



さくらのナレッジ > エンジニア向け > Gitリポジトリ上のソースコードをWebブラウザから検索・分析できるコード検索ツール「Sourcegraph」

Gitリポジトリ上のソースコードを Webブラウザから検索・分析できる コード検索ツール 「Sourcegraph」

エンジニア向け 2019.07.09



The screenshot shows the Sourcegraph web interface. At the top, the search query is 'nodisplay lang:python'. Below the search bar, there are filters for 'lang:python' and 'case:yes'. The search results show 5 results in 0.00 seconds. The first result is from 'gitroot/newslash/newslash.git' and shows a snippet of Python code from 'src/ns_search/newslash_db/polls.py'. The second result is from 'gitroot/newslash/newslash.git' and shows a snippet of Python code from 'src/ns_search/newslash_index/document.py'. The third result is from 'gitroot/newslash/newslash.git' and shows a snippet of Python code from 'src/ns_search/test/test_newslash_index.py'. At the bottom, there is a link to 'Not seeing expected results?'.

[Tweet](#)[シェア 43](#)

68

大規模なソースコードを分析・閲覧する際に有用なのが、ソースコード専用の検索ツールだ。こういったツールは複数あるが、今回はGitリポジトリやGitHubなどのホスティングサービスとの連携機能を特徴とするオープンソースの検索ツール「Sourcegraph」を紹介する。

Webブラウザ上からGUIで操作できるソースコード検索ツール

システム開発の現場において、クラスや関数がどこでどのように定義されているのかを調べたり、メッセージなどで表示される特定の文字列がどこで使われているかを調べたりするといった作業は割と頻繁に発生する。こういった作業はgrepなどの汎用検索ツールでも可能ではあるが、その場合たとえばソースコード以外のファイルに含まれる文字列も検索に引っかかるほか、その出力もやや見にくい。そこで活用したいのが、専用のソースコード検索ツールだ。

ソースコード検索ツールでは、対象のソースコードがどのプログラム言語で書かれているかを判別した上でその文法に従って解析を行う。そのため、単なる全文検索ではなくたとえばクラス名や関数名、変数名のみを対象に検索を行うといったことが可能だ。また、クラス名や関数名などのインデックスを作成し、クラス一覧や関数一覧などを表示したり、そこからクラスや関数を検索したりする機能を持つものもある。

ソースコード検索ツールは商用のものからフリーのものまでさまざまなものがあるほか、Visual Studioなどの統合開発環境にも似たような機能が搭載されている。ただし、その場合現在開いているソースコードやディレクトリ、プロジェクトのみが対象だったり、ローカルにソースコード一式を用意する必要があると、単にソースコードを閲覧したいだけの場合でも利用のための作業がやや面倒な傾向がある。

そこで今回紹介するのが、Webブラウザから操作できるソースコード検索ツールである「Sourcegraph」 (F

[Page Top](#)

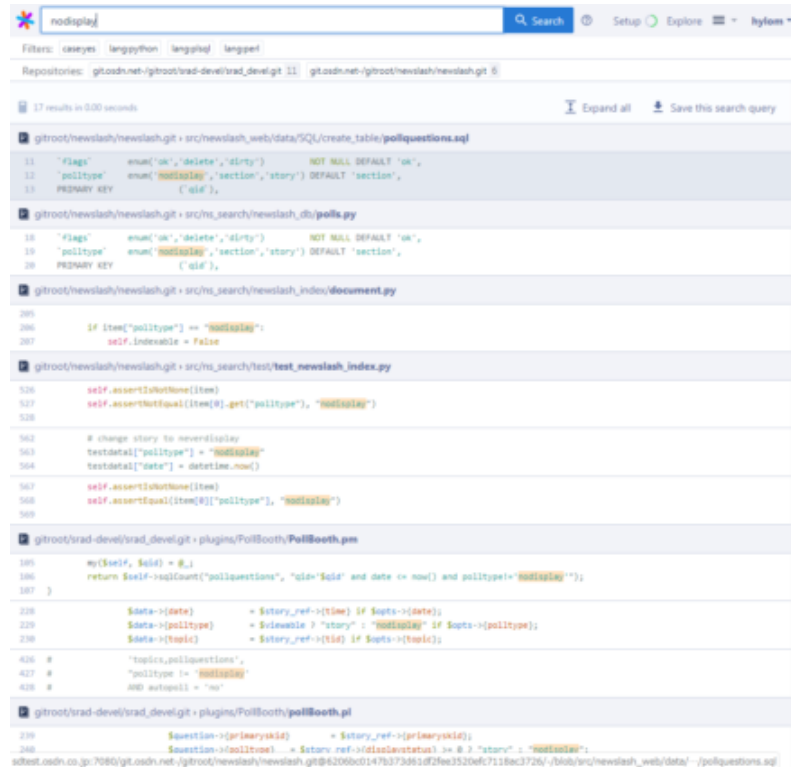


図1 Sourcegraphのソースコード検索画面

Webブラウザ上で操作できるコードブラウザを提供

Sourcegraphは元々は商用サービスとして開発されていたものだが、2018年にオープンソース化された。現在でも商用版 Sourcegraphの提供は行われているが、その主要機能についてはオープンソース版でもそのまま利用できる。

Sourcegraphの特徴としては、以下のような点が挙げられる。

- WebブラウザからGUIで各種設定や検索、コード閲覧が可能
- Gitや各種サービスとの連係が可能
- 高速な検索・インデクシング

まず大きな特徴としては、Webブラウザ上で各種設定や検索、コードの閲覧が完結する点がある。検索対象のソースコードはGitを使って取得する仕組みになっており、たとえば使用するライブラリや連係するサービスの仕様などを調べたい場合、そのGitリポジトリをSourcegraphに登録するだけで手元にそのソースコードを用意することなしにソースコードの閲覧や分析が行える。

検索対象のソースコードを登録する作業や各種設定もGUIで簡単に行えるほか、GitHubやGitLab、Bitbucketといった外部サービスとの連係機能も搭載しており、これらサービス上に登録されているユーザー情報やユーザー権限、リポジトリのパーミッション設定をSourcegraph上で利用することもできる。

また、さまざまな条件を指定して検索を実行することが可能で、たとえばリポジトリ内のファイルだけでなくコミットメッセージも検索対象にできるほか、特定のコミットやタグ、ブランチを指定して検索を実行することも可能だ。複数のリポジトリを対象に横断的に検索を行うこともできる。

Sourcegraphの機能にアクセスするためのAPIも公開されており、これを使えば外部ツールとSourcegraphの連係も可能だ。たとえばコマンドラインからSourcegraphのソースコード検索機能を利用するためのフロントエンドなどが公開されている。また、Sourcegraph自体の機能を拡張するAPIを使った拡張（extensions）も多数公開されている。

なお、Sourcegraphではソースコードを文法的に解析するためのバックエンドとしてLanguage Serverという仕組みを用いている。Language Serverはソースコードを解析してその結果をクライアントに提供するためのサーバーで、クライアントとの通信はLanguage Server Protocol (LSP) として規格化されている。そのため、現在Sourcegraphが対応していないプログラミング言語や構造化マークアップ言語についても、このLSPに対応するLanguage Serverを作成することでSourcegraphからソースコードの解析や検索を行えるようになる。

商用版とオープンソース版の違い

Sourcegraphでは、オープンソース版に加えて商用版の「Enterprise」や「Unlimited」というバージョンも提供されている（[機能および料金体系紹介ページ](#)）。

オープンソース版では登録できるユーザー数が20ユーザーまでに制限されているが、コード検索機能については制限なしに利用できる。Enterprise版やUnlimited版は優先サポートなどが提供されるほか、冗長構成や高度なロギング/モニタリング機能、管理機能などが利用できるようになっている。

なお、有料版ではセルフホスティング（利用者が自分で自前のサーバーにインストールして利用する）形式だけでなく、Sourcegraph社が管理するサーバー上で動作しているSourcegraphインスタンスを利用するSaaS形式でも利用可能なようだ。ただしこの場合、検索対象となるソースコードは一般公開されているGitリポジトリのみになるとのこと。

以下では、オープンソース版のSourcegraphを自前のサーバーにインストールして利用する手順を紹介する。

Sourcegraphの実装と動作環境

オープンソース版のSourcegraphは、[GitHubのsourcegraph/sourcegraphリポジトリ](#)で公開されている。主要コンポーネントはGo言語で実装されており、データベースとして別途PostgreSQLやRedis、フロントエンドとしてnginxなどが必要だ。また、Sourcegraph本体はDockerを利用してコンテナ内で動かすことが想定されており、Dockerを利用できる環境が必須となる。

ソースコードからのインストール作業はドキュメントが公開されているが、依存関係が多くやや大変だ。そのため、データベースやnginx、Redisといった必要となるコンポーネントをすべて単一のコンテナ内に含んだDocker向けのコンテナイメージもDockerHubで提供されている。

Sourcegraphのインストールと基本的な使い方

それでは、Sourcegraphのインストール方法および基本的な設定、使い方について紹介していこう。

Dockerを使ったSourcegraphの起動

前述のとおり、ソースコードからSourcegraphをビルドして利用するのは手間がかかるため、通常は必要なコンポーネントすべてを含んでいるコンテナイメージを利用することをお勧めする。

この場合、設定ファイルなどを格納するディレクトリを用意して次のようにdockerコマンドを実行するだけでSourcegraphサーバーを起動できる。このとき、設定ファイル格納ディレクトリやデータ格納ディレクトリは事前に作成しておく必要がある。

```
# docker run -p 7080:7080 -p 2633:2633 -d --name <コンテナ名> -v <設定ファイルのパス>
```

Page Top

たとえば設定ファイルを/var/sourcegraph/config以下に、データを/var/sourcegraph/data以下に格納し、記事執筆時点の最新版であるSourcegraph 3.4.4を利用してコンテナ名は「sourcegraph」にする場合、次のように実行すれば良い。

```
# docker run -p 7080:7080 -p 2633:2633 -d --name sourcegraph -v /var/sourcegraph/con
```

