

# Как правильно употреблять C2H5OH

<https://github.com/genosse/c2h5oh>  
[misha@genosse.org](mailto:misha@genosse.org)

# “Пейте спирт, не пейте водку” или “что такое C2H5OH”

C2H5OH – это расширение веб-сервера  
NGINX для простой реализации  
высокопроизводительных HTTP API  
к приложениям на базе PostgreSQL

# Спирт поможет тому:

- кто делает веб-приложения
- кто устал от разных фреймворков
- кто хочет сделать веб-разработку простой
- кто не любит ORM и любит PostgreSQL
- кто не хочет усложнять команду

# Почему NGINX?

- надёжность и стабильность
- работает очень быстро
- множество расширений, можно делать свои
- HTTP/2, IPV6, WebSockets, SSL, etc
- свободное ПО

# Почему PostgreSQL?

- надёжность и стабильность
- работает быстро
- хранимки на любых языках
- можно делать свои расширения
- поддержка json
- свободное ПО

# Почему C2H5OH?

- Берём лучшее: NGINX и PostgreSQL
- Минимализм
- Быстродействие, 14k rps на моём “ноуте”
- За данные и их обработку отвечает один инженер
- Безопасность и масштабируемость
- Всем нужны API

# Итак, как ГОТОВИТЬ С2Н5ОН?

Самогонный аппарат не понадобится!

```
# example for ubuntu 14.04

# install cmake
sudo apt-get install software-properties-common
sudo add-apt-repository -y ppa:george-edison55/cmake-3.x
sudo apt-get update
sudo apt-get install cmake

# install libraries
sudo apt-get install libboost-all-dev

# clone repo from bloody github
git clone https://github.com/genosse/c2h5oh

# make it
cd c2h5oh
make debug && make test && make deb

# install, configure and restart
sudo ./deploy/c2h5oh-nginx_0.0.0_amd64.sh
sudo vi /etc/c2h5oh/c2h5oh_nginx.conf
sudo service c2h5oh_nginx restart
```

# Пример конфигурации nginx

```
# /etc/c2h5oh/c2h5oh_nginx.conf
server {

    listen 8080;
    server_name localhost;
    location /api {

        # db connection string, pool size
        c2h5oh_pass "dbname=c2h5oh user=web password=web" 200;

        # api root
        c2h5oh_root /api;

        # route function
        c2h5oh_route route;

        # timeout
        c2h5oh_timeout 1s;
    }
}
```



# Пример, маршрутизация url

```
create or replace function web.route(url varchar, cookies jsonb, query jsonb)
  returns text as
$$
declare
  res_ jsonb;
begin

  execute 'select web.' ||
    quote_ident(replace(regex_replace(url, '^/(?.*?)/?$', '\1'), '/', '_'))
    || '($1,$2)' into res_ using cookies, query;
  return res_;

exception
  when undefined_function then
    raise warning '%', sqlerrm;
    return '{"status":404}';
  when others then
    return '{"content":{"status":"' || sqlerrm || '"}}';
end;
$$ language plpgsql security definer;
```

# Пример функции – сообразим на троих

```
create or replace function web.na_troih(cookies jsonb, query jsonb)
  returns text as
$$
-- @brief Returns number divided by 3
begin

  return json_build_object('content', json_build_object(
    'status', 'ok', 'na_troih_po', (q->>'babki')::numeric / 3.0);

end;
$$ language plpgsql;

curl 'http://localhost:8080/api/na/troih/?babki=2.87'

{
  'status': 'ok',
  'na_troih_po' : 0.95666666
}
```

# Пример функции – сумма чисел

```
create or replace function web.sum(cookies jsonb, query jsonb)
  returns text as
  $$
  -- @brief Returns a + b
  begin
    return json_build_object('content', json_build_object(
      'status', 'ok', 'sum', (q->>'a')::numeric + (q->>'b')::numeric));
  end;
  $$ language plpgsql;
```

```
curl 'http://localhost:8080/api/sum/?a=10&b=20'
```

```
{
  'status': 'ok',
  'sum'    : 30
}
```

