

# C++ и БД. ORM

Владислав Хорев  
Ведущий программист, Andersen



# Проверка связи



Поставьте “+”, если меня видно и слышно



## Если у вас нет звука:

- убедитесь, что на вашем устройстве и на колонках включен звук
- обновите страницу вебинара (или закройте страницу и заново присоединитесь к вебинару)
- откройте вебинар в другом браузере
- перезагрузите компьютер (ноутбук) и заново попытайтесь зайти

# Владислав Хорев

О спикере:

- Ведущий программист в компании Andersen
- Работает в IT с 2011 года
- Опыт разработки на C++ более 11 лет



# Вспоминаем прошрое занятие

**Вопрос:** С помощью какой библиотеки можно подключаться к PostgreSQL из C++?



# Вспоминаем прошрое занятие

**Вопрос:** С помощью какой библиотеки можно подключаться к PostgreSQL из C++?

**Ответ:** Для подключения к PostgreSQL из C++ используется библиотека libpq++



# Вспоминаем прошрое занятие

**Вопрос:** Что такое SQL Injection?



# Вспоминаем прошрое занятие

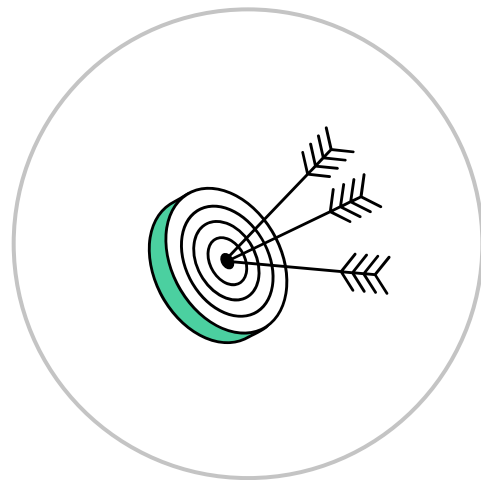
**Вопрос:** Что такое SQL Injection?

**Ответ:** SQL Injection - это атака на базу данных, которая позволит внедрить в запрос произвольный SQL-код.



# Цели занятия

- Разберемся, что такое ORM
- Узнаем, как использовать ORM в C++

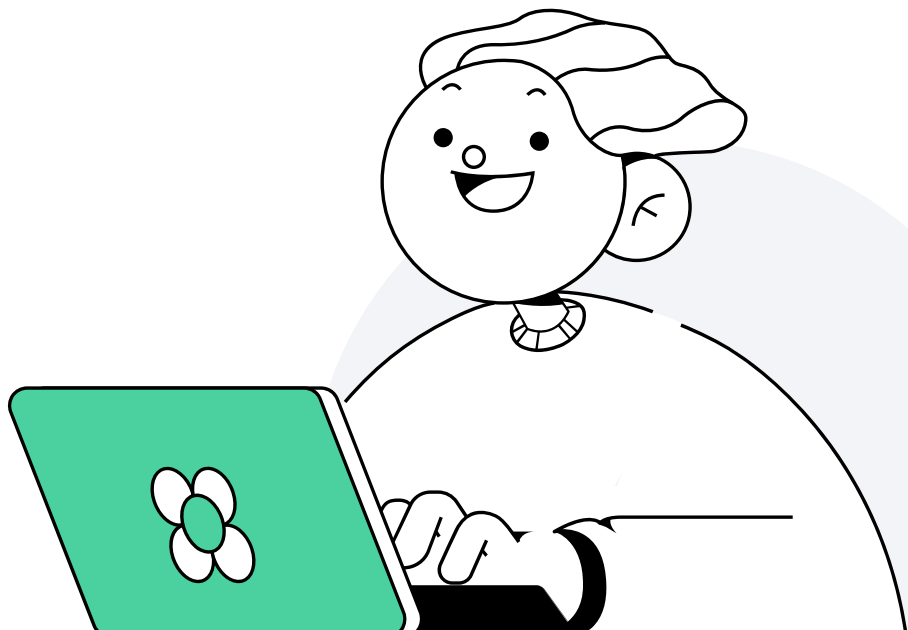




# План занятия

- 1 ORM
- 2 Установка библиотеки wtdbo
- 3 Работа с библиотекой wtdbo
- 4 Домашнее задание