この場合、Sourcegraphは7080番ポートでWebブラウザからのアクセスを待ち受けるようになるので、Webブラウザで「http://<Sourcegraphが稼動するホスト>:7080/」にアクセスするとSourcegraphの「Welcome」画面が表示されるはずだ。

また、各種ログは標準出力経由でdockerに出力される。これらログを閲覧するにはdocker logsコマンドを実行すれば良い。

```
# docker logs sourcegraph
12:44:42      syntect_server | Configured for production.
12:44:42      postgres | 2019-06-19 12:44:42.216 UTC [28] LOG:  listening on IP
12:44:43 management-console | t=2019-06-19T12:44:43+0000 lvl=info msg="management-co
12:44:44      frontend |
12:44:44      frontend | ??????????????????lwnnnwk
12:44:44      frontend | ??????????????????nnnnnnnnnn
12:44:44      frontend | ??????????????????tnnnnnnnnnnn?????????????lwwwwk
12:44:44      frontend | ??????????????????nnnnnnnnnnnk?????????lwnnnnnnnnk
12:44:44      frontend | ??????????????????mnnnnnnnnnnnn?????????lwnnnnnnnnnnn
12:44:44      frontend | ??????????????????nnnnnnnnnnnk???lwnnnnnnnnnnnj
12:44:44      frontend | ???lwwkk?????????nnnnnnnnnn?ljnnnnnnnnnnj
12:44:44      frontend | ?nnnnnnnnnnkwwk???tnnnnnnnnnjnnnnnnnnnnnnj
12:44:44      frontend | tnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnj
12:44:44      frontend | mnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnv
12:44:44      frontend | ??mvnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnk
12:44:44      frontend | ????????mmvnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnwkk
12:44:44      frontend | ??????????????????mjnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnkwwk
12:44:44      frontend | ??????????????????lwnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnn
12:44:44      frontend | ??????????????????lknnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnn
12:44:44      frontend | ??????????????????lknnnnnnnnnnnnnnnnnnnnnnnnnnnnnn?mjvnnnnnnnnnnnnj
12:44:44      frontend | ??????????lwnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnj
12:44:44      frontend | ??????????wnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnj
12:44:44      frontend | ?????????nnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnk
12:44:44      frontend | ??????mnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnn
12:44:44      frontend | ????????mvnnnnnnjmm?????????nnnnnnnnnnnnnk
12:44:44      frontend | ??????????????????nnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnj
12:44:44      frontend | ??????????????????nnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnnj
12:44:44      frontend |
12:44:44      frontend | ? Sourcegraph is ready at: http://127.0.0.1
```

HTTPSで利用するための設定

外部に公開しているサーバーなどでSourcegraphを運用したい場合など、SourcegraphサーバーへのアクセスにHTTPSを利用したいという場合もあるだろう。その場合、SSL/TLS証明書ファイルを用意して設定ファイルを修正すれば良い。

上記の手順でSourcegraphを起動すると、指定した設定ファイル格納ディレクトリ内にいくつかの設定ファイルが作成される。デフォルトではコンテナ内でnginxをリバースプロキシとして使用してリクエストをSourcegraphサーバーに転送する構成になっているので、設定ファイル格納ディレクトリ内にSSL/TLS証明書をコピーし、nginxの設定ファイルである「nginx.conf」でそのSSL/TLS証明書を利用するように指定すれば良い。

設定ファイル格納ディレクトリが/var/sourcegraph/configの場合、作業手順は以下のようになる。まず、このディレクトリ内に適当なディレクトリを作成してそこにSSL/TLS証明書ファイルと対応する鍵ファイルをコピーする。今回はディレクトリ名を「ssl_certs」、証明書ファイルを「client.pem」、鍵ファイルを「client.key」とした。

```
# cd /var/sourcegraph/config
# mkdir ssl_certs
# cp <証明書ファイル> ssl_certs/client.pem
# cp <鍵ファイル> ssl_certs/client.key
# ls ssl_certs/
client.key  client.pem
```

続いて、nginx.conf内の「http」以下を次のように変更する。太字になっている部分が変更した部分だ。

```
http {
    server_tokens off;

    # Do not remove. The contents of sourcegraph_http.conf can change between
    # versions and may include improvements to the configuration.
    include nginx/sourcegraph_http.conf;

    access_log off;
    upstream backend {
        # Do not remove. The contents of sourcegraph_backend.conf can change
        # between versions and may include improvements to the configuration.
        include nginx/sourcegraph_backend.conf;
    }

    server {
        # Do not remove. The contents of sourcegraph_server.conf can change
        # between versions and may include improvements to the configuration.
        include nginx/sourcegraph_server.conf;

        listen 7080 ssl;
        ssl_certificate      /etc/sourcegraph/ssl_certs/client.pem;
        ssl_certificate_key /etc/sourcegraph/ssl_certs/client.key;
```

```
location / {
    proxy_pass http://backend;
    proxy_set_header Host $http_host;
    proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
    proxy_set_header X-Forwarded-Proto $scheme;
}
}
```

また、Sourcegraphでは「management console」というSourcegraph自体の設定を行うための機能があり、そちらはSourcegraphとは別のポートで待ち受けを行っている（後述）。こちらで使用するSSL証明書は設定ファイルディレクトリ以下のmanagementディレクトリに「cert.pem」および「key.pem」というファイル名で用意されているので、用意したSSL証明書ファイルおよび鍵ファイルでこれらを置き換えておく。

```
# cd /var/sourcegraph/config
# cp ssl_certs/client.pem management/cert.pem
# cp ssl_certs/client.key management/key.pem
```

証明書および鍵ファイルのコピーや設定ファイルの作成が完了したら、コンテナを再起動する。

```
# docker restart sourcegraph
```

これで「https://<ホスト名>:7080/」でSourcegraphにアクセスできるようになるはずだ。もし上手くアクセスできない場合、docker logsコマンドでログを確認してみると良いだろう。

SSHアクセスに使用する公開鍵/秘密鍵の作成

非パブリックなGitリポジトリでは、SSHを使った認証が設定されている場合がある。SSH経由でGitリポジトリにアクセスしたい場合、事前にそのための公開鍵/秘密鍵のペアを作成しておく必要がある。これら公開鍵/秘密鍵ファイルは、設定ファイル格納ディレクトリ以下の「ssh」ディレクトリに格納しておく。たとえば設定ファイル格納ディレクトリが/var/sourcegraph/configだった場合、次のようにして公開鍵/秘密鍵ファイルを作成できる。

```
# cd /var/sourcegraph/config
# mkdir ssh
# ssh-keygen -t rsa -f ssh/id_rsa -N ""
```

ここではパスフレーズが空の鍵ペアを作成しているので、これらファイルの取り扱いには注意しよう。これで（id_rsa）および公開鍵（id_rsa.pub）が作成されるので、公開鍵をGitリポジトリのあるサーバーに登録し

あわせて、同じディレクトリにサーバー側のホスト鍵を記録した「known_host」ファイルも用意する必要があります。例えばGitリポジトリのあるサーバーが「example.com」だった場合、まず次のようにして作成した秘密鍵を使ってSSHでいったんログインし、ホスト鍵の情報を取得しておく。

```
# ssh -i ssh/id_rsa example.com
The authenticity of host 'example.com (XXX.XXX.XXX.XXX)' can't be established.
ECDSA key fingerprint is SHA256:9K7tpuAFYBPZdA0DhEJGTGbZDU63vXHVYdtgf7sEArc.
Are you sure you want to continue connecting (yes/no)? yes yesと入力してEnterキーを押す
Warning: Permanently added 'example.com,XXX.XXX.XXX.XXX' (ECDSA) to the list of known hosts.
```

これでsshコマンドを実行したユーザーの~/ssh/known_hostファイル内にexample.comのホスト鍵が記録されるので、次のように実行してそこからexample.comに対する行のみを出力する。

```
# ssh-keygen -F example.com > ssh/known_host
```

以上の作業が完了すると、sshディレクトリ内に次のように3つのファイルが作成されているはずだ。

```
# ls ssh
id_rsa  id_rsa.pub  known_host
```

最後にsourcegraphコンテナを再起動すれば設定が反映される。

```
# docker restart sourcegraph
```

Sourcegraphの初期設定

Sourcegraphでは基本的にほぼすべての設定をWebブラウザ上で行えるようになっており、まずはそれら設定作業を実行するための管理者アカウントを作成する必要がある。管理者アカウントが作成されていない状態で「http://<ホスト名>:7080/」（もしくは「https://<ホスト名>:7080/」）にWebブラウザでアクセスするとWelcome画面が表示されるので、ここで管理者アカウントの情報を入力する（図2）。

The image shows the Sourcegraph 'Welcome' screen. At the top is the Sourcegraph logo, a stylized 'X' made of four colored lines (blue, orange, green, red). Below the logo is the word 'sourcegraph' in a bold, sans-serif font. Underneath is the heading 'Welcome' followed by the instruction 'Create an admin account to start using Sourcegraph.' There are three input fields: 'Email', 'Username', and 'Password'. Below these is a checkbox labeled 'Try Sourcegraph Enterprise free for 30 days' with a question mark icon. At the bottom is a large blue button that says 'Create admin account & continue'.

