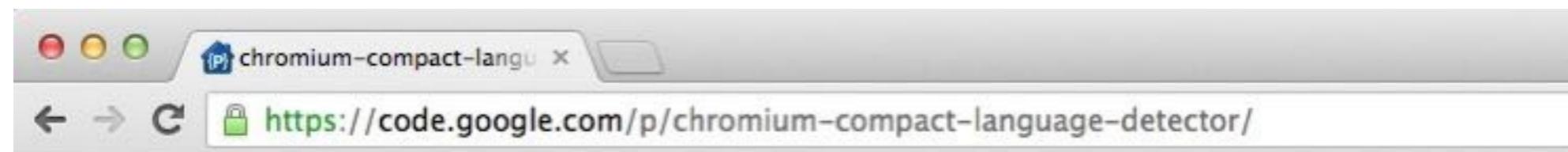
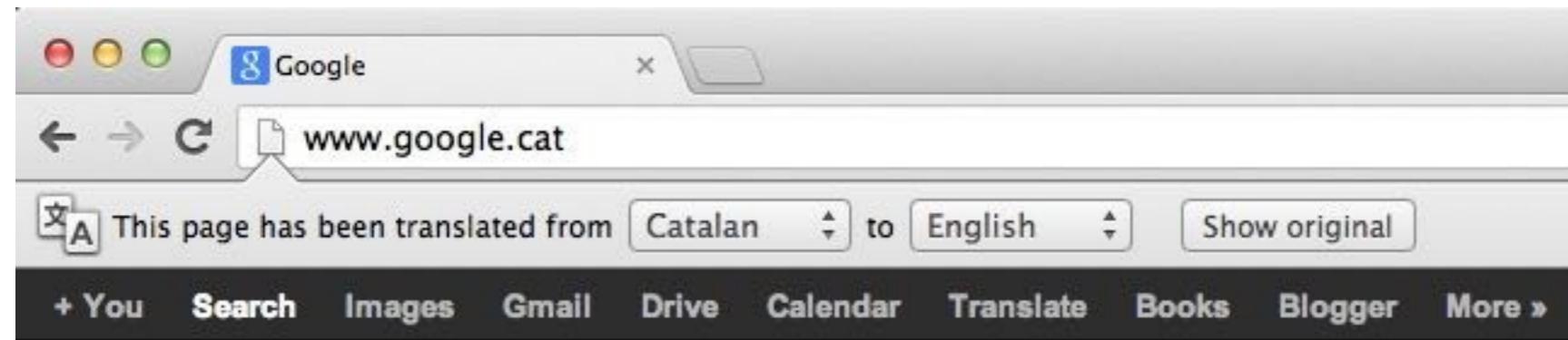
<https://bmtgoncalves.github.io/TorinoCourse/>

Lecture III - Collaborative Platforms

Bruno Gonçalves
www.bgoncalves.com



Chromium Compact Language Detector



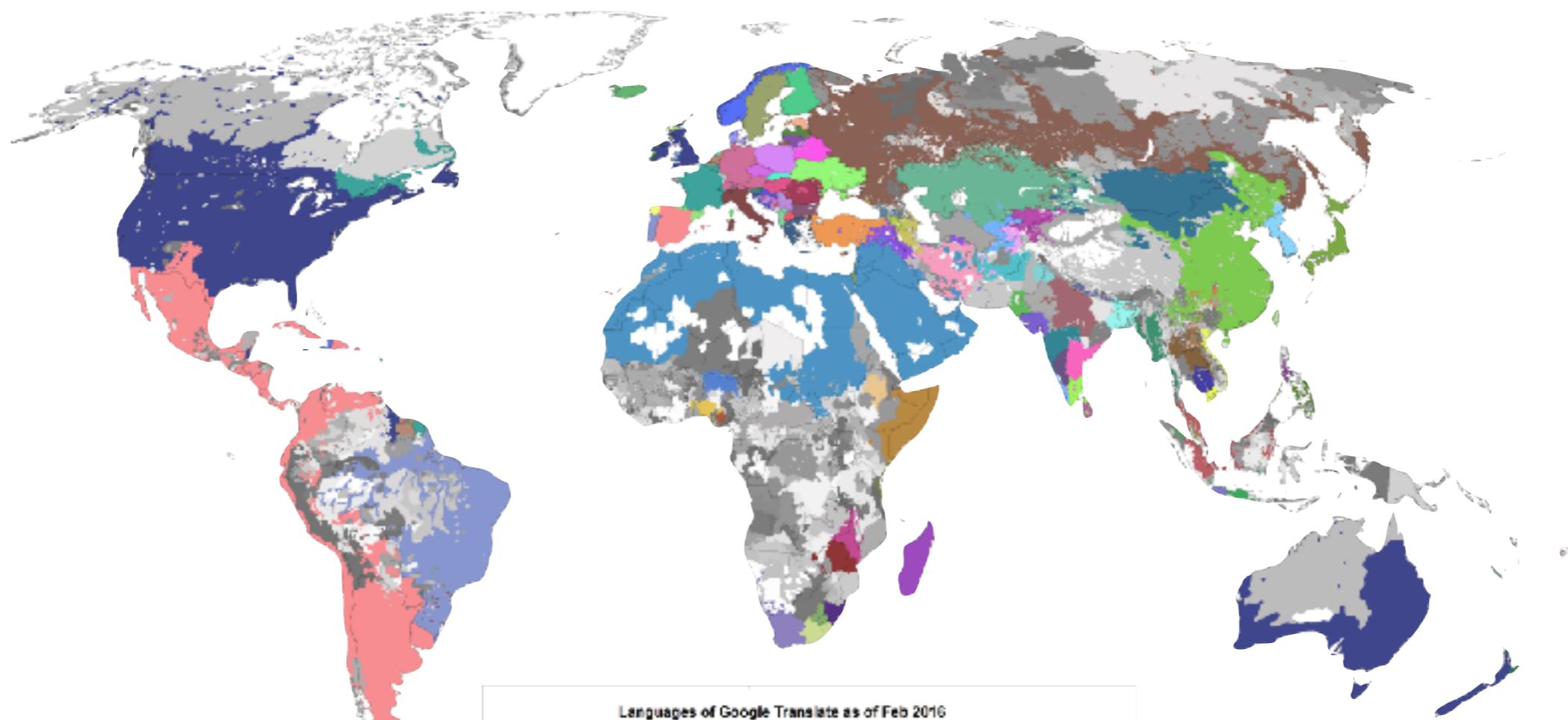
 **chromium-compact-language-detector**

C++ library and Python bindings for detecting language from UTF8 text, extracted from the Chromium browser

[Project Home](#) [Downloads](#) [Wiki](#) [Issues](#) [Source](#)

[Summary](#) [People](#)

Chromium Compact Language Detector



Languages of Google Translate as of Feb 2016			
Languages not present in Google Translate	Catalan	Esperanto	Sorani
Pashto	Chinese (Simpl.)	Ibo	Maly (Universel)
Afrikaans	Corsican	Icelandic	Maltese
Albanian	Croatian	Irish	Maori
Armenian	Czech	Italian	Marathi
Arabic	Danish	Japanese	Roma
Amharic	Dutch	Javanesse	Hmong
Spanish	Basque	Javanese	Mosuo
Armenian	English	Hebrew	Nepali
Swahili	Bulgarian	Kannada	Norwegian
German	Finnish	Kazakh	Punjabi
Azerbaijani	French	Kirundi	Romanian
Bosnian	Georgian	Kirghiz	Russian
Croatian	Hindi	Korean	Sinhalese
Burkinabé	Bosnian-Cyrillic	Kurdish	Sorbian
Bengali	Croatian	Lao	Turkish
Czech	Dari	Lithuanian	Ukrainian
Burkina Faso	Egyptian Arabic	Luxembourgish	Vietnamese
Portuguese	Farsi	Maltese	Welsh
Darijan	Galician	Ukrainian	Xhosa
Burmese	Hausa	Magyar	Zulu
Belarusian	Hawaiian	Malagasy	
Polish	Hindi	Maltese	

cld

<https://github.com/mikemccand/chromium-compact-language-detector>

- C package with C++ and Python bindings
- Particularly messy to install, but very fast and easy to use
- `cld.detect(text)` returns a list of possible languages and their likelihood
- Very conservative. If it thinks that it can't give a reliable answer, it will return "**Unknown**"
- **Version 2** is better suited to longer pieces of text and it allows for the possibility of different sections being in different languages.

```
import cld

text_it = "Wales lancia la Wikipedia delle news. Contro il fake in campo anche Google"
text_en = "Cassini Spacecraft Re-Establishes Contact After 'Dive' Between Saturn And Its Rings"

lang_it = cld.detect(text_it)
lang_en = cld.detect(text_en)

print(text_it, "is in", lang_it)
print(text_en, "is in", lang_en)
```

Signal By Language

PLoS One 8, E61981 (2013)



Signal By Language

PLoS One 8, E61981 (2013)