\*Нажми на нужный раздел для перехода



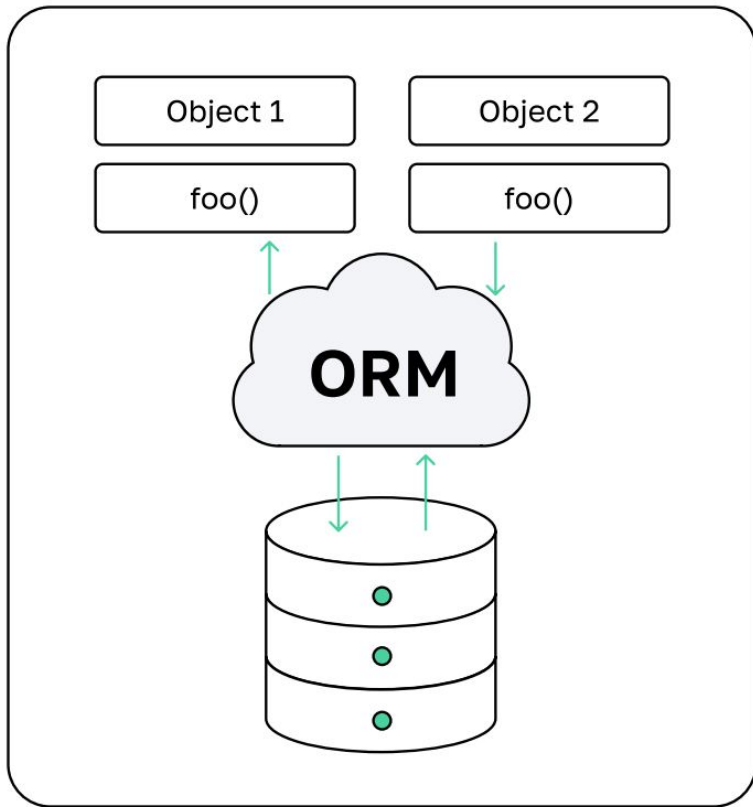
# ORM



1

# Object-Relational Mapping

ORM (объектно-реляционное отображение) — это дополнительный способ взаимодействия с БД из кода, который работает с таблицами и запросами к БД как с классами, объектами и методами в ООП



# Мотивация использовать ORM

- Необязательно знать SQL и специальные функции СУБД
- Возможность десериализации (распаковки) результата из БД в удобном формате для API или обработки
- Не нужно придумывать сложные парсеры массивов, работаем с готовыми структурами данных
- Один и тот же код может работать в разных БД, если на это рассчитана ORM
- Разработчиками ORM продуманы примитивные вопросы безопасности, экранирования и оптимизации запросов

# Недостатки ORM

- Изначально не очевидны итоговые запросы
- Ограничения и читаемость при построении сложных, многоуровневых запросов.

# ORM в C++

В отличие от других языков программирования, в C++ концепция ORM не настолько популярная.

Две самых популярных ORM в C++:

1

**QxOrm**

библиотека для ORM в составе  
фреймворка Qt

2

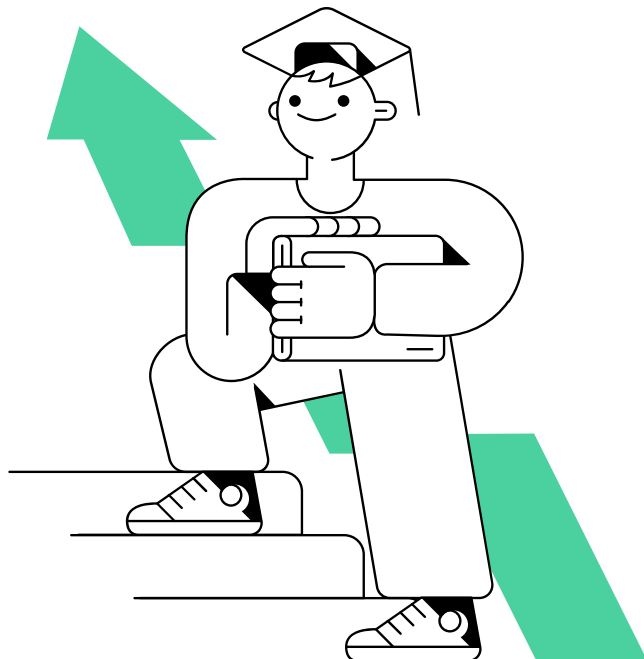
**wtdbo**

библиотека для ORM, часть  
набора библиотек Wt

# ORM в C++

На этом уроке мы будем разбирать работу с ORM на примере **wtdbo**.

Концепция ORM универсальная, и полученные знания вы легко сможете применить для других ORM.



# Установка wtdbo



2



# Установка wtdbo

Библиотека wtdbo находится в составе набора библиотек Wt.