図2 Sourcegraphの「Welcome」画面

入力が必要なのはEmailアドレスとユーザー名、パスワードの3つだ。これらを入力して「Create admin account & continue」をクリックするとアカウントが作成される。アカウント作成後にはサイト管理（Site admin）画面が表示されるので、そこで各種設定を行っていくことになる（図3）。

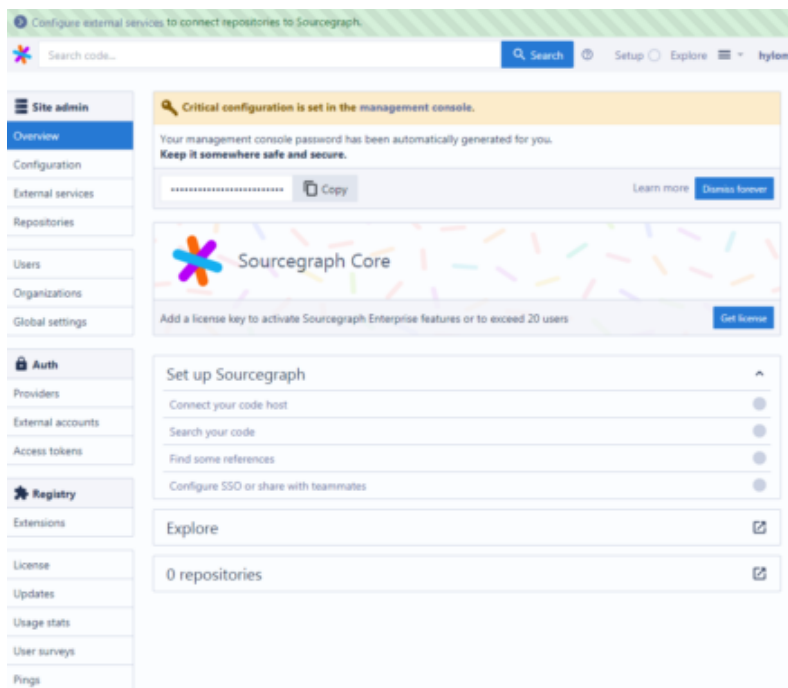


図3 Sourcegraphのサイト管理画面

ちなみにこのSite admin画面には画面右上のユーザーメニュー内にある「Site admin」リンクから移動できアウトなどもこのメニューから実行可能だ。

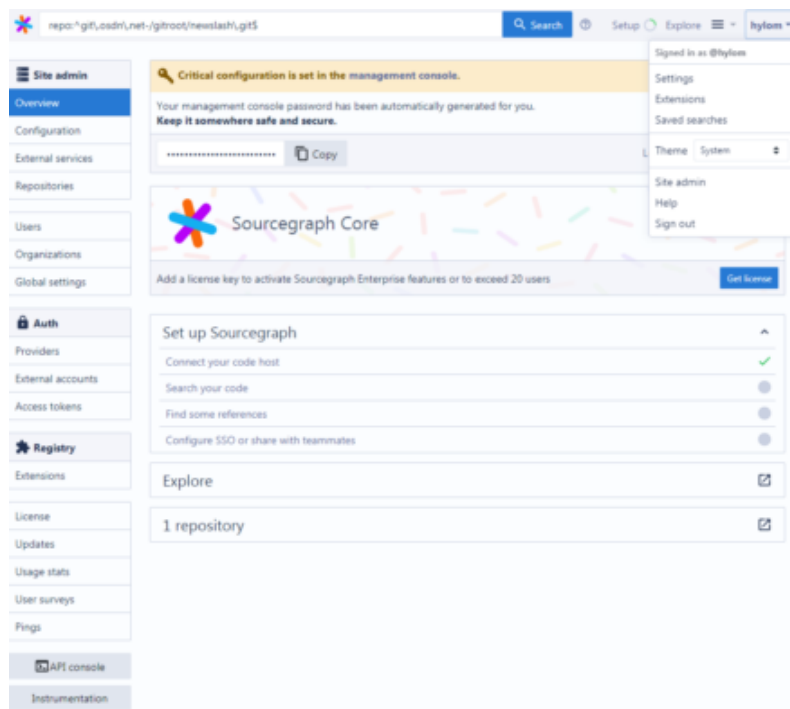


図4 画面右上のユーザーメニューでSite admin画面に遷移できる

サインアップの無効化

管理者アカウントの作成後にログインしていない状態でSourcegraphサーバーにアクセスすると、次のようなサインイン画面が表示される（図5）。

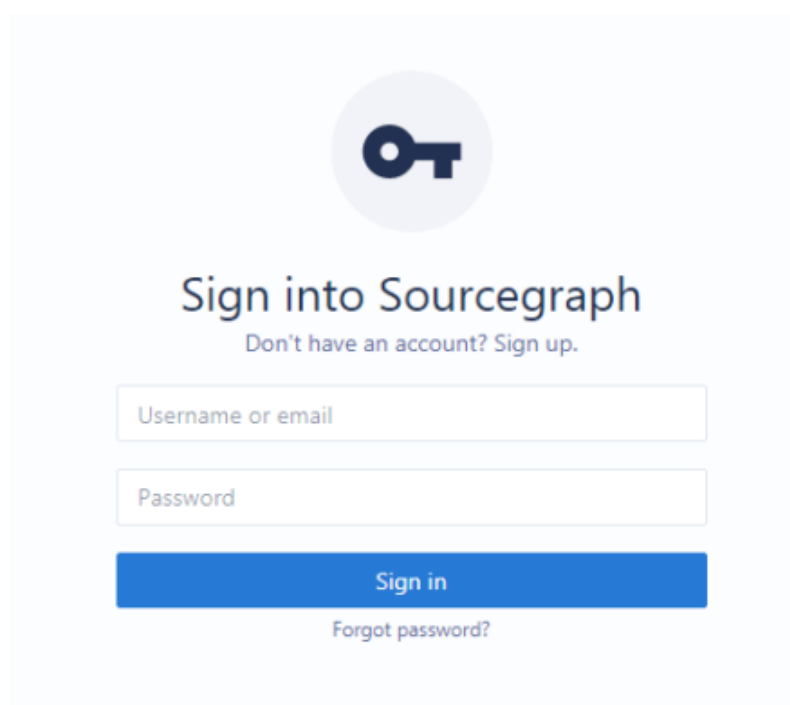


図5 Sourcegraphのサインイン画面

ここで表示されている「Sign up」をクリックすると、新規ユーザーアカウントを作成できる画面が表示される。認証・承認作業などは行われないため、誰もがアカウントを作成してSourcegraphを利用できてしまう。その

がアクセスできるようなサーバーではサインイン画面からのサインアップを無効にしておこう。サインイン画面、[サインアップ画面](#)、[Sourcegraphの管理画面](#)ではなく、別途用意されている「management console」で行う。

management consoleにアクセスするには、Webブラウザで「https:// <Sourcegraphを稼働させているホスト> :2633」にアクセスする。アクセス時にはユーザー名とパスワードが要求されるが、ユーザー名にはSourcegraphの管理者アカウントのユーザー名を入力し、パスワードにはSite admin画面内にある「management console password」をコピーして入力する（図6）。

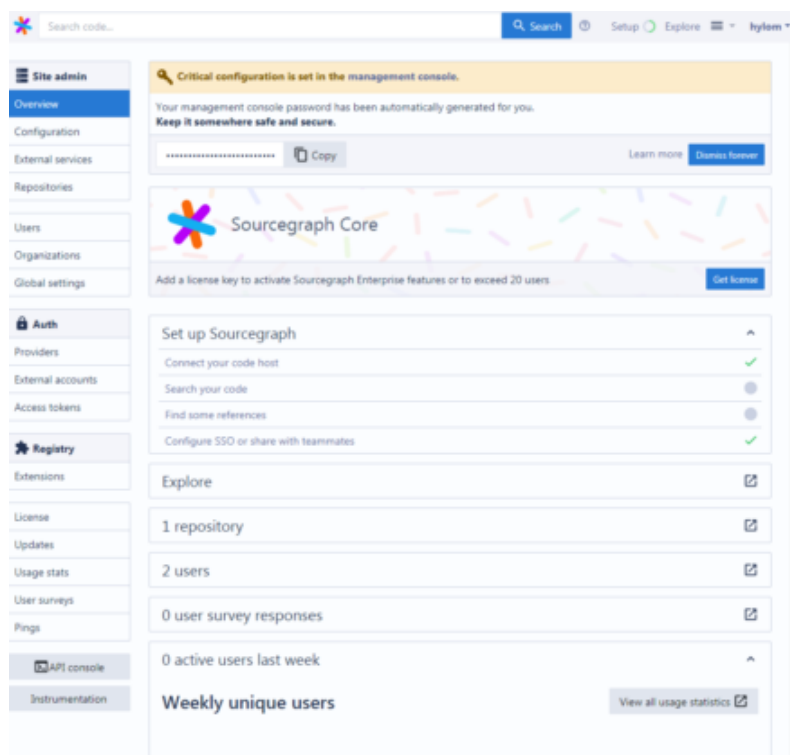


図6 management consoleにアクセスする際に使用するパスワードはSourcegraphのSite admin画面からコピーできる

Sourcegraph management consoleでは設定ファイルを直接書き換えることで各種設定を行う仕組みになっている（図7）。サインアップを無効にするには、「auth.providers」設定内の「allowSignup」の値を「true」から「false」に変更して「save changes」をクリックすれば良い。