- Italian
- English
- Spanish
- Portuguese
- Other

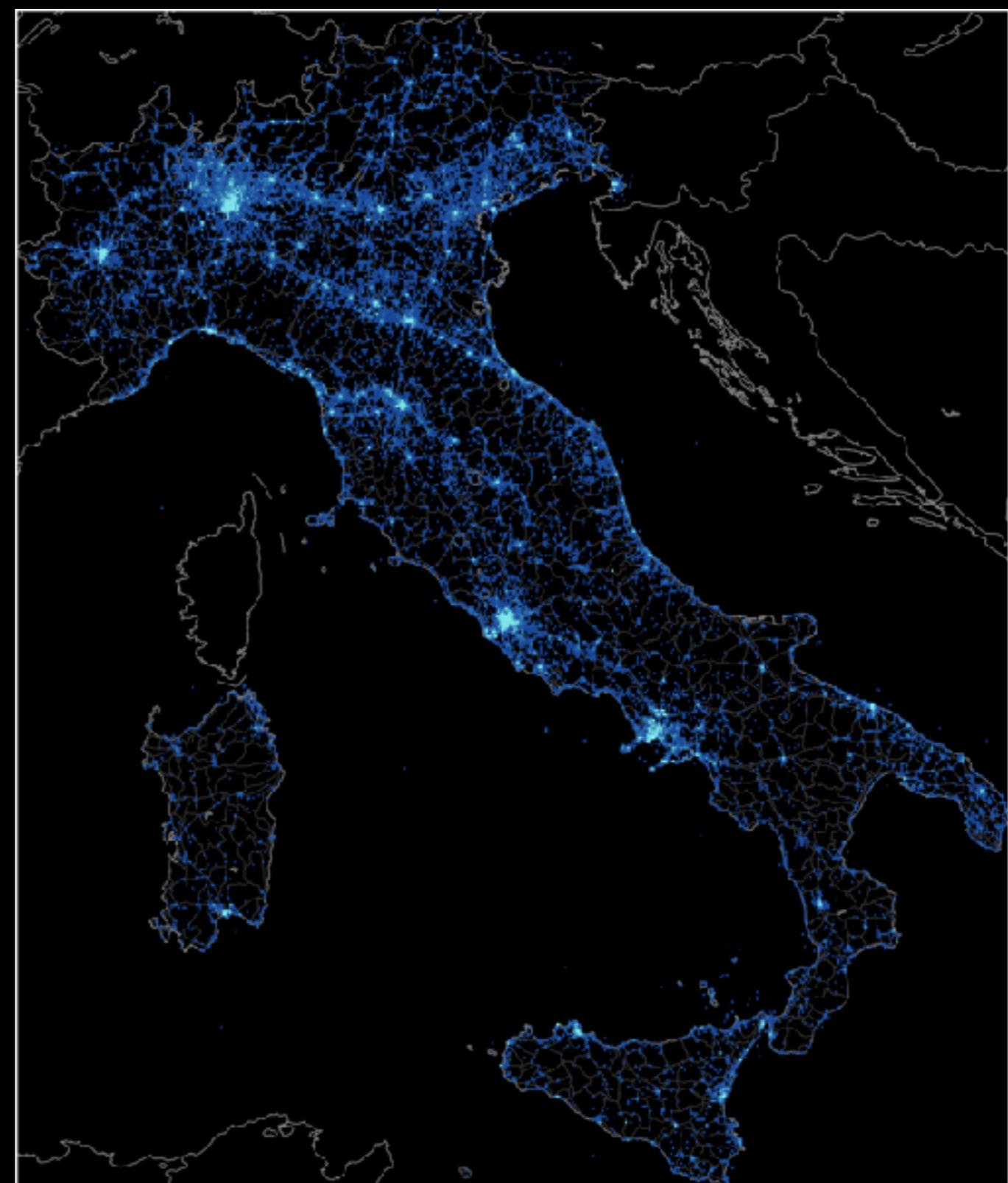
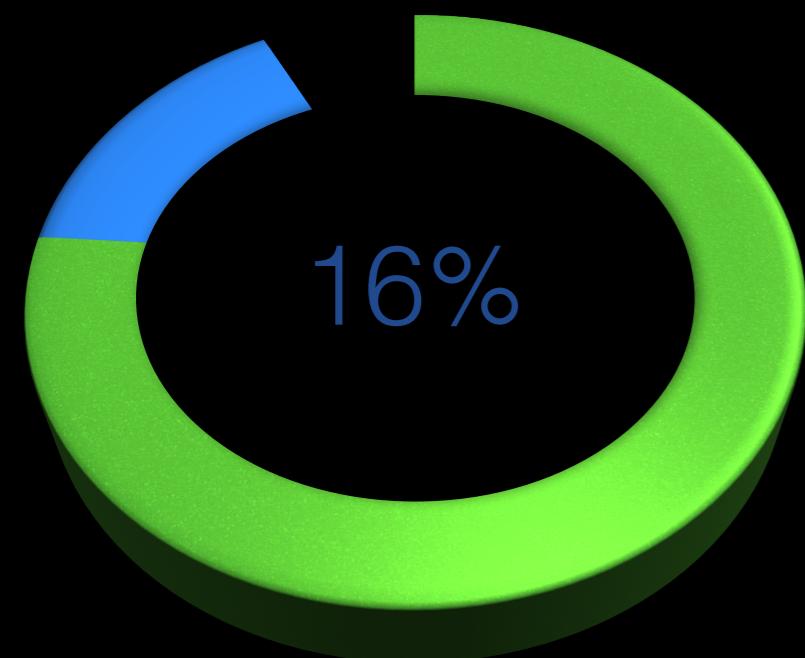
76%



Signal By Language

PLoS One 8, E61981 (2013)

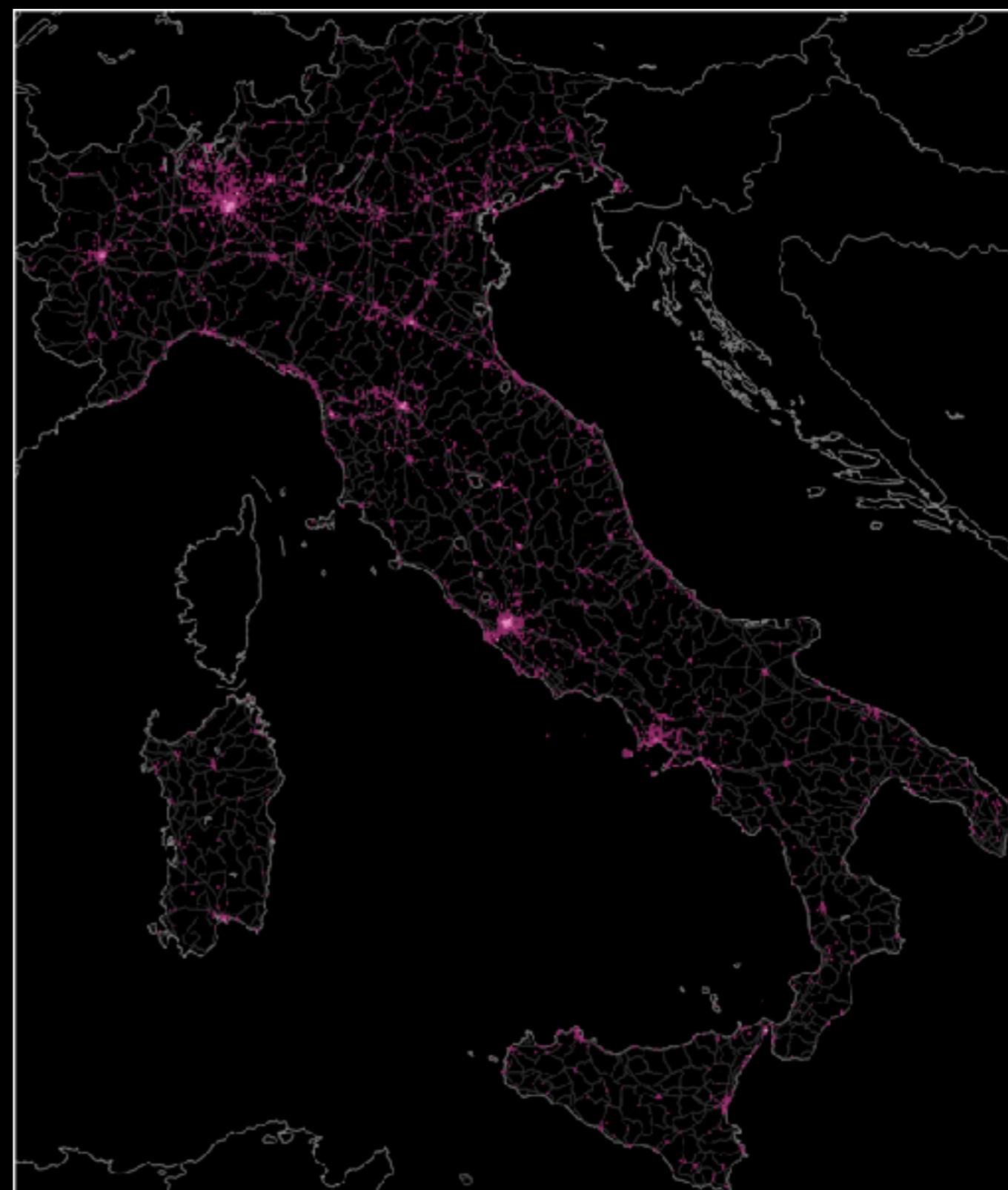
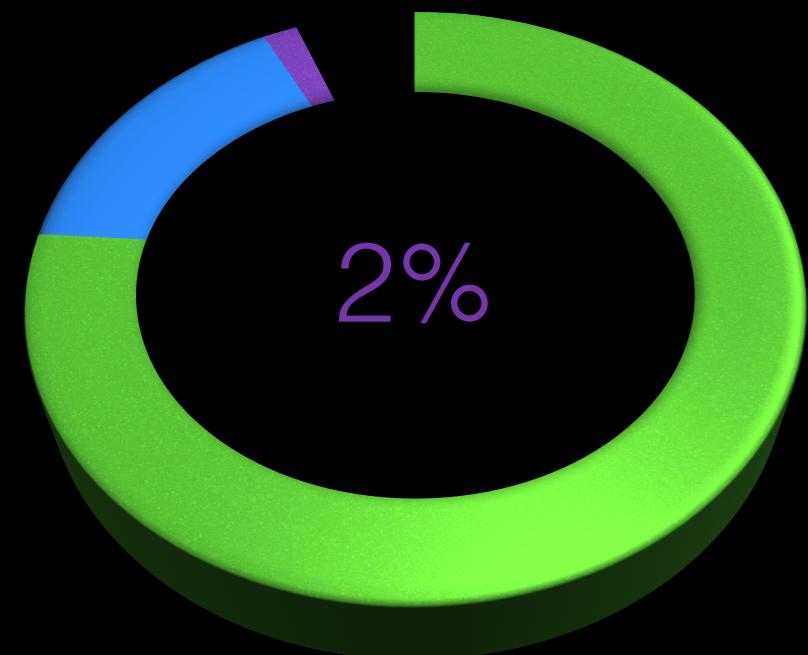
- Italian
- English
- Spanish
- Portuguese
- Other



Signal By Language

PLoS One 8, E61981 (2013)

