

Программирование

- [Новая страница](#) официальная инструкция для программистов ООО «Воронка» по созданию, тестированию и накатыванию миграций в SaaS-платформе

Новая страница официальная инструкция для программистов ООО «Воронка» по созданию, тестированию и накатыванию миграций в SaaS-платформе

Инструкция: создание и применение миграций

Общие положения

- Один код CRM обслуживает несколько клиентов (`vcclient`) через симлинки.
- У каждого клиента — своя база данных.
- Есть эталонный клиент `vcclientcloneimage`, на котором формируются миграции и который участвует во всех обновлениях.
- Прямое редактирование БД (phpMyAdmin, `ALTER`, `INSERT` вручную) запрещено.
- Все изменения происходят через PHP-скрипты миграций с использованием `Vtlib`.
- Все кодовые изменения фиксируются в Git.

Структура миграции

Каждая миграция — это отдельная папка в директории `migrations/`:

```
/migrations/  
└─ 20250626_01_inventory_module/  
    └─ 001_add_module_inventory.php  
    └─ 002_add_field_itemcode.php  
    └─ patch.diff  
    └─ metadata.json
```

- `*.php` — миграции через `Vtlib` API.
- `patch.diff` — патч файловой части кода (создан через `git diff`).
- `metadata.json` — опциональное описание версии и содержимого.

Этапы работы

Этап 1: Подготовка разработки

1. На клиенте `vcclientcloneimage`:
 - создаются новые модули, поля, таблицы;
 - добавляются через `Vtlib` из под учётки с `isMasterUser = true`.
2. Кодовые изменения (новые классы, шаблоны, стили) делаются в `PhpStorm` в основной CRM-папке.
3. После добавления:

- vtlib автоматически вызывает `MigrationManager::addModule(...)` или `addField(...)`, которые формируют миграционные `.php` скрипты в новой папке `/migrations/YYYYMMDD_XX_название/`.

Этап 2: Создание патча кода

1. Все изменения фиксируются в Git:

```
git add .
git commit -m "feat: добавлен модуль Inventory и поле itemcode"
```

2. Формируется патч:

```
git diff HEAD~1 > migrations/20250626_01_inventory_module/patch.diff
```

Этап 3: Тестирование миграции

1. Копируй миграционную папку на сервер тестирования.
2. Выполни:

```
php applyMigrations.php
```

3. Скрипт сам:

- применит `patch.diff`;
- выполнит все `*.php` скрипты для всех клиентов, включая `vclientcloneimage`;
- подключит окружение каждого клиента (через `$_ENV['VCLIENT']` + `DOCUMENT_ROOT`);
- выполнит миграции с использованием `Vtlib`.

4. Проверь работу нового функционала на тест-клиенте (`vclientcloneimage` или любой другой).

Этап 4: Архивация миграции

1. После успешного теста:

```
cd migrations/
tar -czf 20250626_01_inventory_module.tar.gz 20250626_01_inventory_module/
```

2. Архив переносится на продакшн-сервер через SCP или Git (в зависимости от инфраструктуры).

Этап 5: Обновление продакшн

Выполняется только системным администратором:

1. Включить заглушку (maintenance):

- отключить Cron-задачи;
- удалить активные сессии;
- повесить stub `/crm/maintenance.html`.

2. Распаковать архив миграции:

```
tar -xzf 20250626_01_inventory_module.tar.gz -C migrations/
```

3. Запустить:

```
php applyMigrations.php
```

4. После завершения:

- удалить заглушку;
- включить Cron;
- провести проверку клиента `vclientcloneimage` и выборочно других.

Правила безопасности

- **Запрещено:** использовать `phpMyAdmin`, `MySQL Workbench`, прямые SQL-запросы руками.
- **Обязательно:** все изменения только через `.php` миграции и `Vtlib`.
- **Обязательно:** все коммиты фиксируются и сопровождаются `patch.diff`.
- **Обязательно:** перед продакшн обновлением — проверка на сервере тестирования.

Примеры скриптов

Миграция создания модуля

```
<?php
use vtlib\Module as Vtiger_Module;
```

```
$module = new Vtiger_Module();  
$module->name = 'Inventory';  
$module->save();
```

Миграция добавления поля

```
<?php  
$module = Vtiger_Module::getInstance('Inventory');  
$block = Vtiger_Block::getInstance('LBL_INVENTORY_INFORMATION', $module);  
$field = new Vtiger_Field();  
$field->name = 'itemcode';  
$field->label = 'Item Code';  
$field->uitype = 1;  
$field->typeofdata = 'V~0';  
$block->addField($field);
```

Пример команды запуска

```
php applyMigrations.php
```

Примечания

- Все миграции должны быть **идемпотентны** — можно запускать повторно без ошибок.
- Названия миграционных папок должны быть в формате `YYYYMMDD_XX_description`.

Если нужна инструкция для админов по обслуживанию системы миграций — скажи, подготовлю отдельно.