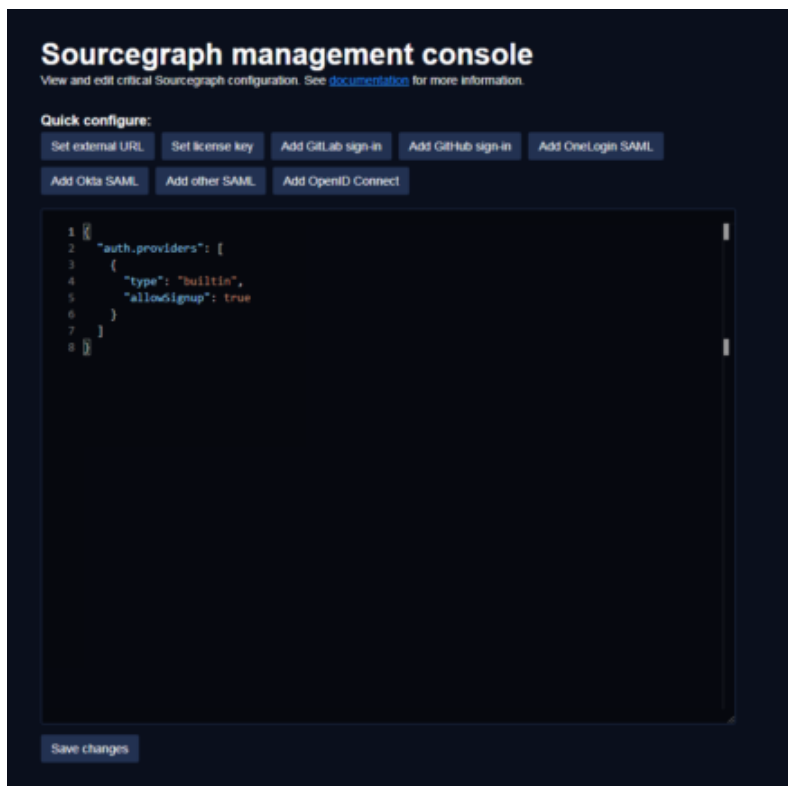


図7 「allowSignup」部分を「false」に書き換えることでサインアップを無効にできる

また、ユーザーを追加したい場合はSite admin画面の「Users」画面から実行できる（図8）。

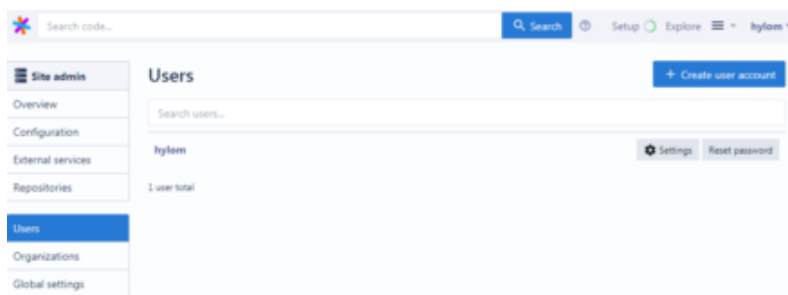


図8 「Users」画面でユーザーの管理を行える

ちなみに、Site admin画面におけるmanagement consoleパスワード表示部分の右端にある「Dismiss forever」ボタンをクリックしてしまうと、この表示は消えてしまう。再度このパスワードを取得したい場合はやや面倒な手順が必要なので注意したい。その場合の復旧作業については[ドキュメント](#)を参照してほしい。

リポジトリの登録

Sourcegraphでは、検索対象とするGitリポジトリをあらかじめ登録しておく必要がある。リポジトリの管理は、「External services」画面で行える（図9）。

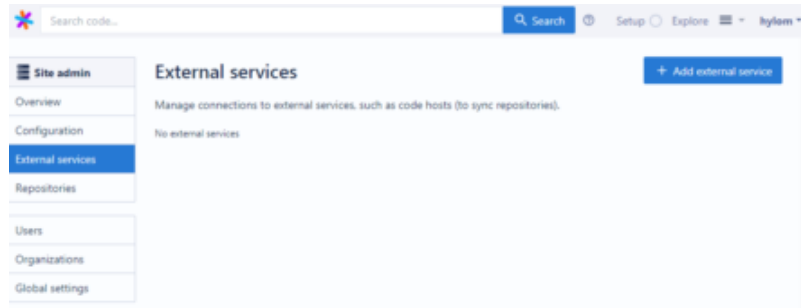


図9 「External services」画面

この画面では登録されているリポジトリの一覧が表示されるが、初期状態ではリポジトリが登録されていないため何も表示されない。リポジトリを追加するには画面右上の「Add external service」をクリックし、表示される「Add external service」画面で追加したいリポジトリの種別を選択する（図10）。

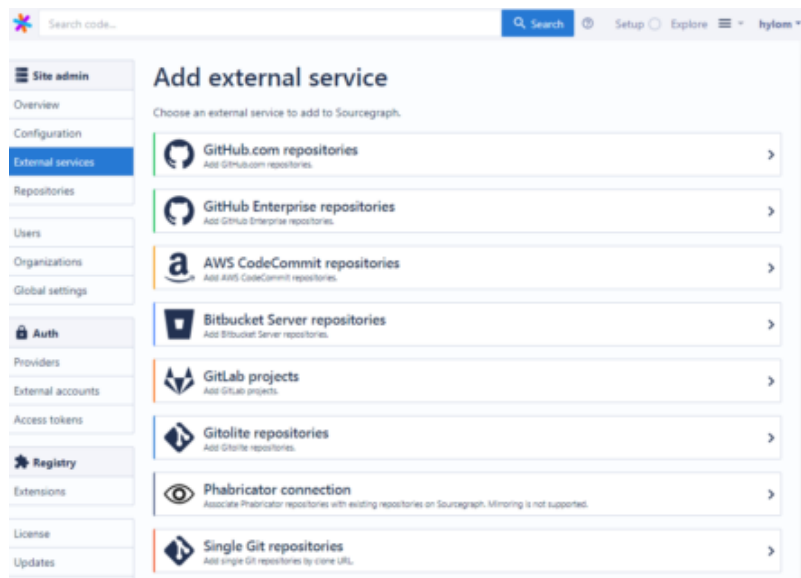


図10 「Add external service」画面

SourcegraphではGitHubやBitbucket、GitLabといった外部サービス上のリポジトリを登録できるほか、任意のGitリポジトリをクローンして検索対象にすることもできる。

Gitリポジトリを登録したい場合は、ここで「Single Git repositories」をクリックする。JSON形式で記述された設定情報を編集するエディタ画面が開くので、「Display name」部分に適当な表示名、「url」部分にリポジトリのURL、「repos」部分にリポジトリ名を入力する（図11）。なお、「repos」はリスト形式で複数を指定できる。

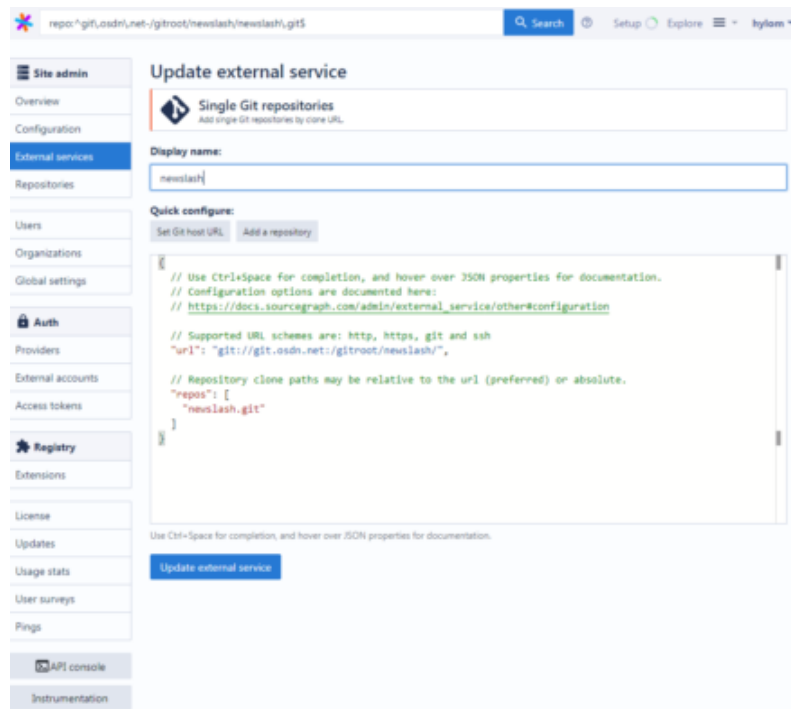


図11 「Add external service」画面

このリポジトリ指定方法がやや分かりにくいのだが、基本的にはリポジトリのURLのうち最後の「/」から後ろの部分を「repos」に、その前の部分を「url」に記述する。たとえばリポジトリにアクセスするためのURLが「https://scm.osdn.net/gitroot/newslash/newslash.git」の場合、「url」は「https://scm.osdn.net/gitroot/newslash/」、「repos」は「newslash.git」となる。リポジトリへのアクセス時にユーザー名やパスワードによる認証が必要な場合は、「url」で「https://<ユーザー名>:<パスワード>@<ホスト名>/<パス名>」のように指定すれば良い。

また、Gitでは「<ユーザー名>@<ホスト名>:<パス>」といったURLでSSH経由でリポジトリにアクセスできるが、この場合は「git://<ホスト名>:<パス>」のように、明示的に「git://」で始まるURLを入力する必要がある。このときユーザー名を指定することはできず、ユーザー認証は先の設定時に登録しておいたSSH鍵ベースで行われる。

最後に「Update external service」をクリックするとリポジトリ情報が保存され、External services画面にその情報が追加される。また、登録されているリポジトリの一覧は、「Repositories」画面で確認できる（図12）。