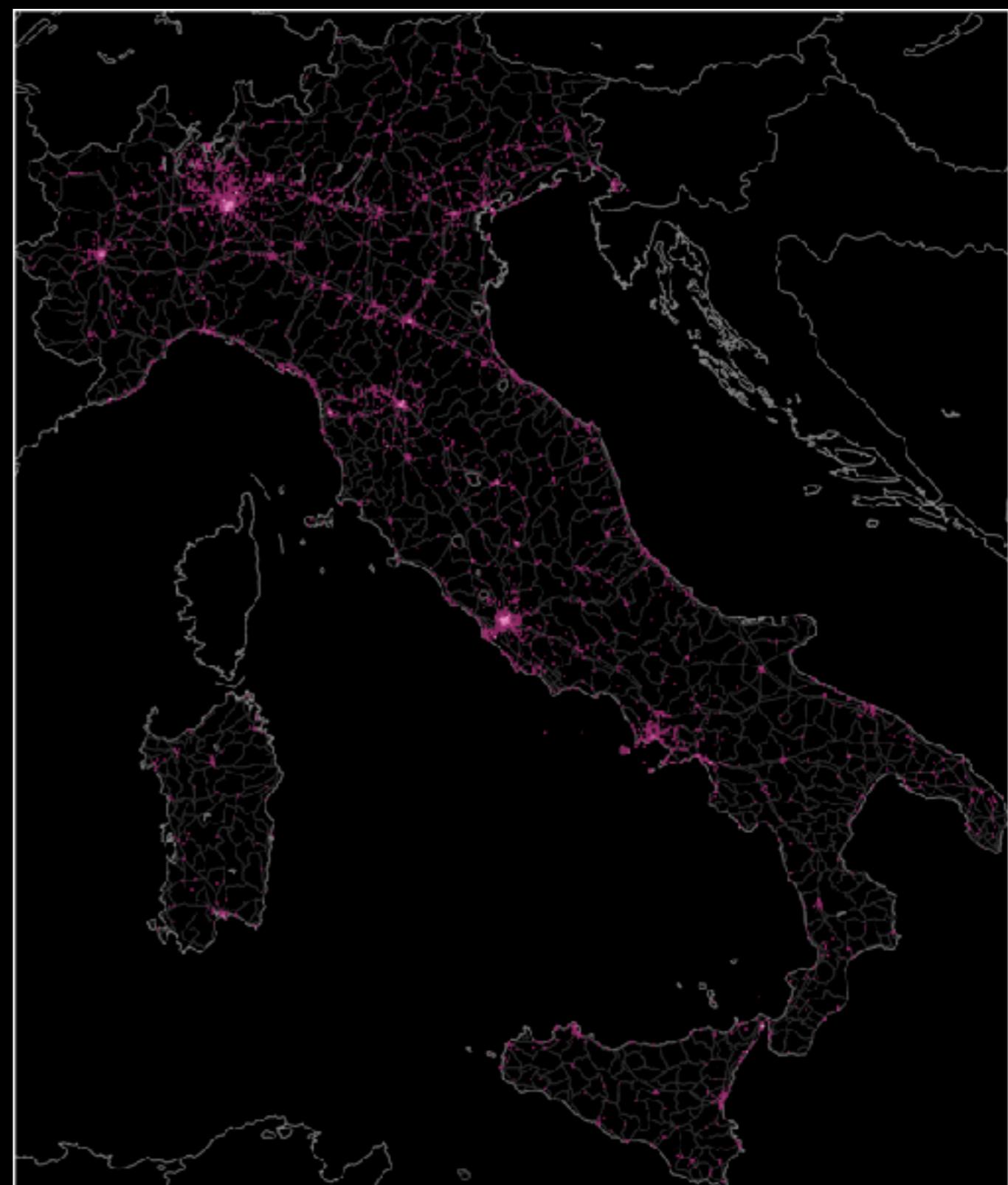
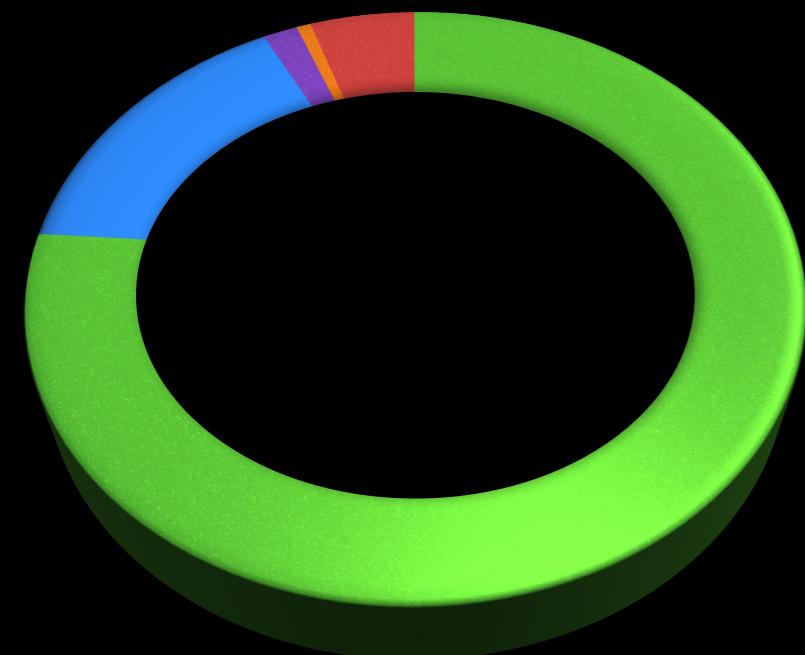
- Italian
- English
- Spanish
- Portuguese
- Other



Signal By Language

PLoS One 8, E61981 (2013)

- Italian
- English
- Spanish
- Portuguese
- Other



langid

<https://github.com/saffsd/langid.py>

- Entirely written in Python
- Supports **97** languages
- Has the option of retrieving the likelihood for every known language.
- **.classify(text)** - Returns the most likely language and it's likelihood
- **.rank(text)** - Returns the results for all known languages.

```
import langid

text_it = "Wales lancia la Wikipedia delle news. Contro il fake in campo anche Google"
text_en = "Cassini Spacecraft Re-Establishes Contact After 'Dive' Between Saturn And Its Rings"

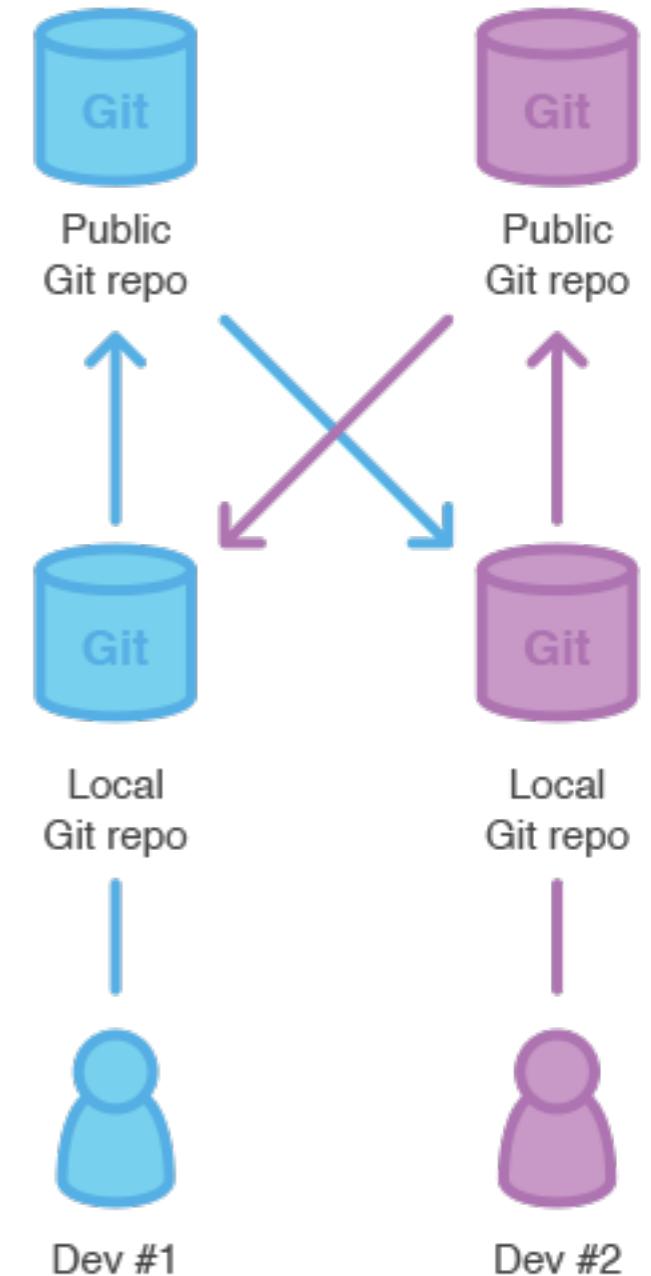
lang_it = langid.classify(text_it)
lang_en = langid.classify(text_en)

print(text_it, "is in", lang_it)
print(text_en, "is in", lang_en)
```

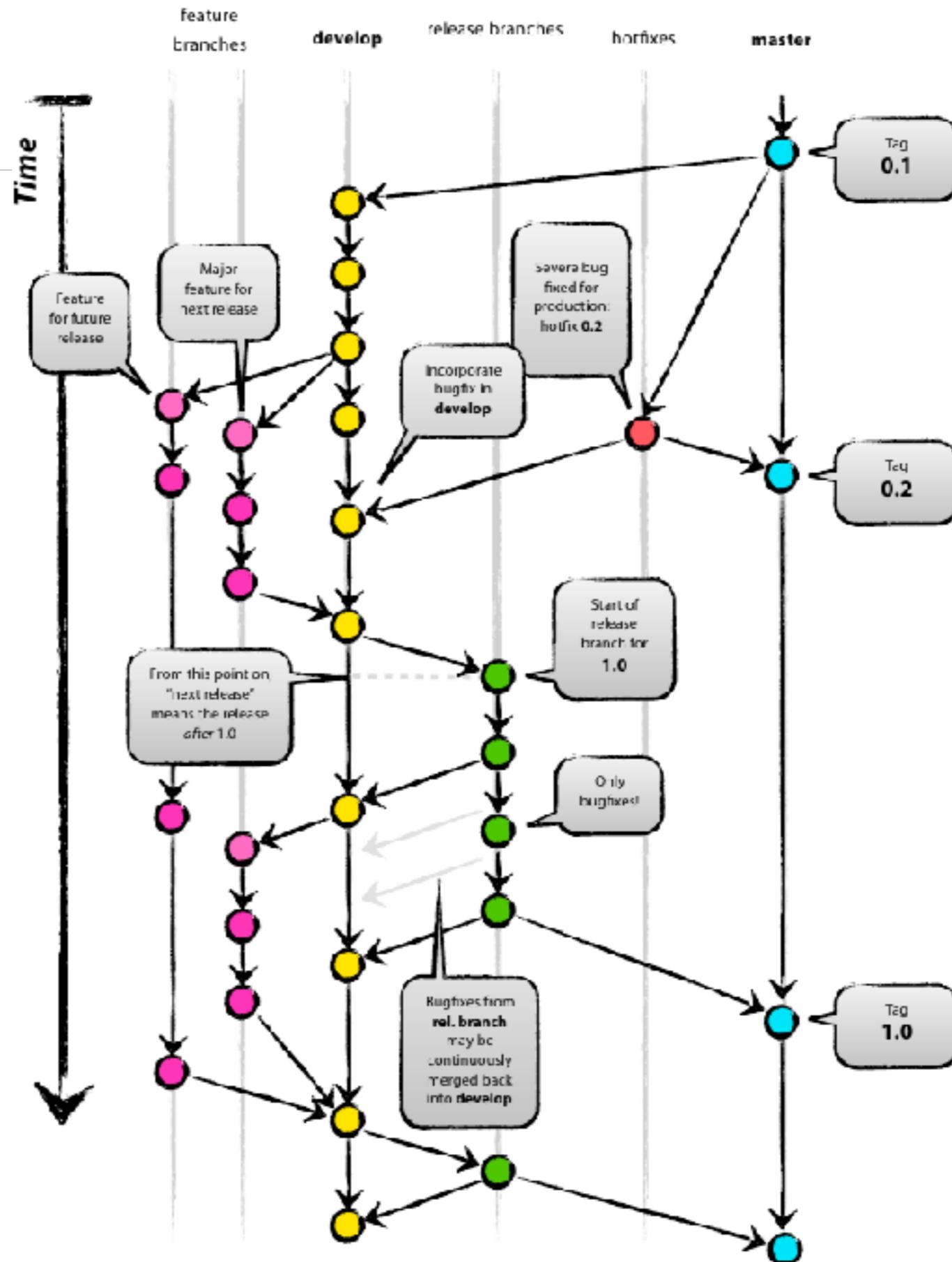
GitHub



- Created by Linus Torvalds in 2005
- Distributed Source Code Management System
- “the stupid content tracker”
- Extremely efficient and popular
- Self hosted in **3** days
- Built to handle linux kernel development
- Designed for remote (offline) collaboration