Библиотека wtdbo распространяется под лицензией GNU GPL v2, и ее можно найти на [Github](#)

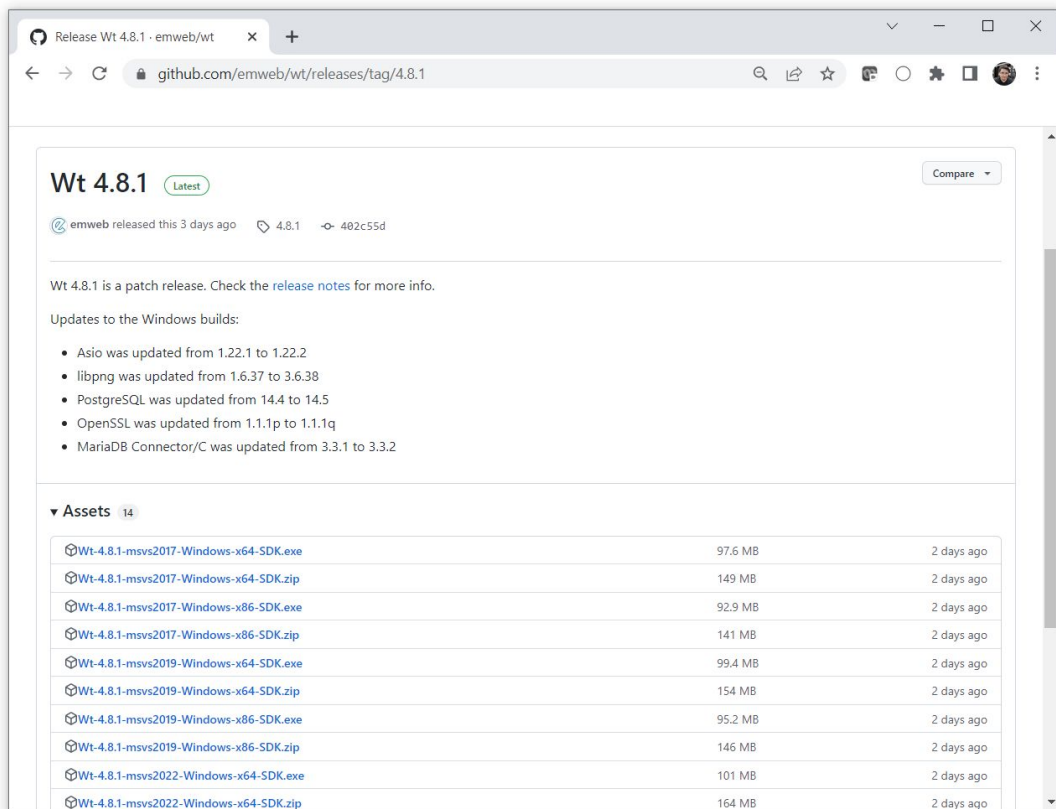
# Установка wtdbo

Установить Wt можно двумя способами:

- Собрать из исходного кода Wt.
- Установить уже скомпилированные библиотеки Wt для своей операционной системы.

Поскольку для сборки Wt требуется много зависимостей, включая Boost, то для урока мы установим уже скомпилированные библиотеки для Windows.

# Рекомендуется скачать самый последний релиз



Release Wt 4.8.1 · emweb/wt

github.com/emweb/wt/releases/tag/4.8.1

## Wt 4.8.1 Latest

emweb released this 3 days ago · 4.8.1 · 402c55d

Wt 4.8.1 is a patch release. Check the [release notes](#) for more info.

Updates to the Windows builds:

- Asio was updated from 1.22.1 to 1.22.2
- libpng was updated from 1.6.37 to 3.6.38
- PostgreSQL was updated from 14.4 to 14.5
- OpenSSL was updated from 1.1.1p to 1.1.1q
- MariaDB Connector/C was updated from 3.3.1 to 3.3.2