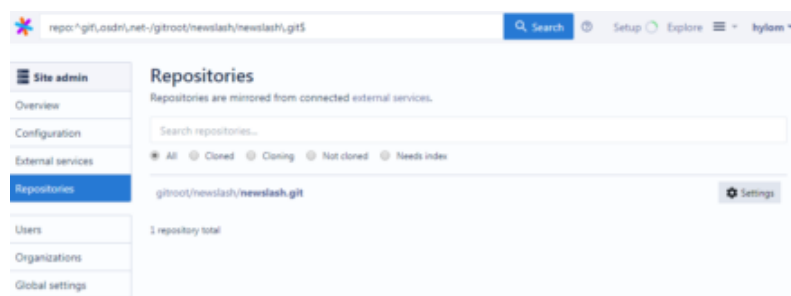


図12 「Repositories」画面

ここでリポジトリ横に表示されている「Settings」ボタンをクリックすると、リポジトリに関する情報を確認できる。

「Indexing」画面では検索インデックスの作成状況を、「Mirroring」画面ではリポジトリのクローン状況を確認（図13、14）。

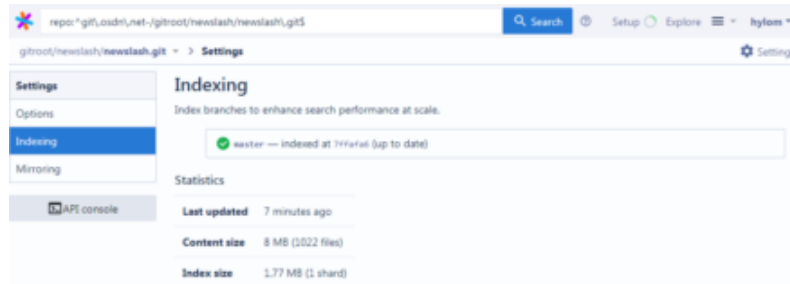


図13 「Indexing」画面では検索インデックスの作成状況を確認できる

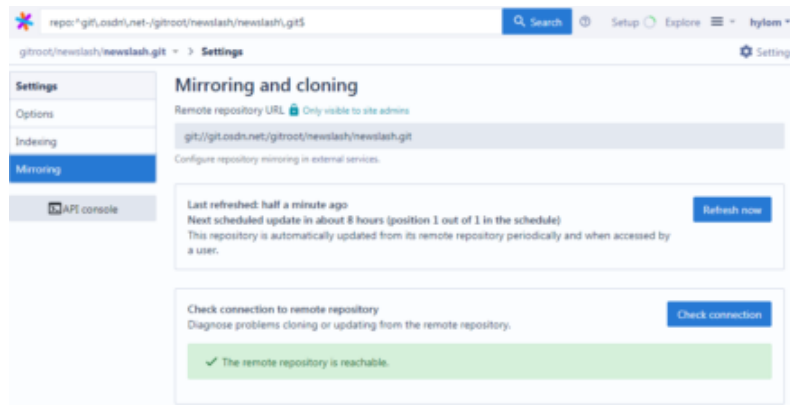


図14 「Mirroring」画面ではリポジトリのクローン状況を確認できる

なお、設定が間違っていたり、認証に失敗したりといった理由でリポジトリをクローンできない場合はこのMirroring画面でエラーメッセージを確認できる。

GitHub上のリポジトリを登録する

GitHub上のリポジトリを登録する場合、GitHubとの連携機能を利用することで特定のユーザーアカウントなどに紐付けられたリポジトリをまとめて登録できる。

GitHubとの連携にはGitHubにアクセスするためのアクセストークンが必要となる。まずGitHubにログインし、「Personal Settings」内にある「Developer settings」をクリックし、続いて「Personal access tokens」をクリックする（図15）。

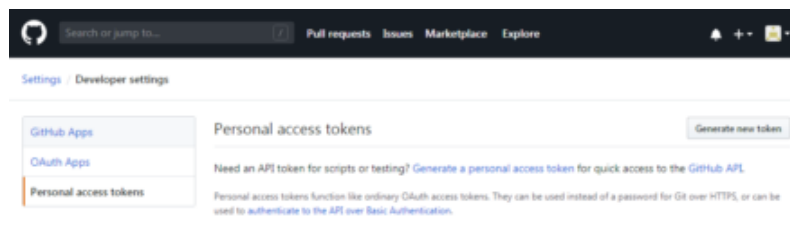


図15 「Personal access tokens」画面

ここで「Generate new token」をクリックするとトークンの作成画面が表示されるので、「Note」欄に適当し、画面下に表示されている「Generate token」ボタンをクリックする（図16）。ここではトークンに対してもできるが、特に設定する必要はない。

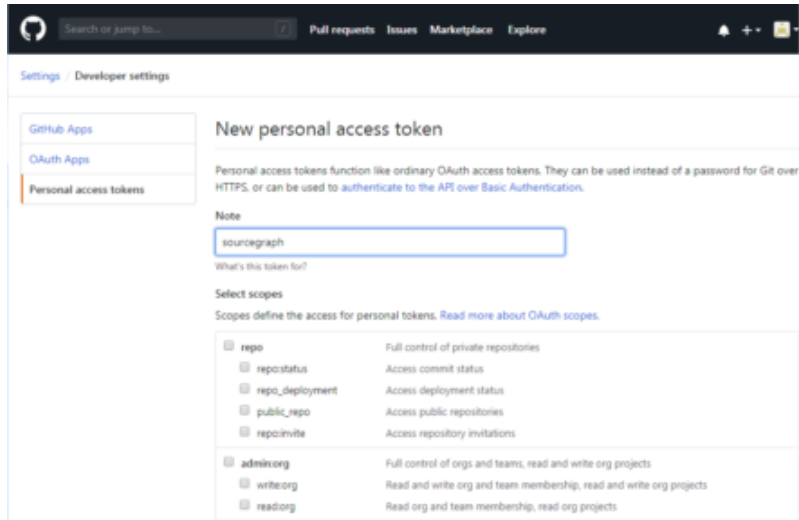


図16 「New personal access token」画面

するとトークン文字列が表示されるので、これをコピーしておく（図17）。作成されたトークンはこの作成直後の画面でしか確認できず、再度トークンが必要となった場合は再生成するしかないため注意しておこう。

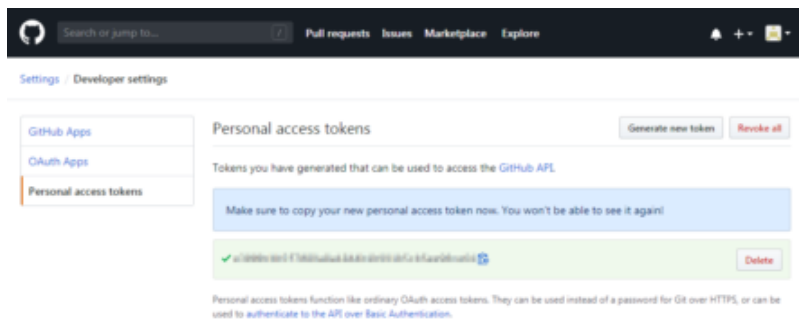


図17 トークンが表示される

トークンを準備したら、「Add external service」画面で「GitHub.com repositories」を選択し、設定ファイル内の「token」部分にコピーしたアクセストークンを入力する（図18）。

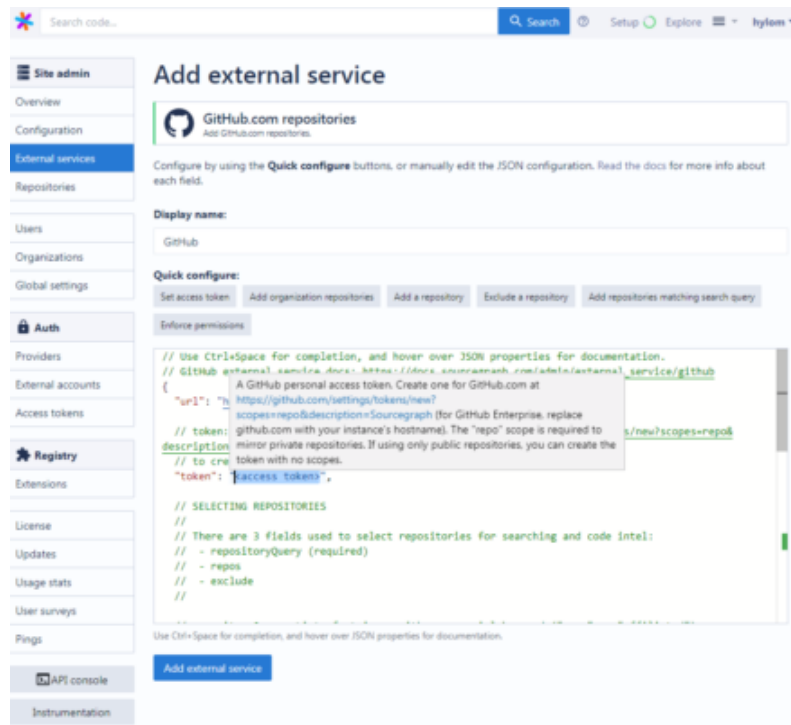


図18 「token」部分にコピーしたアクセストークンを入力する

次に、「repositoryQuery」でどのリポジトリを検索対象としてインデクシングするかを指定する。ユーザーに紐付けられているすべてのリポジトリを対象とする場合は「affiliated」と指定する。また、特定のリポジトリのみを対象としたい場合は「none」を指定し、続けて「repos」という設定項目内で対象とするリポジトリを「<ユーザー名>/<リポジトリ名>」の形で入力する（図19）。

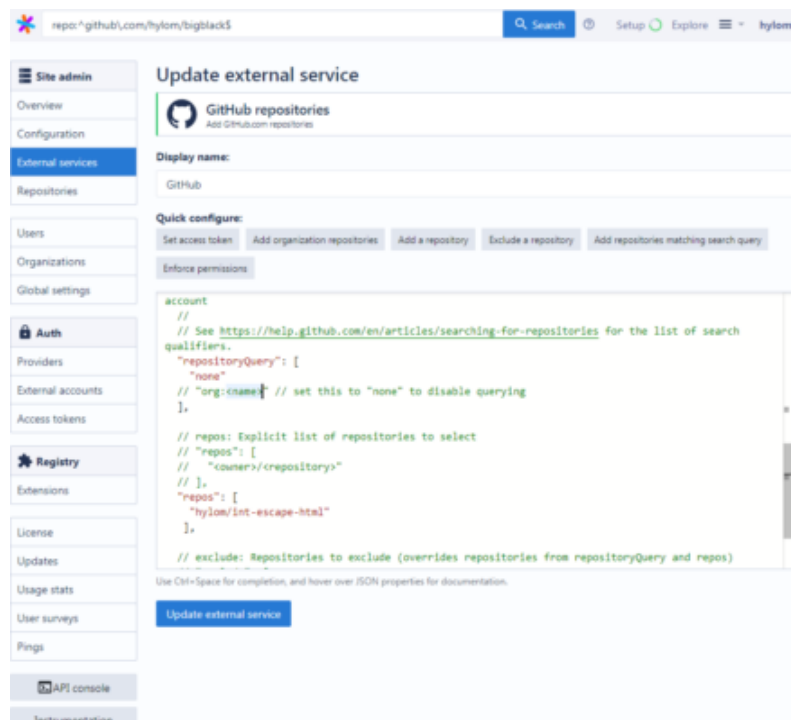
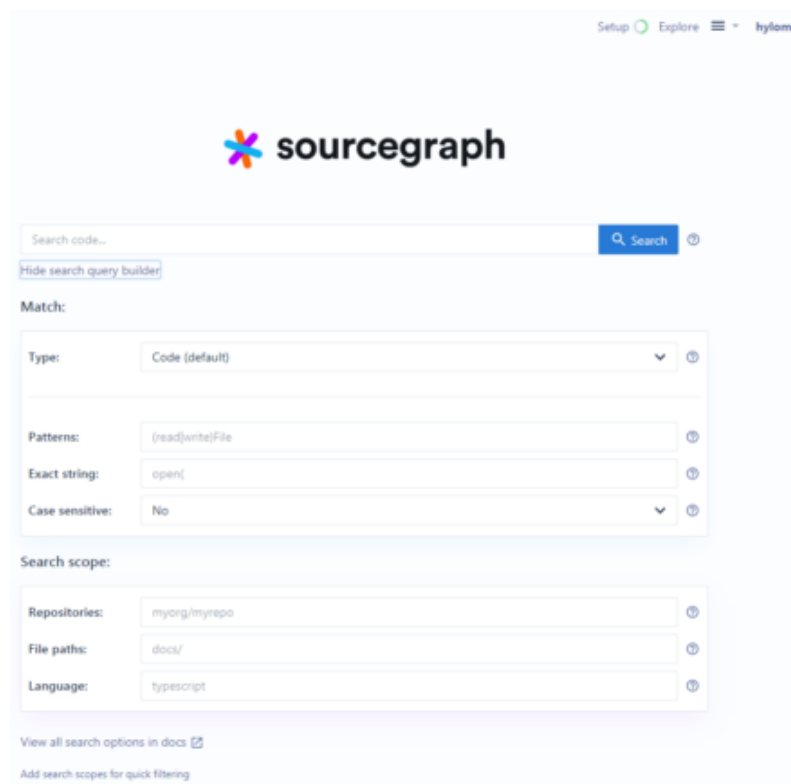