Git Branches





torvalds / linux

Star

18,834

Fork

7,500

Linux kernel source tree

495,037 commits

1 branch

402 releases

4,528 contributors



branch: master ▾

linux / +



Merge branch 'leds-fixes-for-3.19' of git://git.kernel.org/pub/scm/li...



torvalds authored 21 hours ago

latest commit 188c901941

	Documentation	Merge git://git.kernel.org/pub/scm/linux/kernel/git/nab/target-pending	2 days ago
	arch	Merge tag 'stable/for-linus-3.19-rc4-tag' of git://git.kernel.org/pub...	a day ago
	block	Revert "blk-mq: Micro-optimize bt_get()"	a month ago
	crypto	crypto: af_alg - fix backlog handling	23 days ago
	drivers	Merge branch 'leds-fixes-for-3.19' of git://git.kernel.org/pub/scm/li...	21 hours ago
	firmware	kbuild: remove obj-n and lib-n handling	3 months ago
	fs	Merge branch 'sched-urgent-for-linus' of git://git.kernel.org/pub/scm...	3 days ago
	include	Merge tag 'for-linus' of git://git.kernel.org/pub/scm/linux/kernel/gi...	21 hours ago
	init	init: fix read-write root mount	28 days ago
	ipc	Merge branch 'for-linus' of git://git.kernel.org/pub/scm/linux/kernel...	29 days ago
	kernel	Merge branch 'sched-urgent-for-linus' of git://git.kernel.org/pub/scm...	3 days ago
	lib	Merge tag 'for_linus-3.19-rc4' of git://git.kernel.org/pub/scm/linux/...	5 days ago
	mm	mm: mmu_gather: use tlb->end != 0 only for TLB invalidation	2 days ago

Code

Pull Requests

61

Pulse

Graphs

HTTPS clone URL

<https://github.com>You can clone with [HTTPS](#) or [Subversion](#).

Download ZIP

GitHub

- Most popular Git repository hosting service
- Web based
- Launched April 2008
- Adds functionality on top of Git
- “Gists” - Version controlled paste-bins
- Users create accounts and are able to comment, fork, clone, etc...

GitHub

GitHub Developer | GitHub Dev x Bruno

Secure https://developer.github.com

GitHub Developer API Blog Early Access Support Search...

Build your app on the GitHub platform

Not sure where to start? We've put together some handy guides and libraries you can use to start building.

GitHub

Personal Access Tokens

GitHub, Inc. [US] https://github.com/settings/tokens Bruno

Search GitHub Pull requests Issues Gist

Personal settings

Profile

Account

Emails

Notifications

Billing

SSH and GPG keys

Security

Blocked users

Repositories

Organizations

Saved replies

Authorized applications

Installed integrations

Developer settings

OAuth applications

Integrations

Personal access tokens

Personal access tokens

Generate new token Revoke all

Tokens you have generated that can be used to access the GitHub API.

Data Mining — admin:gpg_key, admin:org, admin:org_hook, admin:public_key, admin:repo_hook, delete_repo, gist, notifications, repo, user

Last used within the last day Edit Delete

② Personal access tokens function like ordinary OAuth access tokens. They can be used instead of a password for Git over HTTPS, or can be used to authenticate to the API over Basic Authentication.

GitHub

New personal access token

GitHub, Inc. [US] https://github.com/settings/tokens/new Bruno

Search GitHub Pull requests Issues Gist

Personal settings

Profile Account Emails Notifications Billing SSH and GPG keys Security Blocked users Repositories Organizations Saved replies Authorized applications Installed integrations Developer settings OAuth applications Integrations Personal access tokens

New personal access token

Personal access tokens function like ordinary OAuth access tokens. They can be used instead of a password for Git over HTTPS, or can be used to [authenticate to the API over Basic Authentication](#).

Token description

Testing...

What's this token for?

Select scopes

Scopes define the access for personal tokens. [Read more about OAuth scopes](#).

<input checked="" type="checkbox"/> repo	Full control of private repositories
<input type="checkbox"/> repo:status	Access commit status
<input type="checkbox"/> repo_deployment	Access deployment status
<input type="checkbox"/> public_repo	Access public repositories
<input checked="" type="checkbox"/> admin:org	Full control of orgs and teams
<input type="checkbox"/> write:org	Read and write org and team membership
<input type="checkbox"/> read:org	Read org and team membership
<input checked="" type="checkbox"/> admin:public_key	Full control of user public keys
<input type="checkbox"/> write:public_key	Write user public keys
<input type="checkbox"/> read:public_key	Read user public keys
<input checked="" type="checkbox"/> admin:repo_hook	Full control of repository hooks
<input type="checkbox"/> write:repo_hook	Write repository hooks
<input type="checkbox"/> read:repo_hook	Read repository hooks
<input checked="" type="checkbox"/> admin:org_hook	Full control of organization hooks

GitHub

Personal Access Tokens

GitHub, Inc. [US] https://github.com/settings/tokens

Search GitHub Pull requests Issues Gist

Personal settings Profile Account Emails Notifications Billing SSH and GPG keys Security Blocked users Repositories Organizations Saved replies Authorized applications Installed integrations Developer settings OAuth applications Integrations Personal access tokens

Personal access tokens

Generate new token Revoke all

Tokens you have generated that can be used to access the GitHub API.

Make sure to copy your new personal access token now. You won't be able to see it again!

Token	Scopes	Last used	Action
Data Mining	admin:gpg_key, admin:org, admin:org_hook, admin:public_key, admin:repo_hook, delete_repo, gist, notifications, repo, user	Last used within the last day	Edit Delete

Personal access tokens function like ordinary OAuth access tokens. They can be used instead of a password for Git over HTTPS, or can be used to authenticate to the API over Basic Authentication.



API Basics

<https://developer.github.com/v3/>

- The `github` module provides all the necessary interface. We just need to provide the right client token.
- As before, best to keep the credentials in a `dict` and parametrize our calls with the `dict` key. This way we can switch between different accounts easily.
- `.Github(token, per_page=100)` takes a client token as argument and returns a `Github` object that we can use to interact with the API
- 3 basic types of objects:
 - Users
 - Repositories
 - Commits
- The python library we are using works a bit differently from the Twitter one we saw before.
- API calls return `objects`
- Data is stored as `properties` instead of dictionary fields

Authenticating with the API

<https://developer.github.com/v3/>

```
import github

accounts = {
    "social" : TOKEN,
}

token = accounts["social"]

client = github.Github(token, per_page=100)
```

Users

<https://developer.github.com/v3/users/>

- `.get_user(user)` returns the `NamedUser` object corresponding to `user`
- `NamedUser` objects contain many properties with information about the user:
 - `.id` - User ID
 - `.name` - User name
 - `.email` - Email address (not always visible)
 - `.followers / .following` - Number of people follow user / are followed by user
 - `.repos` - Number of public repositories
- Some properties have an associated `get_*` function that returns an iterator over its elements:
 - `.get_followers()`
 - `.get_following()`
 - `.get_repos()`

Challenge

<https://developer.github.com/v3/users/>

- Print the name of the first 10 followers of user:

torvalds

Challenge

<https://developer.github.com/v3/users/>

- Print the name of the first 10 followers of user:

torvalds

```
import github
from github_accounts import accounts

token = accounts["social"]

client = github.Github(token, per_page=100)

screen_name = "torvalds"

user = client.get_user(screen_name)

follow_count = 0

for follow in user.get_followers():
    print(follow_count, follow.name)

    follow_count += 1

    if follow_count == 10:
        break
```

Repos

<https://developer.github.com/v3/repos/>

- Multiple users can have repositories with the same name
- Must ask for a repository belonging to a specific [NamedUser](#) instance.
- `.get_repo(repo)` returns a [Repository](#) object with a similar structure to [NamedUser](#) objects (properties and methods)
 - `.fork` - Whether or not it is a fork
 - `.forks` - How many times it was forked (`.get_forks()`)
 - `.get_commits()` - returns iterator over all [Commit](#) objects
 - `.get_commit(commit_sha)` - return a specific [Commit](#) object
 - `.stargazers_count` - How many times it was stared (`.get_stargazers()`)

Challenge - Repos

- Get the name of the first 10 stargazers for repo

linux

- of user

torvalds

Challenge - Repos

- Get the name of the first 10 stargazers for repo

linux

- of user

```
import github
from github_accounts import accounts

token = accounts["social"]

client = github.Github(token,
per_page=100)

screen_name = "torvalds"
repository_name = "linux"

user = client.get_user(screen_name)
repo = user.get_repo(repository_name)

user_count = 0

for user in repo.get_stargazers():
    print(user_count, user.name)

    user_count += 1

    if user_count == 10:
        break
```

Commits

<https://developer.github.com/v3/repos/commits/>

- **Commit** objects correspond to one of the most fundamental concepts, a change in a piece of code
- Commits belong to one branch and are logically part of a DAG with one or more parents, siblings and children
 - More than one parent indicates that this commit is a merge of multiple previous commits
 - Each commit is identified by a cryptographic **sha** id and refers to its parents



Challenge - Commits

- Build the commit DAG of repository:

Mining-the-Social-Web

- belonging to user:

ptwobrussell

- and print the total number of nodes and edges



Challenge - Commits

- Build the commit DAG of repository:

Mining-the-Social-Web

- belonging to user:

```
import github
from github_accounts import accounts
import networkx as NX

token = accounts["social"]

client = github.Github(token, per_page=100)

screen_name = "ptwobrussell"
repository_name = "Mining-the-Social-Web"

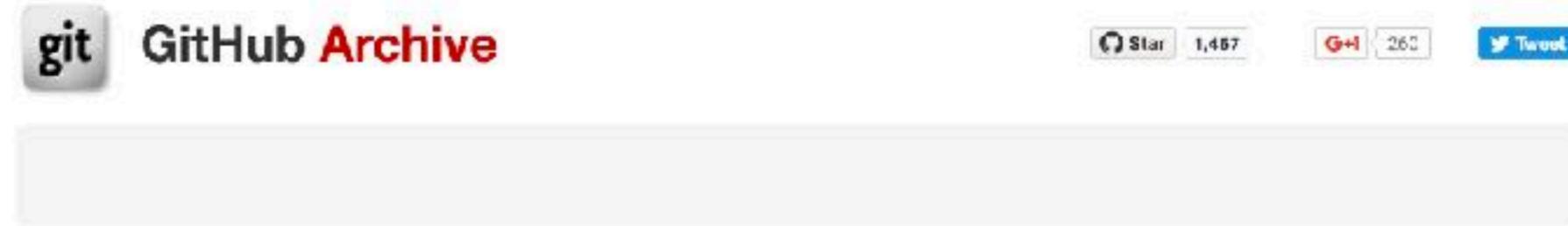
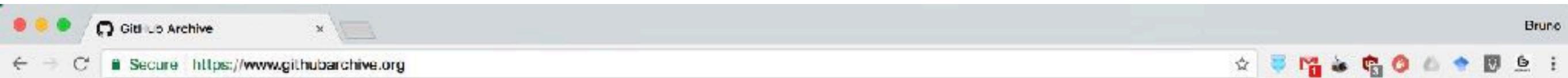
user = client.get_user(screen_name)
repo = user.get_repo(repository_name)

G = NX.DiGraph()

for commit in repo.get_commits():
    for parent in commit.parents:
        G.add_edge(parent.sha, commit.sha)

print(G.number_of_nodes(), G.number_of_edges())
```

GitHub Archive



Open source developers all over the world are working on millions of projects: writing code & documentation, fixing & submitting bugs, and so forth. GitHub Archive is a project to **record** the public GitHub timeline, **archive it**, and **make it easily accessible** for further analysis.

GitHub provides [20+ event types](#), which range from new commits and fork events, to opening new tickets, commenting, and adding members to a project. These events are aggregated into hourly archives, which you can access with any HTTP client:

Query	Command
Activity for 1/1/2015 @ 3PM UTC	<code>wget https://data.githubarchive.org/2015-01-01-15.json.gz</code>
Activity for 1/1/2015	<code>wget https://data.githubarchive.org/2015-01-01-{0..23}.json.gz</code>
Activity for all of January 2015	<code>wget https://data.githubarchive.org/2015-01-{01..30}-{0..23}.json.gz</code>

Each archive contains JSON encoded events as reported by the GitHub API. You can download the raw data and apply own processing to it - e.g. write a custom aggregation script, import it into a database, and so on! An example Ruby script to download and iterate over a single archive:

```
1 require 'open-uri'
2 require 'zlib'
3 require 'yaml'
4
5 gz = open('https://data.githubarchive.org/2015-01-01-12.json.gz')
6 js = Zlib::GzipReader.new(gz).read
7
```

Event Types

- i. [CommitCommentEvent](#)
- ii. [CreateEvent](#)
- iii. [DeleteEvent](#)
- iv. [DeploymentEvent](#)
- v. [DeploymentStatusEvent](#)
- vi. [DownloadEvent](#)
- vii. [FollowEvent](#)
- viii. [ForkEvent](#)
- ix. [ForkApplyEvent](#)
- x. [GistEvent](#)
- xi. [GollumEvent](#)
- xii. [IssueCommentEvent](#)
- xiii. [IssuesEvent](#)
- xiv. [MemberEvent](#)
- xv. [MembershipEvent](#)
- xvi. [PageBuildEvent](#)
- xvii. [PublicEvent](#)
- xviii. [PullRequestEvent](#)
- xix. [PullRequestReviewCommentEvent](#)
- xx. [PushEvent](#)
- xxi. [ReleaseEvent](#)
- xxii. [RepositoryEvent](#)
- xxiii. [StatusEvent](#)
- xxiv. [TeamAddEvent](#)
- xxv. [WatchEvent](#)

Event Structure

<https://developer.github.com/v3/activity/events/types>

- Events Contain information about:
 - Repository (“**repo**”)
 - Actor
 - Type
 - ID
 - Payload - Event specific information

CreateEvent

<https://developer.github.com/v3/activity/events/types>

- Triggered when a repo, branch or tag is created
- Payload includes:
 - “**ref_type**” - type of structure created (repository, branch, tag)
 - “**description**”

ForkEvent

<https://developer.github.com/v3/activity/events/types>

- Triggered when a repository is forked
- Payload contains information about repository that was created (**“forkee”**)

Challenge - ForkEvent

- From the file
<http://data.githubarchive.org/2015-01-01-15.json.gz>
- that you downloaded in the first class, list the “**full_name**” of all “**forkee**” and created repos

Challenge - ForkEvent

- From the file

<http://data.githubarchive.org/2015-01-01-15.json.gz>

- that you downloaded in the first class, list the “**full_name**” of all “**forkee**” and created repos

```
import gzip
import json

filename = "data/2015-01-01-15.json.gz"

for line in gzip.open(filename, 'r'):
    event = json.loads(line.strip().decode())

    if event["type"] == "CreateEvent":
        print("Create", event["repo"]["name"])
    elif event["type"] == "ForkEvent":
        print("Fork", event["payload"]["forkee"]["full name"])
```

PushEvent

<https://developer.github.com/v3/activity/events/types>

- Triggered when commits are pushed to a branch
- Payload includes:
 - List of “**commits**” with “**sha**” ids, “**message**” and “**author**” information
 - ids can then be used with the regular API to obtain more information about the commit or author
 - as “**head**” the sha reference of the head of the branch that is being pushed to

Challenge - PushEvent

- From the file
<http://data.githubarchive.org/2015-01-01-15.json.gz>
- list all the commit ids ("sha") that were pushed along with it parent (you can get information on the parent by querying the API)

```
import gzip
import json
import github
from github_accounts import accounts
import sys

token = accounts["social"]

client = github.Github(token, per_page=100)

filename = "data/2015-01-01-15.json.gz"

for line in gzip.open(filename):
    event = json.loads(line.strip().decode())

    if event["type"] == "PushEvent":
        user_name, repo_name = event["repo"]["name"].split('/')

        try:
            user = client.get_user(user_name)
            repo = user.get_repo(repo_name)

            for commit in event["payload"]["commits"]:
                commit_id = commit["sha"]

                commit = repo.get_commit(commit_id)

                for parent in commit.parents:
                    print(commit_id, parent.sha)
        except Exception as e:
            print("Error processing", event["repo"]["name"], e.status, file=sys.stderr)
```

MemberEvent

<https://developer.github.com/v3/activity/events/types>

- Triggered when a new member is added to a repository
- There is no event for when a developer is removed from a repository
- Payload includes basic information about **member** added
- Actor is the developer who is adding the new member to the team

Challenge - MemberEvent

- Build a social network based on "MemberEvent"s

Challenge - MemberEvent

- Build a social network based on "MemberEvent"s

```
import gzip
import json
import networkx as NX

filename = "data/2015-01-01-15.json.gz"

G = NX.DiGraph()

for line in gzip.open(filename):
    event = json.loads(line.strip().decode())

    if event["type"] == "MemberEvent":
        actor = event["actor"]["login"]
        member = event["payload"]["member"]["login"]

        G.add_edge(actor, member)

print(G.number_of_nodes(), G.number_of_edges())
```

Wikipedia

Turin - Wikipedia

Secure https://en.wikipedia.org/wiki/Turin

Not logged in Talk Contributions Create account Log in

Article Talk Read Edit View history Search Wikipedia

Turin

From Wikipedia, the free encyclopedia

Coordinates: 45°04'N 07°42'E

For other uses, see [Turin \(disambiguation\)](#).

"[Torino](#)" redirects here. For other uses, see [Torino \(disambiguation\)](#).

Turin (/*vɪərɪn/; Italian: *Torino*, pronounced [toˈriːno] ([listen](#)); Piedmontese: *Turin*, pronounced [tyrɪn])^[2] is a city and an important business and cultural centre in northern Italy, capital of the Piedmont region and was the first capital city of Italy. The city is located mainly on the western bank of the Po River, in front of Susa Valley and surrounded by the western Alpine arch and by the Superga Hill. The population of the city proper is 892,649 (August 2015) while the population of the urban area is estimated by Eurostat to be 1.7 million inhabitants. The Turin metropolitan area is estimated by the OECD to have a population of 2.2 million.^[3]*

In 1997 a part of the historical center of Torino was inscribed in the World Heritage List under the name Residences of the Royal House of Savoy.

The city has a rich culture and history, and is known for its numerous art galleries, restaurants, churches, palaces, opera houses, piazzas, parks, gardens, theatres, libraries, museums and other venues. Turin is well known for its Renaissance, Baroque, Rococo, Neo-classical, and Art Nouveau architecture.

Many of Turin's public squares, castles, gardens and elegant *palazzi* such as [Palazzo Madama](#), were built between the 16th and 18th centuries. This was after the capital of the Duchy of Savoy (later Kingdom of Sardinia) was moved to Turin from Chambery (now in France) as part of the urban expansion.

The city used to be a major European political center. Turin was Italy's first capital city in 1861 and home to the [House of Savoy](#), Italy's royal family.^[4] From 1563, it was the capital of the [Duchy of Savoy](#), then of the Kingdom of Sardinia ruled by the Royal House of Savoy and finally the first capital of the [unified Italy](#).^[5] Turin is sometimes called "the cradle of Italian liberty" for having been the birthplace and home of notable politicians and people who contributed to the [Risorgimento](#), such as [Cavour](#).^[6]

The city currently hosts some of Italy's best universities, colleges, academies, lycea and gymnasia, such as the University of Turin, founded in the 15th century, and the [Turin Polytechnic](#). In addition, the city is home to museums such as the [Museo Egizio](#)^[7] and the [Mole Antonelliana](#). Turin's attractions make it one of the world's top 250 tourist destinations and the tenth most visited city in Italy in 2008.^[8]

Even though much of its political significance and importance had been lost by [World War II](#), Turin became a major European crossroad for industry, commerce and trade, and is part of the famous "industrial triangle" along with [Milan](#) and [Genoa](#). Turin is ranked third in Italy, after Milan and Rome, for economic strength.^[9] With a GDP of \$53 billion, Turin is the world's 78th richest city by purchasing power.^[10] As of 2010, the city has

Turin
Torino
Comune
Città di Torino

Flag

Coat of arms

Map of Italy showing location of Turin

Map of Piedmont showing location of Turin

Image of the Turin skyline

Turin: Revision history - Wikipedia

Secure https://en.wikipedia.org/w/index.php?title=Turin&action=history

Not logged in Talk Contributions Create account Log in

Article Talk Read Edit View history Search Wikipedia

Turin: Revision history

View logs for this page

Search for revisions

From year (and earlier): 2017 From month (and earlier): all Tag filter: Show

For any version listed below, click on its date to view it. For more help, see Help:Page history and Help>Edit summary.