▼ Assets 14

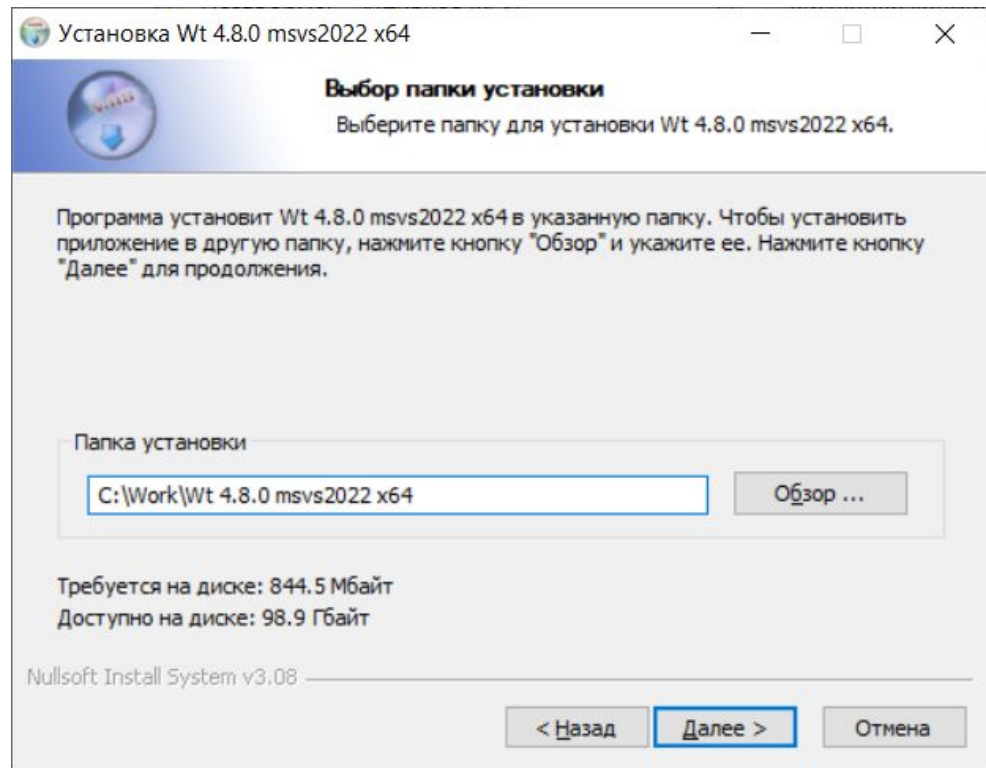
Wt-4.8.1-msvs2017-Windows-x64-SDK.exe	97.6 MB	2 days ago
Wt-4.8.1-msvs2017-Windows-x64-SDK.zip	149 MB	2 days ago
Wt-4.8.1-msvs2017-Windows-x86-SDK.exe	92.9 MB	2 days ago
Wt-4.8.1-msvs2017-Windows-x86-SDK.zip	141 MB	2 days ago
Wt-4.8.1-msvs2019-Windows-x64-SDK.exe	99.4 MB	2 days ago
Wt-4.8.1-msvs2019-Windows-x64-SDK.zip	154 MB	2 days ago
Wt-4.8.1-msvs2019-Windows-x86-SDK.exe	95.2 MB	2 days ago
Wt-4.8.1-msvs2019-Windows-x86-SDK.zip	146 MB	2 days ago
Wt-4.8.1-msvs2022-Windows-x64-SDK.exe	101 MB	2 days ago
Wt-4.8.1-msvs2022-Windows-x64-SDK.zip	164 MB	2 days ago

# Выбирайте версию для вашего компилятора

Если вы используете Windows и работаете в Visual Studio 2022, выбирайте установщик с названием **Wt-\*-msvs2022-Windows-x64-SDK.exe**

# Установка wtdbo

Установите Wt по шагам, выбрав расположение библиотеки:



# Установка wtdbo

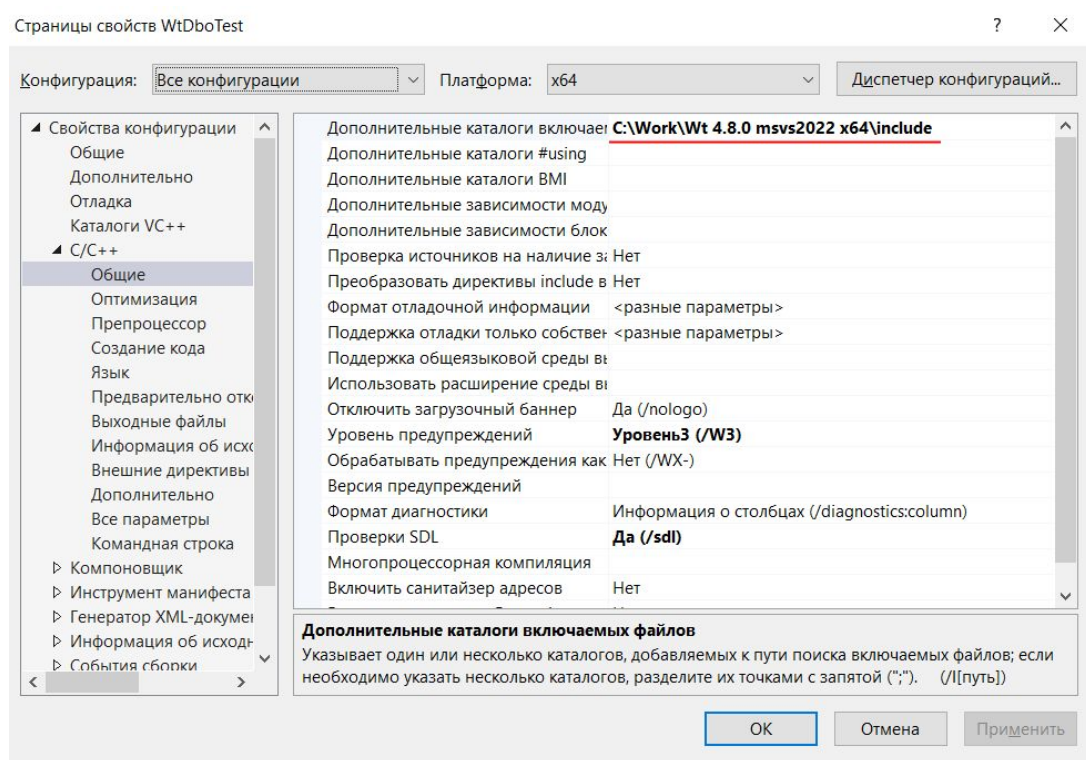
После установки вы сможете использовать wtdbo в своем проекте Visual Studio.

Для того, чтобы использовать wtdbo вместе с PostgreSQL в своем проекте, вам необходимо настроить проект, а именно:

- Указать путь к заголовочным файлам Wt.
- Указать путь к статическим библиотекам Wt.
- Указать библиотеки, с которыми вы будете линковать проект.

# Заголовочные файлы Wt

В настройках проекта Visual Studio укажите путь к заголовочным файлам Wt:



# Заголовочные файлы Wt

После этого вы сможете подключать заголовки libwtdbo в вашем проекте:

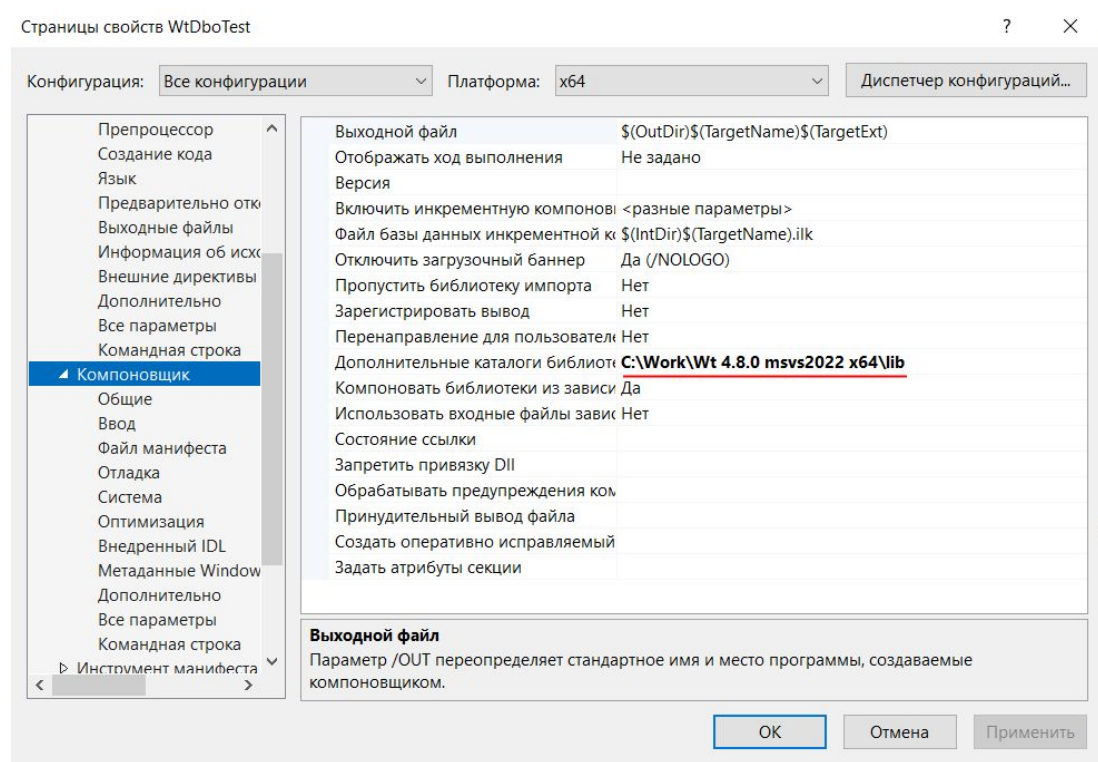
```
#include <Wt/Dbo/Dbo.h>
#include <Wt/Dbo/backend/Postgres.h>

int main()
{
    return 0;
}
```