図19 特定のリポジトリのみを検索対象としたい場合は「repos」設定を追加する

リポジトリ内の検索を行う

Sourcegraphでは、画面上部の検索バーやトップ画面からキーワードを入力して登録したリポジトリ内を検索できる。トップ画面（/search）ではフォームからオプションを指定して検索を実行することも可能だ（図20）。



The screenshot shows the Sourcegraph search interface. At the top, there's a search bar with the placeholder text "Search code..." and a "Search" button. Below the search bar, there's a "Hide search query builder" link. The "Match:" section contains a "Type:" dropdown set to "Code (default)", a "Patterns:" text input with "(read|write)File", an "Exact string:" text input with "open(", and a "Case sensitive:" dropdown set to "No". The "Search scope:" section contains a "Repositories:" text input with "myorg/myrepo", a "File paths:" text input with "docs/", and a "Language:" text input with "typescript". At the bottom, there are links for "View all search options in docs" and "Add search scopes for quick filtering".

図20 Sourcegraphのトップ画面

検索を実行すると、検索結果としてファイル名とともに指定したキーワードが登場する部分が抜粋表示される（図21）。

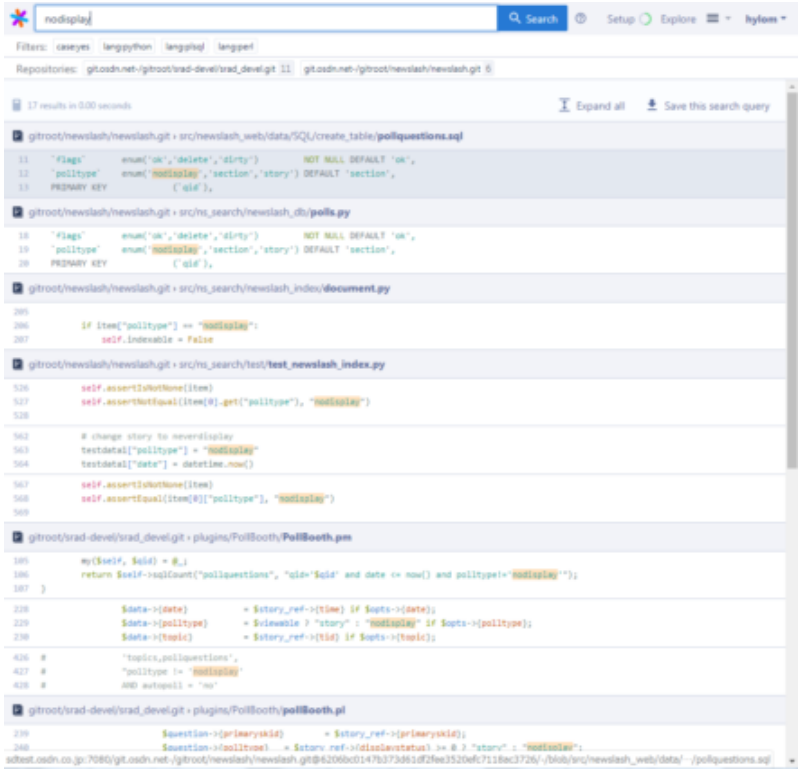


図21 検索結果一覧画面

この画面上部には絞り込みを行うための「Filters:」や特定のリポジトリのみの結果を表示する「Repositories:」といった項目が用意されている。たとえばこの例の場合、Filters:で「lang:python」をクリックすると検索結果のうちPythonコードのみが表示されるようになる（図22）。

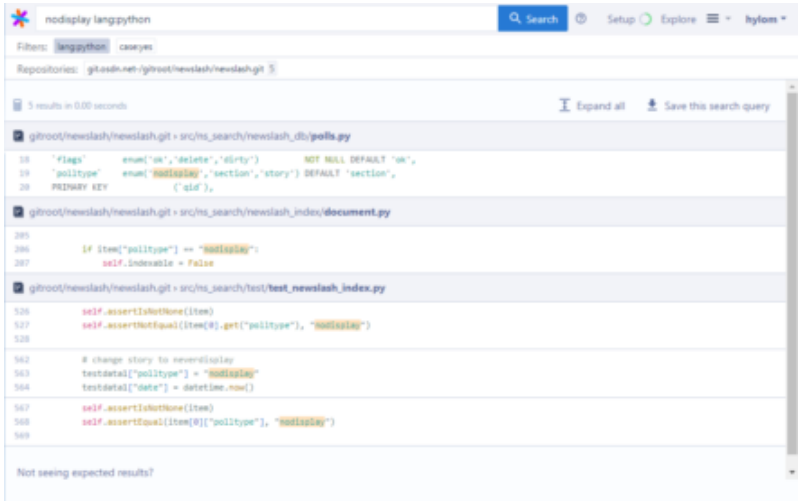


図22 画面上部の「Filters:」や「Repositories:」で検索結果の絞り込みを行える

また、検索結果中のファイル名をクリックするとそのファイルの中身を閲覧できる（図23）。

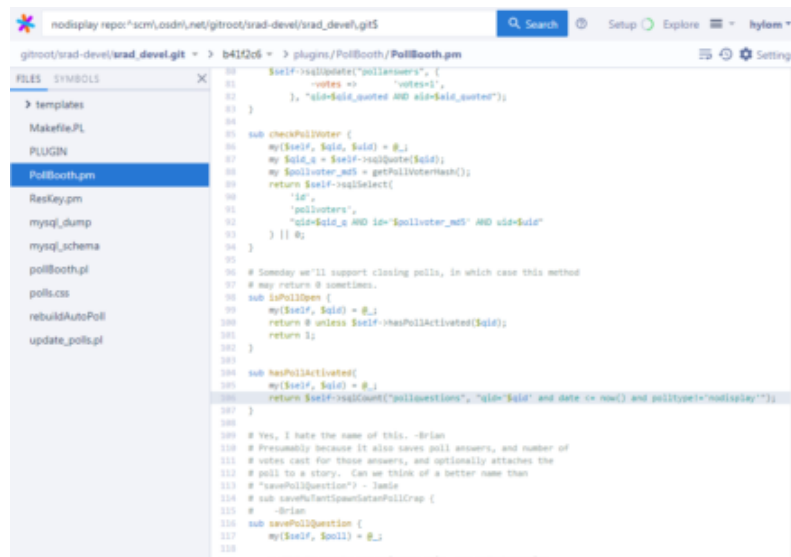


図23 ファイル全文を表示することも可能

この画面ではリポジトリ内のファイルを自由に閲覧できる。ブランチやタグ、コミットを指定してファイルを閲覧することも可能だ（図24）。

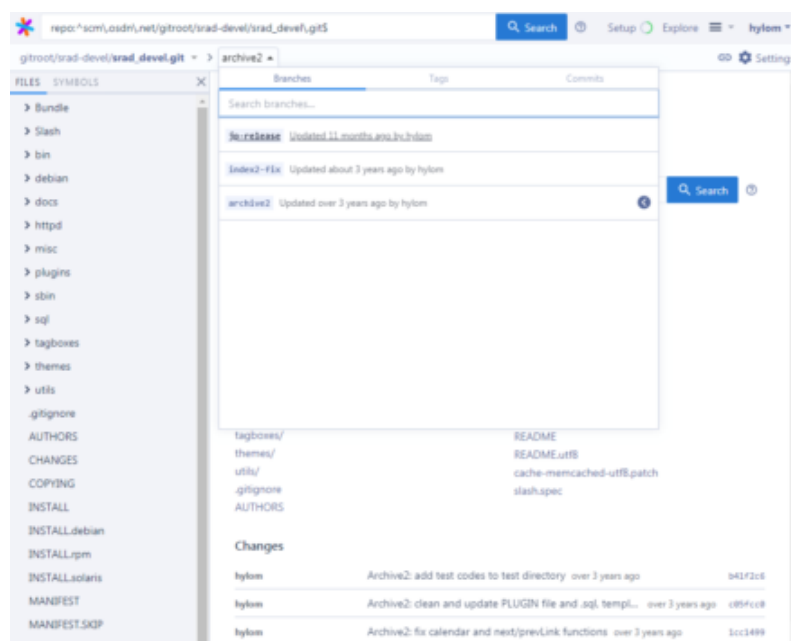


図24 ブランチやタグ、コミットを指定してファイルを閲覧できる

特定のファイルを開いた状態で画面左側に表示されるサイドバー内の「SYMBOLS」をクリックすると、そのファイル中で定義されているクラスや関数、変数といったシンボル一覧が表示される（図25）。