External tools: Revision history statistics · Revision history search · Edits by user · Number of watchers · Page view statistics · Fix dead links

(cur) = difference from current version. (prev) = difference from preceding version, m = minor edit, → = section edit, ← = automatic edit summary
(newest | oldest) View (newer 50 | older 50) (20 | 50 | 100 | 250 | 500)

Compare selected revisions

- (cur | prev) 21:36, 11 April 2017 79.40.21.128 (talk) . . (104,629 bytes) (+39) . . (→Media) (undo)
- (cur | prev) 02:57, 2 April 2017 GreenC bot (talk | contribs) m . . (104,490 bytes) (+37) . . (Reformat 1 archive link. Wayback Media 2.1) (undo)
- (cur | prev) 19:12, 26 March 2017 Crisatuco (talk | contribs) . . (104,453 bytes) (+73) . . (→External links) (undo)
- (cur | prev) 21:14, 25 March 2017 84.220.92.23 (talk) . . (104,380 bytes) (+20) . . (other Latin name) (undo)
- (cur | prev) 21:12, 25 March 2017 84.220.92.23 (talk) . . (104,360 bytes) (0) . . (undo)
- (cur | prev) 21:08, 25 March 2017 84.220.92.23 (talk) . . (104,360 bytes) (-21) . . (why Lombard??) (undo)
- (cur | prev) 20:28, 25 March 2017 Kind Tennis Fan (talk | contribs) m . . (104,381 bytes) (+15) . . (Consistent date format. Date formats per MOS:DATEFORMAT by script!) (undo)
- (cur | prev) 13:39, 22 March 2017 Alaney2k (talk | contribs) m . . (104,366 bytes) (+16) . . (updated city to include province using AWB) (undo)
- (cur | prev) 22:28, 17 March 2017 84.221.236.224 (talk) . . (104,350 bytes) (+1) . . (→City centre) (undo)
- (cur | prev) 22:27, 17 March 2017 84.221.236.224 (talk) . . (104,349 bytes) (+16) . . (→City centre) (undo)
- (cur | prev) 22:26, 17 March 2017 84.221.236.224 (talk) . . (104,330 bytes) (+15) . . (→City centre) (undo)
- (cur | prev) 19:20, 17 March 2017 84.223.252.94 (talk) . . (104,315 bytes) (+208) . . (undo) (Tag: Visual edit)
- (cur | prev) 21:23, 28 February 2017 86.131.110.191 (talk) . . (104,107 bytes) (+22) . . (undo)
- (cur | prev) 21:19, 28 February 2017 Cristianjf (talk | contribs) . . (104,085 bytes) (+6) . . (undo)
- (cur | prev) 21:18, 28 February 2017 Cristianjf (talk | contribs) . . (104,079 bytes) (-6) . . (undo)

User:Kind Tennis Fan - Wikipedia

Secure https://en.wikipedia.org/wiki/User:Kind_Tennis_Fan

Not logged in Talk Contributions Create account Log in

User page [Talk](#) Read Edit View history Search Wikipedia

User:Kind Tennis Fan

From Wikipedia, the free encyclopedia



Main page
Contents
Featured content
Current events
Random article
Donations to Wikipedia
Wikipedia store

Interaction
Help
About Wikipedia
Community portal
Recent changes
Contact page

Tools
What links here
Related changes
User contributions
Logs
View user groups
Upload file
Special pages
Permanent link
Page information

Print/export
Create a book
Download as PDF
Printable version

About me [edit]


This editor is a Veteran Editor IV and is entitled to display this Gold Editor Star.

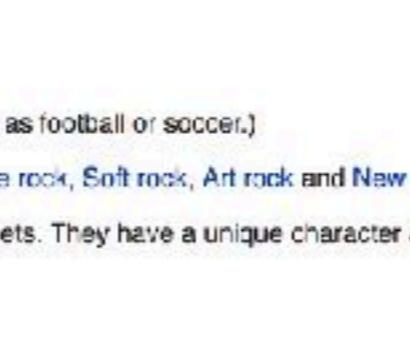
I'm a male, born in the United Kingdom, and I've spent the vast majority of my life so far living in the south of England. My marital status is single and I currently have a girlfriend.

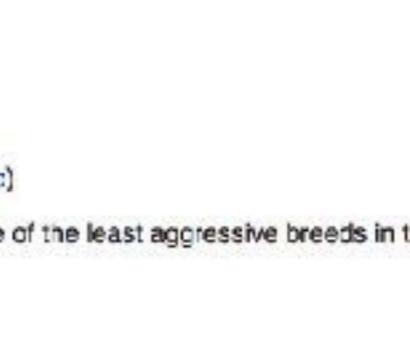
I registered as a Wikipedia user in July 2013, as I have many different interests and subjects that I like to read about.

Passions [edit]

Tennis

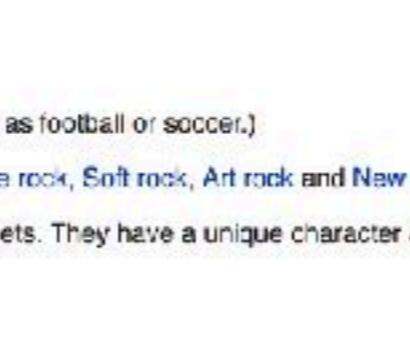

Golf

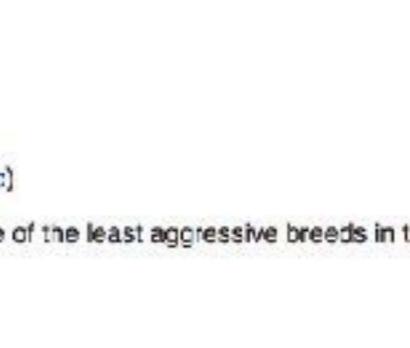

Association football (more commonly known as football or soccer.)


Rock music (In particular: melodic Alternative rock, Soft rock, Art rock and New wave music)


Pugs (Highly recommended as nice gentle pets. They have a unique character and are one of the least aggressive breeds in the world.)


Other interests [edit]

Politics


Contemporary history (Particularly the tragic conflict known as The Troubles where more than 3,500 people have been killed and over 50,000 people wounded. I would like to see the two main communities continue to work towards peace and reconciliation.)


Restaurants and Cuisine


British New Wave films with an element of social realism. (Such as *Room at the Top* and *Saturday Night and Sunday Morning*.)


Psychology

User:GreenC bot - Wikipedia

Secure https://en.wikipedia.org/wiki/User:GreenC_bot

Not logged in Talk Contributions Create account Log in

User page Talk Read View source View history Search Wikipedia

User:GreenC bot

From Wikipedia, the free encyclopedia

This user account is a **bot** that uses **AutoWikiBrowser**, operated by **Green Cardamom** ([talk](#)). It is a legitimate alternative account, used to make repetitive automated or semi-automated edits that would be extremely tedious to do manually. The bot is approved and currently active – the relevant request for approval can be seen [here](#).

To stop this bot until restarted by the bot's owner, edit its [talk](#) page. If that page is a redirect, edit that original redirecting page, not the target of the redirect.

You can stop the bot by pushing the **stop button**. The bot sees and immediately stops running. Unless it is an emergency please consider reporting problems first to my [talk page](#).

GreenC Bot is a bot account operated by [GreenC](#).

Contents [hide]

- 1 [Bot jobs](#)
 - 1.1 [Job #1](#)
 - 1.2 [Job #2](#)
 - 1.3 [Job #3](#)

Bot jobs

Job #1

Green C Bot Job #1 ("Wayback Medic"). WaybackMedic fixes known problems with Internet Archive Wayback Machine links.

✓ - Job completed.

Job #2

Green C Bot Job #2 ("Wayback Medic 2"). WaybackMedic 2 fixes known problems with Internet Archive Wayback Machine links.

✓ - Initial job completed. Further work as new links are added.

Main page
Contents
Featured content
Current events
Random article
Donations to Wikipedia
Wikipedia store

Interaction
Help
About Wikipedia
Community portal
Recent changes
Contact page

Tools
What links here
Related changes
User contributions
Logs
View user groups
Upload file
Special pages
Permanent link
Page information

Print/export
Create a book
Downloaded as PDF
Printable version

W Talk:Turin - Wikipedia

Secure https://en.wikipedia.org/wiki/Talk:Turin

Not logged in Talk Contributions Create account Log in

Article Talk Read Edit New section View history Search Wikipedia

Talk:Turin

From Wikipedia, the free encyclopedia

 Turin has been listed as a level-4 **vital article** in Geography. If you can improve it, please do. This article has been rated as C-Class.

 This article is of interest to the following WikiProjects: [hide]

- WikiProject Italy (Rated C-class, Top-importance) [show]
- WikiProject Cities (Rated C-class, High-importance) [show]
- WikiProject Olympics / Paralympics (Rated C-class, Mid-importance) [show]
- WikiProject Version 1.0 Editorial Team / v0.5 / Vital (Rated C-class) [show]

 This article is/was the subject of a Wiki Education Foundation-supported course assignment. Further details are available on the course page. Assigned peer reviews: NicholasKZalewski.

Contents [hide]

- 1 Expansion of Main Sights
- 2 Legends
- 3 Nazi
- 4 Requested move
 - 4.1 Discussion
 - 4.1.1 Torino v Turin
 - 4.1.2 From "Turin" to "Torino"
- 5 Education
- 6 Google
- 7 Fiat
- 8 Torino

This article contains a translation of Torino from [it.wikipedia.org](#).

Wikipedia Dumps

<https://dumps.wikimedia.org>

The screenshot shows a web browser window titled "Wikimedia Downloads". The address bar indicates a secure connection to "https://dumps.wikimedia.org". The main content area features a large, bold heading "Wikimedia Downloads". Below it is a note about rate limiting and a call to action for volunteers to host mirrors. The page then details "Database backup dumps" and "Static HTML dumps", both of which are currently not running. A link to "DVD distributions" is also present.

Wikimedia Downloads

If you are reading this on Wikimedia servers, please note that we have rate limited downloaders and we are capping the number of per-ip connections to 2. This will help to ensure that everyone can access the files with reasonable download times. Clients that try to evade these limits may be blocked. Our mirror sites do not have this cap.

Data downloads

The Wikimedia Foundation is requesting help to ensure that as many copies as possible are available of all Wikimedia database dumps. Please [volunteer to host a mirror](#) if you have access to sufficient storage and bandwidth.

Database backup dumps

A complete copy of all Wikimedia wikis, in the form of wikitext source and metadata embedded in XML. A number of raw database tables in SQL form are also available.

These snapshots are provided at the very least monthly and usually twice a month. If you are a regular user of these dumps, please consider subscribing to [xmldatadumps-l](#) for regular updates.