# Статические библиотеки Wt

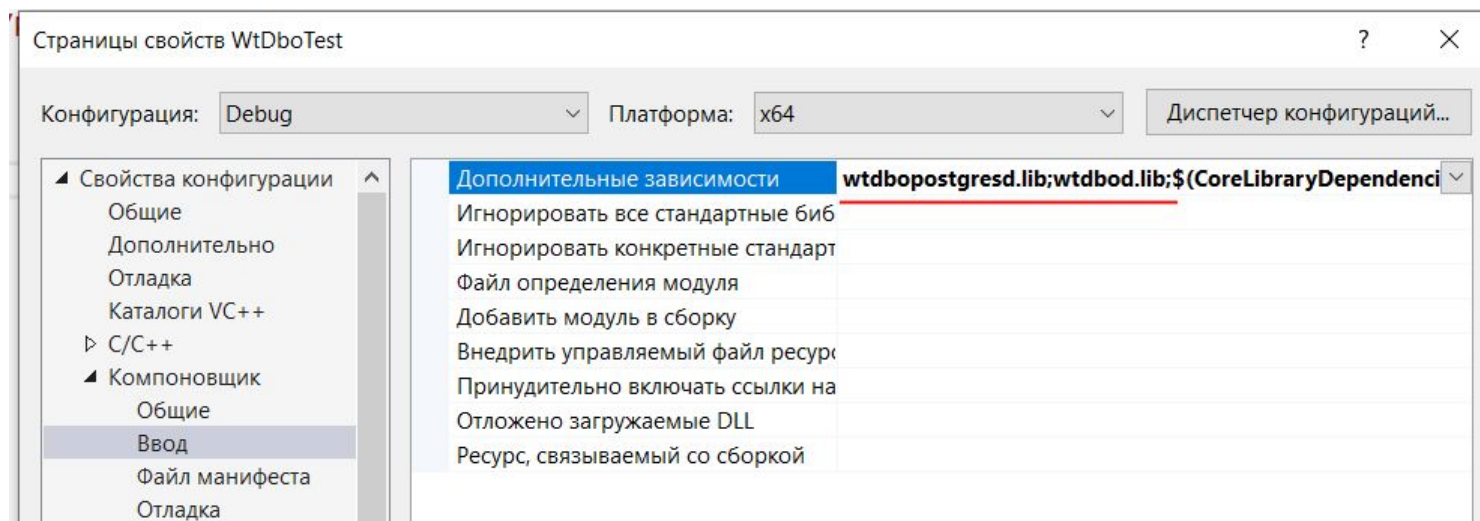
В настройках проекта Visual Studio укажите путь к статическим библиотекам Wt:



# Статические библиотеки Wt

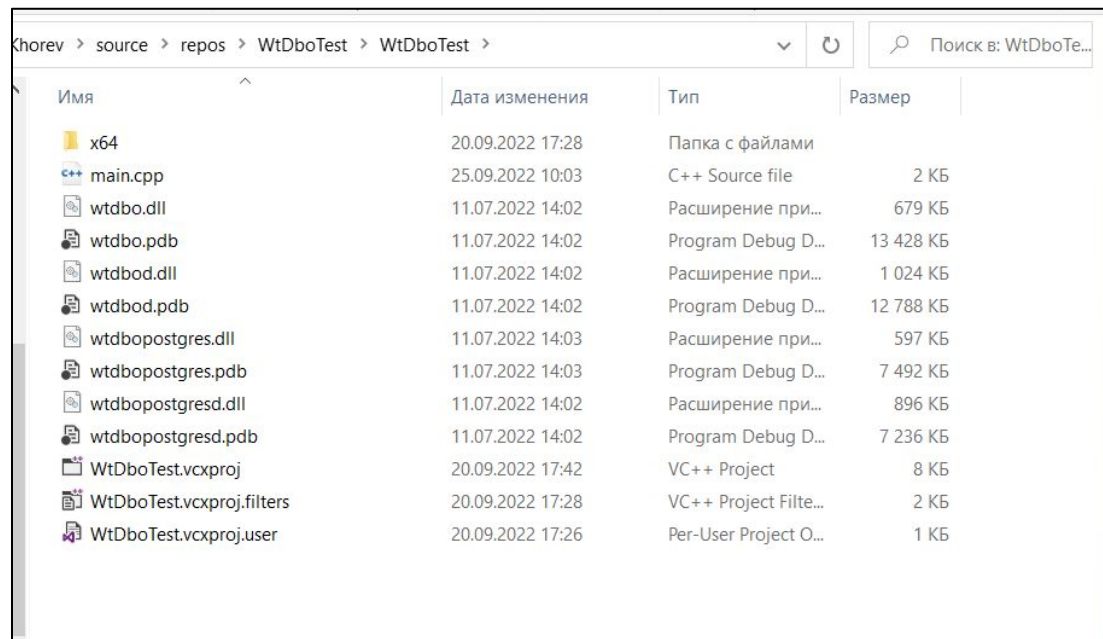
В настройках проекта Visual Studio в зависимостях укажите две статические библиотеки:

- **wtdbopostgres.lib** (для конфигурации Debug - **wtdbopostgresd.lib**)
- **wtdbo.lib** или (для конфигурации Debug - **wtdbod.lib**)



# Динамические библиотеки Wt и PostgreSQL

После сборки проекта, поместите в каталог с проектом динамические библиотеки **wtdbo.dll**, **wtdbod.dll**, **wtdbopostgres.dll** и **wtdbopostgresd.dll** из каталога, где установлен Wt:



Имя	Дата изменения	Тип	Размер
x64	20.09.2022 17:28	Папка с файлами	
main.cpp	25.09.2022 10:03	C++ Source file	2 КБ
wtdbo.dll	11.07.2022 14:02	Расширение при...	679 КБ
wtdbo.pdb	11.07.2022 14:02	Program Debug D...	13 428 КБ
wtdbod.dll	11.07.2022 14:02	Расширение при...	1 024 КБ
wtdbod.pdb	11.07.2022 14:02	Program Debug D...	12 788 КБ
wtdbopostgres.dll	11.07.2022 14:03	Расширение при...	597 КБ
wtdbopostgres.pdb	11.07.2022 14:03	Program Debug D...	7 492 КБ
wtdbopostgresd.dll	11.07.2022 14:02	Расширение при...	896 КБ
wtdbopostgresd.pdb	11.07.2022 14:02	Program Debug D...	7 236 КБ
WtDboTest.vcxproj	20.09.2022 17:42	VC++ Project	8 КБ
WtDboTest.vcxproj.filters	20.09.2022 17:28	VC++ Project Filte...	2 КБ
WtDboTest.vcxproj.user	20.09.2022 17:26	Per-User Project O...	1 КБ

# Работа с wtdbo



2

# Объявление ORM-класса

Объявление ORM-класса записывается так:

```
class User {  
public:  
    std::string name = "";  
    std::string phone = "";  
    int         karma = 0;  
  
    template<class Action>  
    void persist(Action& a)  
    {  
        Wt::Dbo::field(a, name, "name");  
        Wt::Dbo::field(a, phone, "phone");  
        Wt::Dbo::field(a, karma, "karma");  
    }  
};
```

# Подключение к PostgreSQL

Для работы ORM, необходимо подключиться к базе данных PostgreSQL:

```
int main()
{
    std::string connectionString =
        "host=localhost"
        " port=5432"
        " dbname=lesson03"
        " user=lesson03user"
        " password=lesson03user";

    auto postgres = std::make_unique<Wt::Dbo::backend::Postgres>(connectionString);

    Wt::Dbo::Session session;
    session.setConnection(std::move(postgres));
    session.mapClass<User>("user");

    // ...
}
```

# Подключение к PostgreSQL

**Совет:** для упрощения обработки ошибок, оборачивайте вызовы к Wt::Dbo в блок try, а в блоке catch ловите исключение Wt::Dbo::Exception. Пример:

```
try
{
    std::string connectionString =
        "host=localhost "
        "port=5432 "
        "dbname=my_database "
        "user=my_database_user "
        "password=my_password_123";

    // ...

} catch (const Wt::Dbo::Exception& e)
{
    std::cout << e.what() << std::endl;
}
```

# Создание таблиц






При работе ORM не нужно заранее создавать таблицы с необходимой структурой - ORM позволяет сделать это автоматически:

```
/*  
 * Try to create the schema (will fail if already exists).  
 */  
  
session.createTables();
```



# Создание таблиц

Этот код создаст такую таблицу:

Название	#	Тип данных	Автоувеличение	Правило сортировки	Not Null	По умолчанию	Комментарий
 id	1	bigserial			[v]	nextval('user_i...	
 version	2	int4			[v]		
 name	3	text		<a href="#">default</a>	[v]		
 phone	4	text		<a href="#">default</a>	[v]		
 karma	5	int4			[v]		

Если таблица уже создана, повторно создавать ее не нужно.