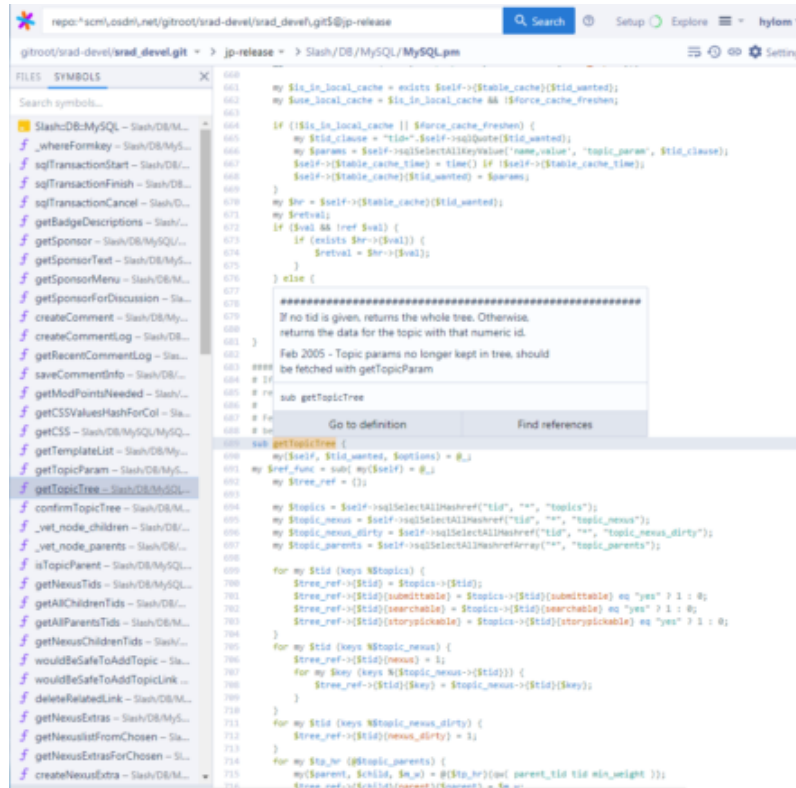


図25 画面左のサイドバーで「SYMBOLS」を選択するとシンボル一覧を表示できる

ここでシンボルをクリックすると、コード内でそのシンボルが定義されている部分が表示される。また、コード内でシンボルをクリックするとそのシンボルに対するコメントが表示される(図26)。

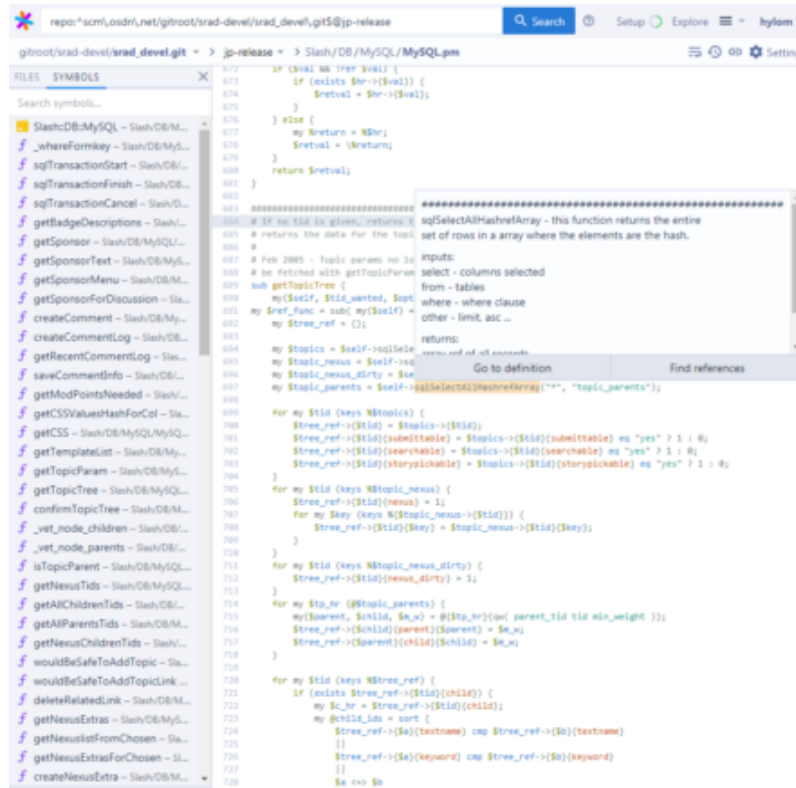


図26 ソースコード中のシンボルをクリックするとそのシンボルに対するコメントが表示される

2019/8/7 Gitリポジトリ上のソースコードをWebブラウザから検索・分析できるコード検索ツール「Sourcegraph」 | さくらのナレッジ

「Go to definition」をクリックするとその定義部分にジャンプできる。また、「Find references」をクリックすると画面下にシンボルを参照している部分が一覧表示される（図27）。

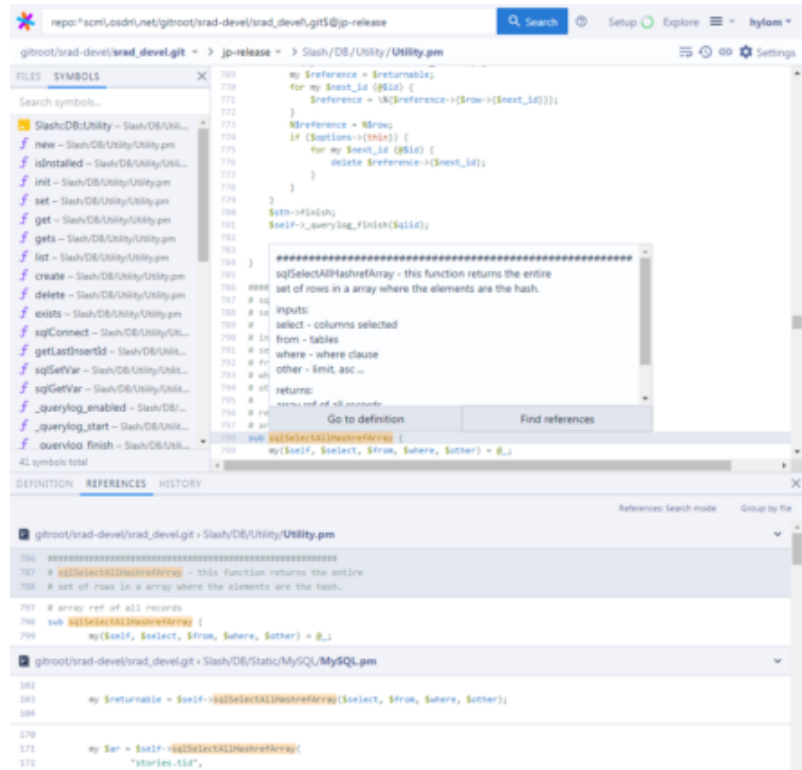


図27 画面下の「REFERENCES」でシンボルを参照しているコードを確認できる

画面下の「DEFINITION」ではシンボルの定義部分を、「HISTORY」ではそのファイルに対する変更履歴を確認できる（図28）。

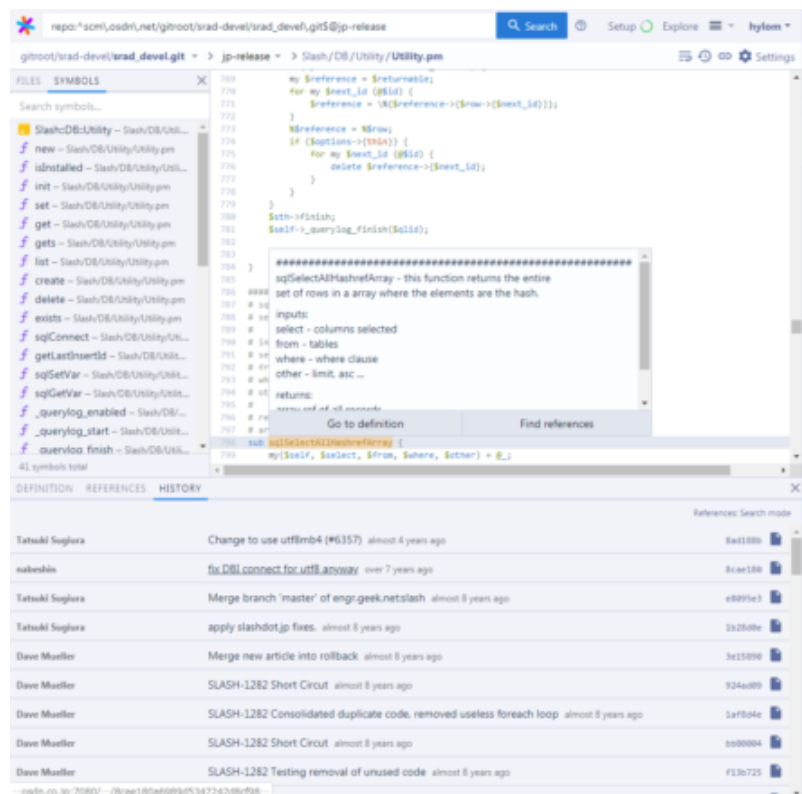


図28 「HISTORY」でそのファイルに対する変更履歴を確認できる

Sourcegraphのブラウザ拡張

SourcegraphではGoogle ChromeおよびFirefoxで利用可能なブラウザ拡張が用意されている。このブラウザ拡張はWebブラウザでGitHubやGitHub Enterprise、GitLab、Phabricator、Bitbucket Server上にあるソースコードなどを閲覧している際に、シンボルに関する情報をポップアップ表示したり、シンボルの定義位置やシンボルを参照している部分を検索したりするといった機能を提供するものだ。この拡張機能は、[公式ドキュメントのブラウザ拡張ページ](#)からインストールできる。