Mirror Sites of the XML dumps provided above

Check the [complete list](#).

Static HTML dumps

A copy of all pages from all Wikipedia wikis, in HTML form.

These are currently not running.

DVD distributions

Wikipedia Dumps

<https://dumps.wikimedia.org>

- The Wikimedia foundation makes freely available regular dumps of all Wikimedia project databases.
- In particular, for the various language editions of Wikipedia, we have:
 - ***.pages-articles.xml.bz2** - Complete wiki page and revision content.
 - ***.stub-meta-history.xml.gz** - Wiki page revision metadata
 - ***.pagelinks.sql.gz** - Wiki page-to-page link records
 - ***.geo_tags.sql.gz** - List of pages' geographical coordinates
 - ***.externallinks.sql.gz** - Wiki external URL link records.
 - ***.page.sql.gz** - Base per-page data (id, title, old restrictions, etc).
 - ***.langlinks.sql.gz** - Wiki interlanguage link records

Wikipedia Dumps

<https://dumps.wikimedia.org>

- The Wikimedia foundation makes freely available regular dumps of all Wikimedia project databases.
- In particular, for the various language editions of Wikipedia, we have:
 - *.pages-articles.**xml.bz2** - Complete wiki page and revision content.
 - *.stub-meta-history.**xml.gz** - Wiki page revision metadata
 - *.pagelinks.**sql.gz** - Wiki page-to-page link records
 - *.geo_tags.**sql.gz** - List of pages' geographical coordinates
 - *.externallinks.**sql.gz** - Wiki external URL link records.
 - *.page.**sql.gz** - Base per-page data (id, title, old restrictions, etc).
 - *.langlinks.**sql.gz** - Wiki interlanguage link records

Wikipedia Dumps

<https://dumps.wikimedia.org>

- The Wikimedia foundation makes freely available regular dumps of all Wikimedia project databases.
- In particular, for the various language editions of Wikipedia, we have:
 - *.pages-articles.**xml.bz2** - Complete wiki page and revision content.
 - *.stub-meta-history.**xml.gz** - Wiki page revision metadata
 - *.pagelinks.**sql.gz** - Wiki page-to-page link records
 - *.geo_tags.**sql.gz** - List of pages' geographical coordinates
 - *.externallinks.**sql.gz** - Wiki external URL link records.
 - *.page.**sql.gz** - Base per-page data (id, title, old restrictions, etc).
 - *.langlinks.**sql.gz** - Wiki interlanguage link records

(Wikipedia Dump “Dumping”)

- I've written a simple script to easily download the most recent version of specific files for many different languages.
- You can find it in the GitHub repo: [wikidump.py](#)
- To customize to your needs, you just need to list the files you want in the **allowed_files** list and the languages you're interested in **allowed_wikis**
- If you want to download all the files from **acewiki** required for this tutorial, you would simply set:

```
allowed_files = ["stub-meta-history.xml.gz",
                 "geo_tags.sql.gz",
                 "langlinks.sql.gz",
                 ]
allowed_wikis = ["acewiki"]
```

- But with what you learn so far you should be able to easily write your own version 😊

SQL files

- Standard format, well suited for loading the data directly to a relational database (MySQL, MariaDB, PostgreSQL, etc...)
- Databases are optimized for fast querying of information, but not suitable for large scale processing where you touch all or most rows.
- `mysqldump_to_csv.py` - convert a wikipedia dump to a CSV file.
 - Slightly modified version of <https://github.com/jamesmishra/mysqldump-to-csv>
 - Available in the courses GitHub repository
 - First row is column names as defined in the SQL file

langlinks

- Just a few fields:
 - 0 - `ll_from` - The page in **this** wikipedia edition
 - 1 - `ll_lang` - The language it's linking to
 - 2 - `ll_title` - The title of the page in the **target** wikipedia edition
- This is a good example of some of the problems of working with wikipedia data, or any other self organize collaboration platform
- Many of the file formats and conventions were created in an ad hoc way, to serve one very specific need and ended up becoming adopted as "standard".
 - How can we convert the **language/title** pairs into a unique **page_id** in the target wikipedia?
 - Can we be sure that two pages didn't accidentally switch titles?
 - As pages get edited, their titles change. To **when** (which revision) do these titles correspond to?
 - Does a link A -> B imply a link B -> A?

Challenge - langlinks

- convert the
`data/acewiki-20170420-langlinks.sql.gz`
- SQL file to CSV using the `mysqldump2csv.py` script.

Challenge - langlinks

- convert the
data/acewiki-20170420-langlinks.sql.gz
- SQL file to CSV using the **mysqldump2csv.py** script.

```
python mysqldump_to_csv.py data/acewiki-20170420-langlinks.sql.gz | gzip -c > data/  
acewiki-20170420-langlinks.csv.gz
```

geo_tags

- Several interesting fields:
 - 0 - **gt_id** - Unique geo tag ID
 - 1 - **gt_page_id** - Corresponding Page ID
 - 2 - **gt_globe** - Not all coordinates are on Earth (Mars, Moon, Venus, Titan, etc...)
 - 4 - **gt_lat** - Latitude
 - 5 - **gt_lon** - Longitude
 - 7 - **gt_type** - city, railwaystation, landmark, airport, etc...

Challenge - geo_tags

- Convert `data/enwiki-20170420-geo_tags.sql.gz` to csv using `mysqldump_to_csv.py`
- Extract all the `lat`, `lon` pairs on planet "earth".

Challenge - geo_tags

- Convert `data/enwiki-20170420-geo_tags.sql.gz` to csv using `mysqldump_to_csv.py`
- Extract all the **lat, lon** pairs on planet "earth".

```
import gzip

header = {}
line_count = 0

for line in gzip.open("data/enwiki-20170420-geo_tags.csv.gz", "rt"):
    fields = line.strip().split(',')

    if line_count == 0:
        header = dict(zip(fields, range(len(fields)))))

    line_count += 1

    if(fields[header["gt_globe"]] == "earth"):
        print(fields[header["gt_lat"]], fields[header["gt_lon"]])
```

expat - (semi) sane XML parsing

<https://docs.python.org/3/library/pyexpat.html>

- C library for parsing XML with bindings in most modern programming languages
- Extremely fast
- Well suited to handle large xml files:
 - Stream oriented - Reads the file line by line
 - Non-validating - doesn't check for the validity of the XML file (expensive and prone to failure)
- In Python it lives inside the `xml.parsers` package

```
from xml.parsers import expat
```

- The `.ParserCreate()` method returns a new `xmlparser` instance

expat - (semi) sane XML parsing

<https://docs.python.org/3/library/pyexpat.html>

- Defines event handlers that get called whenever it encounters something "interesting"
- The default behavior is to do nothing (very efficient!) but you can override the ones that you are interested in.
- In particular:
 - `.StartElementHandler(name, attrs)` - every time it encounters a `<name ...>`
 - `.EndElementHandler(name)` - whenever it encounter a `</name>`
 - `.CharacterDataHandler(data)` - any textual data in between the opening and closing of a tag:
 - `<name>data</name>`
 - If the amount of data between these two tags is too large, it sometimes results in multiple `char_data` events. You should always concatenate the results as you get it
- After you overwrite the relevant methods, you can process the file by providing a file handle to `.ParserFile(fp)`

expat - (semi) sane XML parsing

<https://docs.python.org/3/library/pyexpat.html>

```
import sys
from xml.parsers import expat

buffer = ""
level = 0

def start_element(name, attrs):
    global buffer, level
    print("\t" * level, "Opening:", name, "with attributes:", attrs)
    buffer = ""
    level += 1

def end_element(name):
    global buffer, level
    level -= 1
    print("\t" * level, "Closing:", name, "with data:", buffer)
    buffer = ""

def char_data(data):
    global buffer
    buffer += data

if __name__ == "__main__":
    p = expat.ParserCreate()
    p.StartElementHandler = start_element
    p.EndElementHandler = end_element
    p.CharacterDataHandler = char_data

    try:
        p.ParseFile(open(sys.argv[1], 'rb'))
    except Exception as e:
        print(e, file=sys.stderr)
```

```
<mediawiki xmlns="http://www.mediawiki.org/xml/export-0.10/" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://www.mediawiki.org/xml/export-0.10/ http://www.mediawiki.org/xml/export-0.10.xsd"
version="0.10" xml:lang="en">
<siteinfo>
<sitename>Wikipedia</sitename>
<dbname>enwiki</dbname>
<base>https://en.wikipedia.org/wiki/Main_Page</base>
<generator>MediaWiki 1.29.0-wmf.20</generator>
<case>first-letter</case>
<namespaces>
<namespace key="-2" case="first-letter">Media</namespace>
<namespace key="-1" case="first-letter">Special</namespace>
<namespace key="0" case="first-letter" />
<namespace key="1" case="first-letter">Talk</namespace>
<namespace key="2" case="first-letter">User</namespace>
<namespace key="3" case="first-letter">User talk</namespace>
<namespace key="4" case="first-letter">Wikipedia</namespace>
<namespace key="5" case="first-letter">Wikipedia talk</namespace>
<namespace key="6" case="first-letter">File</namespace>
<namespace key="7" case="first-letter">File talk</namespace>
<namespace key="8" case="first-letter">MediaWiki</namespace>
<namespace key="9" case="first-letter">MediaWiki talk</namespace>
<namespace key="10" case="first-letter">Template</namespace>
<namespace key="11" case="first-letter">Template talk</namespace>
<namespace key="12" case="first-letter">Help</namespace>
<namespace key="13" case="first-letter">Help talk</namespace>
<namespace key="14" case="first-letter">Category</namespace>
<namespace key="15" case="first-letter">Category talk</namespace>
<namespace key="100" case="first-letter">Portal</namespace>
<namespace key="101" case="first-letter">Portal talk</namespace>
<namespace key="108" case="first-letter">Book</namespace>
<namespace key="109" case="first-letter">Book talk</namespace>
<namespace key="118" case="first-letter">Draft</namespace>
<namespace key="119" case="first-letter">Draft talk</namespace>
<namespace key="446" case="first-letter">Education Program</namespace>
<namespace key="447" case="first-letter">Education Program talk</namespace>
<namespace key="710" case="first-letter">TimedText</namespace>
<namespace key="711" case="first-letter">TimedText talk</namespace>
<namespace key="828" case="first-letter">Module</namespace>
<namespace key="829" case="first-letter">Module talk</namespace>
<namespace key="2300" case="first-letter">Gadget</namespace>
<namespace key="2301" case="first-letter">Gadget talk</namespace>
<namespace key="2302" case="case-sensitive">Gadget definition</namespace>
<namespace key="2303" case="case-sensitive">Gadget definition talk</namespace>
</namespaces>
</siteinfo>
```

```

<mediawiki xmlns="http://www.mediawiki.org/xml/export-0.10/" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:schemaLocation="http://www.mediawiki.org/xml/export-0.10/ http://www.mediawiki.org/xml/export-0.10.xsd" version="0.10" xml:lang="en">
<siteinfo>
<sitename>Wikipedia</sitename>
<dbname>enwiki</dbname>
<base>https://en.wikipedia.org/wiki/Main_Page</base>
<generator>MediaWiki 1.29.0-wmf.20</generator>
<case>first-letter</case>
<namespaces>
<namespace key="-2" case="first-letter">Media</namespace>
<namespace key="-1" case="first-letter">Special</namespace>
<namespace key="0" case="first-letter" />
<namespace key="1" case="first-letter">Talk</namespace>
<namespace key="2" case="first-letter">User</namespace>
<namespace key="3" case="first-letter">User talk</namespace>
<namespace key="4" case="first-letter">Wikipedia</namespace>
<namespace key="5" case="first-letter">Wikipedia talk</namespace>
<namespace key="6" case="first-letter">File</namespace>
<namespace key="7" case="first-letter">File talk</namespace>
<namespace key="8" case="first-letter">MediaWiki</namespace>
<namespace key="9" case="first-letter">MediaWiki talk</namespace>
<namespace key="10" case="first-letter">Template</namespace>
<namespace key="11" case="first-letter">Template talk</namespace>
<namespace key="12" case="first-letter">Help</namespace>
<namespace key="13" case="first-letter">Help talk</namespace>
<namespace key="14" case="first-letter">Category</namespace>
<namespace key="15" case="first-letter">Category talk</namespace>
<namespace key="100" case="first-letter">Portal</namespace>
<namespace key="101" case="first-letter">Portal talk</namespace>
<namespace key="108" case="first-letter">Book</namespace>
<namespace key="109" case="first-letter">Book talk</namespace>
<namespace key="118" case="first-letter">Draft</namespace>
<namespace key="119" case="first-letter">Draft talk</namespace>
<namespace key="446" case="first-letter">Education Program</namespace>
<namespace key="447" case="first-letter">Education Program talk</namespace>
<namespace key="710" case="first-letter">TimedText</namespace>
<namespace key="711" case="first-letter">TimedText talk</namespace>
<namespace key="828" case="first-letter">Module</namespace>
<namespace key="829" case="first-letter">Module talk</namespace>
<namespace key="2300" case="first-letter">Gadget</namespace>
<namespace key="2301" case="first-letter">Gadget talk</namespace>
<namespace key="2302" case="case-sensitive">Gadget definition</namespace>
<namespace key="2303" case="case-sensitive">Gadget definition talk</namespace>
</namespaces>
</siteinfo>