# Добавление записей

Пример создания транзакции для добавления новой записи в таблицу:

```
Wt::Dbo::Transaction transaction{ session };

std::unique_ptr<User> user{ new User() };
user->name = "Joe";
user->phone = "1234567890";
user->karma = 13;

Wt::Dbo::ptr<User> userPtr = session.add(std::move(user));

transaction.commit();
```

# Добавление записей

После выполнения вышеописанного кода, такая запись будет добавлена в таблицу:

Свойства

Данные

Диаграмма

lesson03

Базы

user

Введите SQL выражение чтобы отфильтровать результаты

Таблица	<div>123id</div>	<div>123version</div>	<div>ABCname</div>	<div>ABCphone</div>	<div>123karma</div>	
	1	0	Joe	1234567890	13	
Текст						

# Получение записей

Пример кода для получения всех записей:

```
typedef Wt::Dbo::collection<Wt::Dbo::ptr<User>> Users;

Wt::Dbo::Transaction transaction{ session };

Users users = session.find<User>();

std::cout << "We have " << users.size() << " users:" << std::endl;

for (const Wt::Dbo::ptr<User>& user : users)
{
    std::cout << " user " << user->name
               << " with karma of " << user->karma << std::endl;
}

transaction.commit();
```

# Получение записей

Если вам нужно получить только одну запись:

```
Wt::Dbo::ptr<User> joe = session.find<User>().where("name = ?").bind("Joe");  
  
std::cout << "Joe has karma: " << joe->karma << std::endl;
```

# Редактирование записей

В отличие от других ORM, по умолчанию все записи Read-Only. Если вы хотите редактировать запись, вам следует вызвать метод `modify()`:

```
Wt::Dbo::ptr<User> joe = session.find<User>().where("name = ?").bind("Joe");

std::cout << "Joe has karma: " << joe->karma << std::endl;

joe.modify()->name = "John";
joe.modify()->karma = 100;
```

# Отношение Many-To-One

Предположим, что у нас есть отношения между таблицами user и post. Один пользователь может написать много постов. Для этого определим класс Post:

```
class User;

class Post {
public:
    std::string title = "";
    std::string text = "";
    dbo::ptr<User> user;

    template<class Action>
    void persist(Action& a)
    {
        Wt::Dbo::field(a, title, "title");
        Wt::Dbo::field(a, text, "text");
        dbo::belongsTo(a, user, "user");
    }
};
```

# Отношение Many-To-One

Класс User тоже необходимо дополнить:

```
class User {  
public:  
    std::string name = "";  
    std::string phone = "";  
    int         karma = 0;  
    Wt::Dbo::collection< dbo::ptr<Post> > posts;  
  
    template<class Action>  
    void persist(Action& a)  
    {  
        Wt::Dbo::field(a, name, "name");  
        Wt::Dbo::field(a, phone, "phone");  
        Wt::Dbo::field(a, karma, "karma");  
        Wt::Dbo::hasMany(a, posts, dbo::ManyToOne, "user");  
    }  
};
```



# Отношение Many-To-One

Теперь вы можете легко добавлять пользователям посты, или наоборот - назначать посту пользователя. Система ORM автоматически создаст внешние ключи, и будет сама управлять связями между сущностями:

```
Wt::Dbo::ptr<Post> post = session.add(std::unique_ptr<Post>{new Post()});  
post.modify()->user = joe; // or joe.modify()->posts.insert(post);  
  
// will print 'Joe has 1 post(s).'
```

```
std::cout << "Joe has " << joe->posts.size() << " post(s)." << std::endl;
```

# Итоги



# Итоги занятия

Сегодня мы

- 1 Разобрались, что такое ORM
- 2 Узнали, как использовать библиотеку wtddbo с PostgreSQL



# Домашнее задание

Давайте посмотрим ваше [домашнее задание](#).

- 1 Вопросы по домашней работе задавайте в чате группы
- 2 Задачи можно сдавать по частям
- 3 Зачёт по домашней работе ставят после того, как приняты все задачи



# Дополнительные материалы

- [Документация по wtdbo \(на английском языке\)](#)



# Задавайте вопросы и пишите отзыв о лекции

Владислав Хорев  
Ведущий программист, Andersen