ちなみに、Sourcegraphではsourcegraph.comというサイトでGitHub上にある公開リポジトリを対象とした検索サービスが提供されており、このブラウザ拡張をインストールするだけですべての公開リポジトリ上でSourcegraphの機能を利用できるようになる（図29、30）。

```

122
123 // settings
124 var app = this.app;
125
126 // allow status / body
127 if (arguments.length === 2) {
128 // res.send(body, status) backwards compat
129 if (typeof arguments[0] !== 'number' && typeof arguments[1] === 'number') {
130   var deprecate = any;
131   res.status(status).send(body) instead";
132 }
133 }
134 deprecate('res.send(status, body): Use res.status(status).send(body) instead');
135 this.statusCode = arguments[0];
136 chunk = arguments[1];
137 }
138 }

```

図29 GitHubの公開リポジトリでは特に設定せずとも拡張を導入するだけでSourcegraphの機能が利用できる

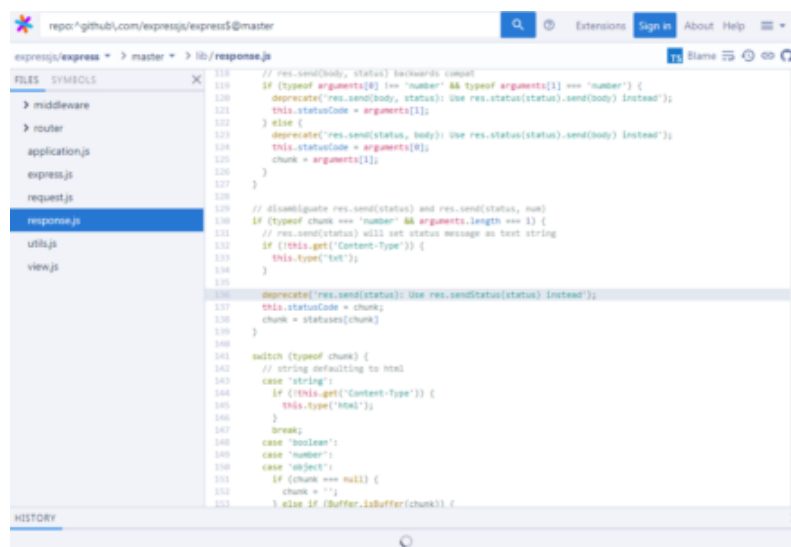


図30 sourcegraph.com上で任意のソースコードを閲覧することも可能

非公開リポジトリを対象にこの機能を利用したい場合は、独自に用意したSourcegraphサーバー上でそのリポジトリをホストし、拡張機能のオプション設定でそのサーバーのURLを設定すれば良い（図31、32）。

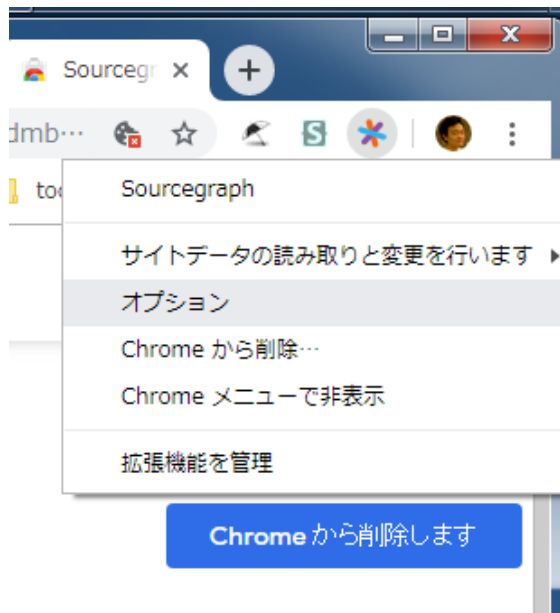


図31 拡張をインストールするとツールバーにアイコンが表示され、そこからオプション設定にアクセスできる

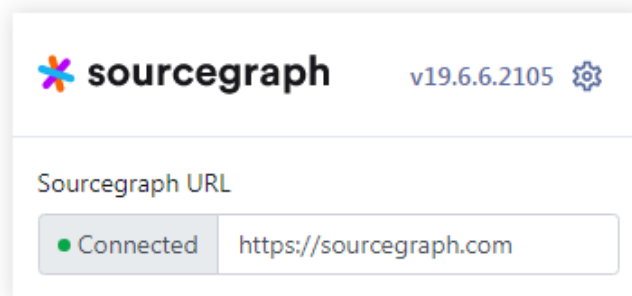


図32 デフォルトではサーバーURLとして「https://sourcegraph.com」が設定されているが、これを独自に用意したサーバーのURLに置き換えることで非公開リポジトリに対しても拡張を利用できるようになる

また、同様にテキストエディタからSourcegraphの検索などを実行することを可能にするエディタ統合機能も提供されている（[公式ドキュメントの「Editor integrations」ページ](#)）。現時点での対応エディタはVisual Studio CodeおよびAtom、IntelliJ、Sublime Textとなっている。

そのほか、コマンドラインで検索を実行する「[Sourcegraph CLI \(src-cli\)](#)」というツールもある。こちらはコマンドラインベースで作業を行っている際に有用だろう。

Sourcegraph自体を拡張するExtensions機能

Sourcegraphでは、Sourcegraphの機能自体を拡張する「Extensions」という仕組みも用意されている。公開されているExtensions一覧は右上のユーザーメニュー内にある「Extensions」からアクセスできる「Extensions」ページで確認できる（[図33](#)）。

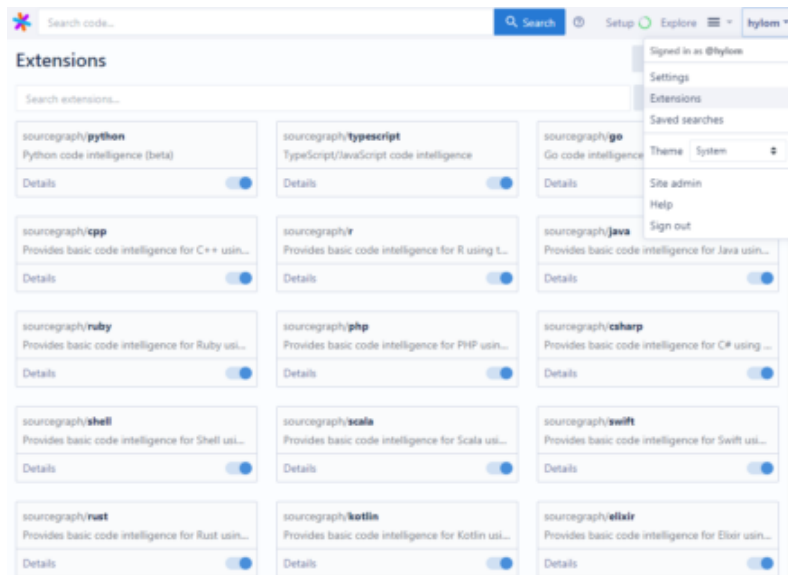


図33 「Extensions」 ページ

Sourcegraphではさまざまなプログラミング言語に対するサポートもこのExtensionsの仕組みを使って実装されており、デフォルトで主要な言語に対するExtensionsが有効になっている。それ以外のExtensionを利用するには、利用したいExtensionを選んでスライドバー部分をクリックすれば良い（青色になっているものが有効化されている拡張）。

ただし、「WIP」（Word in Progress）と表示されているものはまだ正式リリースされていないものなので、有効化する際には注意したい（図34）。

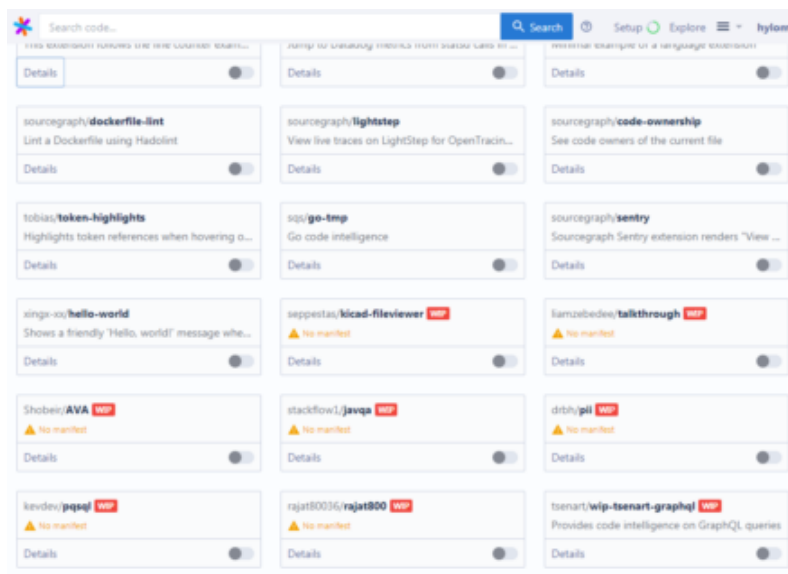


図34 「WIP」と表示されているものは正式版ではないため注意したい

Webブラウザ上でのソースコード検索は意外に便利

前述のようにVisual Studioのようにソースコードの検索機能を備える統合開発環境もあるが、それとは別のワークスペースにクローンしていないコードを検索できるのは意外に有用だ。たとえば別のブランチのコードを検索し、コミットを対象に検索したりできる機能などは開発作業に役立つだろう。

2019/8/7 Gitリポジトリ上のソースコードをWebブラウザから検索・分析できるコード検索ツール「Sourcegraph」 | さくらのナレッジ

また、公開されているGitリポジトリであれば簡単に検索対象にそれを追加できるため、たとえばオープンソースで公開されているライブラリのサンプルコードやそのライブラリを使っている別のソフトウェアから特定の関数やクラスの使い方を調べるなど、自分が作業していないコードに対して検索を行う際にも便利だ。

Dockerが動作する環境さえあれば比較的導入も容易なので、大規模なソースコードを扱っているような現場では是非導入してみてはいかがでしょうか。

💡 Linux , サーバー管理 , ソフトウェア , オープンソース , 開発ツール , コラボレーション

Tweet

シェア 43

68



[このサイトについて](#)

[企業情報](#)

[さくらのクラウドニュース](#)

[さくらのVPSニュース](#)

© SAKURA Internet Inc.