```

Wikipedia data structure		
Namespaces		
Subject namespaces	Talk namespaces	
0 (Main/Article)	Talk	1
2 User	User talk	3
4 Wikipedia	Wikipedia talk	5
6 File	File talk	7
8 MediaWiki	MediaWiki talk	9
10 Template	Template talk	11
12 Help	Help talk	13
14 Category	Category talk	15
100 Portal	Portal talk	101
108 Book	Book talk	109
118 Draft	Draft talk	119
446 Education Program	Education Program talk	447
710 TimedText	TimedText talk	711
828 Module	Module talk	829
2300 Gadget	Gadget talk	2301
2302 Gadget definition	Gadget definition talk	2303
Virtual namespaces		
-1 Special		
-2 Media		

Revision file format

```
<page>
  <title>Ôn Keuë</title>
  <ns>0</ns>
  <id>1</id>
  <revision>
    <id>1028</id>
    <timestamp>2008-04-13T07:53:23Z</timestamp>
    <contributor>
      <ip>125.162.38.87</ip>
    </contributor>
    <comment>New page: Jinoë droën neuh ka neutamong lam Wikipèdia Acèh. Wikipèdia Acèh  
nyoë mantöng geu'ijoë, geukalön peuë ék na soë peudawôk peuë h'an. Meunyoë le nyang pakoë,  
Wikipèdi...</comment>
    <model>wikitext</model>
    <format>text/x-wiki</format>
    <text id="815" bytes="3106" />
    <sha1>43iy7hfjh19xt1683z27ii0ie35z9am</sha1>
  </revision>
  <revision>
    <id>1029</id>
    <parentid>1028</parentid>
    <timestamp>2008-04-13T08:01:10Z</timestamp>
    <contributor>
      <username>Si Gam Acèh</username>
      <id>0</id>
    </contributor>
    <comment>Removing all content from page</comment>
    <model>wikitext</model>
    <format>text/x-wiki</format>
    <text id="816" bytes="0" />
    <sha1>phoiac9h4m842xq45sp7s6u21eteeq1</sha1>
  </revision>
</page>
```

Challenge - expat

- Extract the **article** revision information onto a csv file with the format:

page_id, revision_id, timestamp, title

In the file:

data/acewiki-20170420-stub-meta-history.xml.gz

```
<page>
  <title>Ôn Keuë</title>
  <ns>0</ns>
  <id>1</id>
  <revision>
    <id>1028</id>
    <timestamp>2008-04-13T07:53:23Z</timestamp>
    <contributor>
      <ip>125.162.38.87</ip>
    </contributor>
    <comment>New page: Jinoë droën neuh ka neutamong lam Wikipèdia Acèh. Wikipèdia Acèh nyöe mantöng geu' ujoë, geukalön peuë ék na soë peudawôk peuë h'an Meunyoë le nyang pakoë, Wikipèdi...</comment>
    <model>wikitext</model>
    <format>text/x-wiki</format>
    <text id="815" bytes="3106" />
    <sha1>43iy7hfjh19xt1683z27ii0ie35z9am</sha1>
  </revision>
  <revision>
    <id>1029</id>
    <parentid>1028</parentid>
    <timestamp>2008-04-13T08:01:10Z</timestamp>
    <contributor>
      <username>Si Gam Acèh</username>
      <id>0</id>
    </contributor>
    <comment>Removing all content from page</comment>
    <model>wikitext</model>
    <format>text/x-wiki</format>
    <text id="816" bytes="0" />
    <sha1>phoiac9h4m842xq45sp7s6u21eteeq1</sha1>
  </revision>
</page>
```

data/page.xml

Challenge - expat

- Extract the **article** revision information onto a csv file with the format:

page_id, revision_id, timestamp, title

In the file:

data/acewiki-20170420-stub-meta-history.xml.gz

There are 3 <id> tags. You have to keep track of which one you're in!

```
<page>
  <title>Ôn Keuë</title>
  <ns>0</ns>
  <id>1</id>
  <revision>
    <id>1028</id>
    <timestamp>2008-04-13T07:53:23Z</timestamp>
    <contributor>
      <ip>125.162.38.87</ip>
    </contributor>
    <comment>New page: Jinoë droën neuh ka neutamong lam Wikipèdia Acèh. Wikipèdia Acèh nyoë mantöng geu' ujoë, geukalön peuë ék na soë peudawôk peuë h'an Meunyoë le nyang pakoë, Wikipèdi...</comment>
    <model>wikitext</model>
    <format>text/x-wiki</format>
    <text id="815" bytes="3106" />
    <sha1>43iy7hfjh19xt1683z27ii0ie35z9am</sha1>
  </revision>
  <revision>
    <id>1029</id>
    <parentid>1028</parentid>
    <timestamp>2008-04-13T08:01:10Z</timestamp>
    <contributor>
      <username>Si Gam Acèh</username>
      <id>0</id>
    </contributor>
    <comment>Removing all content from page</comment>
    <model>wikitext</model>
    <format>text/x-wiki</format>
    <text id="816" bytes="0" />
    <sha1>phoiac9h4m842xq45sp7s6u21eteeq1</sha1>
  </revision>
</page>
```

data/page.xml

Challenge - expat

```
import sys
import gzip
from xml.parsers import expat

isContributor = False
isArticle = False
isPage = False
isData = False
buffer = u"""
page_id = None
timestamp = None
revision_id = None

fields = set(["timestamp", "page", "id", "ns", "revision", "contributor"])

def start_element(name, attrs):
    global buffer, isData, isPage, isContributor

    if name in fields:
        buffer = ""

        if name == "page":
            isPage = True
        elif name == "revision":
            isPage = False
        elif name == "contributor":
            isContributor = True
        else:
            isData = True
```

Challenge - expat

```
def end_element(name):
    global buffer, isData, isPage, isArticle, isContributor, timestamp, page_id, revision_id

    if name in fields:
        if name == "ns":
            if int(buffer) == 0:
                isArticle = True
            else:
                isArticle = False
        elif name == "timestamp":
            timestamp = buffer
        elif name == "id":
            if isPage:
                page_id = buffer
                isPage = False
            elif not isContributor:
                revision_id = buffer
        elif name == "revision":
            if isArticle:
                print(", ".join([page_id, revision_id, timestamp]))
        elif name == "page":
            isArticle = False
        elif name == "contributor":
            isContributor = False

    buffer = ""
    isData = False
```

Challenge - expat

```
def char_data(data):
    global isData, buffer

    if isData:
        buffer += data

if __name__ == "__main__":
    p = expat.ParserCreate()

    p.StartElementHandler = start_element
    p.EndElementHandler = end_element
    p.CharacterDataHandler = char_data

    print(",".join(["page_id", "revision_id", "timestamp", "title"]))

try:
    p.ParseFile(gzip.open(sys.argv[1]))
except Exception as e:
    print(e, file=sys.stderr)
```

Challenge - Matching titles

- As we saw before, matching titles and page_id is not easy. The only place where the two field appear together is in the revisions files. Fortunately, we already know how to process those.
- Even more fortunately, the Wikimedia foundation also makes available the **stub-meta-current.xml.gz** that have a similar format to the **stub-meta-history.xml.gz** files but include only the current revision of each page.
- This challenge has two parts:
 - Convert **data/abwiki-20170420-stub-meta-current.xml.gz** to CSV
 - Use your newly generated **data/abwiki-20170420-stub-meta-current.csv.gz** to match the titles in **data/acewiki-20170420-langlinks.csv.gz** to the page_id in for the **abwiki** wikipedia edition.

```

import gzip

header = {}
line_count = 0

ll_from = {}
for line in gzip.open("data/acewiki-20170420-langlinks.csv.gz", "rt"):
    fields = line.strip().split(',')

    if line_count == 0:
        header = dict(zip(fields, range(len(fields)))))

    line_count += 1

    if fields[header["ll_lang"]] == "ab":
        key = fields[header["ll_title"]]

        ll_from[key] = fields[header["ll_from"]]

header = {}
line_count = 0

for line in gzip.open("data/abwiki-20170420-stub-meta-current.csv.gz", "rt"):
    fields = line.strip().split(',')

    if line_count == 0:
        header = dict(zip(fields, range(len(fields)))))

    line_count += 1

    title = fields[header["title"]]

    if title in ll_from:
        print("ace", ll_from[title], "ab", fields[header["page_id"]])

```