# Пример функции – get\_cookie, set\_cookie

```
create or replace function web.cookie_set(c jsonb, q jsonb) returns text as
$$
begin
    return
        json_build_object('content', json_build_object('status', 'ok'),
            'headers',
                'Set-Cookie: sid=' || (q->>'v') || '; Domain= .genosse.org; Expires=' ||
                    to_char('2016-01-01'::timestamp + interval '14 days',
                        'Dy, DD-Mon-YYYY HH24:MI:SS GMT') ||
                    '; Path=/; Secure; HttpOnly');
end; $$ language plpgsql;
```

```
create or replace function web.cookie_get(c jsonb, q jsonb) returns text as
$$
begin
    return json_build_object('content', json_build_object(
        'status', 'ok', 'sid', c->>'sid'));
end;
$$ language plpgsql;
```

```
curl 'http://localhost:8080/api/cookie/set/?v=cookie_value'
```

```
curl 'http://localhost:8080/api/cookie/get/'
```

# Пример функции – redirect

```
create or replace function web.redirect(cookies jsonb, query jsonb)
    returns text as
$$
-- @brief Returns redirect to specified url
begin

    return json_build_object('content', 'Redirected to: ' || (query->>'u'),
        'status', 302, 'headers', 'Location: ' || (query->>'u'));

end;
$$ language plpgsql;
```

# Пример функции – заголовки http

```
create or replace function web.headers(c jsonb, q jsonb)
  returns text as
  $$
  -- @brief Returns redirect to specified url

  begin
    return json_build_object('content', 'ok',
      'headers', json_build_array(
        'Header1: ' || (q->>'h1'),
        'Header2: ' || (q->>'h2'),
        'Content-Type: text/plain'
      )
    );
  end;
  $$ language plpgsql;
```

# Пример – загрузка файлов

```
location = /api/upload/ {
    client_max_body_size 16m;
    auth_request /auth;
    limit_except POST { deny all; }
    client_body_in_file_only on;
    client_body_temp_path /var/lib/c2h5oh/uploads;
    proxy_pass $scheme://127.0.0.1:$server_port/dummy;
    proxy_set_header          X-FILE $request_body_file;
    proxy_set_header          X-FILE-SIZE $content_length;
    proxy_set_body             off;
    proxy_redirect             off;
}

location = /dummy {
    client_max_body_size 16m;
    allow    127.0.0.1;
    deny    all;
    access_log off;
    if ($http_x_file ~ "/(\w+)$") {
        set $file_name $1;
    }
    return 200 '{"name":"$file_name","size":"$http_x_file_size"}';
}
```

# Задача: отправка e-mail

- Индекс и внешний скрипт
- Listen/Notify и внешний скрипт
- Очередь mbus и внешний скрипт
- Отправка почты из PostgreSQL – плохо



# Задача: авторизация пользователей

- Создание токенов и хранение их в БД
- Подпись токенов с помощью `pgcrypto`
- Реализация JSON Web Tokens – [jwt.io](https://jwt.io)

# Отладка и тестирование

- Я использую VIM IDE
- Тесты, тесты, тесты
- Смотрите доклад про отладку хранимок

# Задачи на будущее

- Nicotine – шаблонизатор для nginx
- Автоматическое создание документации для API
- Автоматическое создание песочницы для API
- Расширить документацию и примеры
- Описать типовые сценарии использования
- “Честные” сравнительные тесты
- Сохранить минимализм, увеличить быстродействие

# На закуску

- PostgREST
- Nginx + Lua
- Tarantool
- Тонкий клиент на Php/Python/Perl

**Благодарю за внимание, земной  
поклон!**

Готов ответить на любые вопросы  
искренне и без хамства

[misha@genosse.org](mailto:misha@genosse.org)