Migration Yii2

Control, Creación y Versionado de BD.

¿Problemas con la versiones de la BD?

¿Dónde?

Descargué el script con los nuevos cambios en el modelo y realice el insert. Necesitaria saber qué cambios específicos contiene el mismo!?.

Cuando tienes que depurar un nuevo script que rompe tu proyecto actual.

¿ Migrations? ¿que son?¿que solucionan?

- Permite tener un seguimiento de los cambios en la base de datos ,teniendo un historial de las modificaciones que se han hecho, ofreciendo una símil-versionado local que además puede complementarse con una herramienta de versionado como lo es git. Pudiendo de esta forma distribuir estas migraciones o versiones de la base de datos local a nuestro grupo de trabajo subiendo los cambios al repositorio remoto.
- La base irá creciendo de manera iterativa e incremental a medida que el proyecto avance proveyendo una trazabilidad en el desarrollo de cada requerimiento o feature .
- Nos independiza de la base de datos, las migraciones funcionaran para cualquier base soportada por el framework.

EJEMPLO- ESCENARIO

Los siguientes pasos muestran como una migración de la base de datos puede ser usada por un equipo durante el desarrollo.

1. Ariel crea una nueva migración (ej: crea una nueva tabla; cambia la definición de una columna)
2. Ariel realiza un commit con el controlador de versiones (ej: git)
1. Micaela actualiza su repositorio local y recibe la migración.
2. Micaela aplica la migración en su base de datos local de desarrollo y ve reflejados los cambios que realizo Ariel.

Los siguientes pasos muestran cómo deployar una nueva release en una base de datos en producción:

1. Martin realiza un merge de las ramas de desarrollo y producción.
2. Martin aplica las migraciones (migration up) y ve reflejados los cambios en la base de datos.
3. Martin encuentra un bug producido por el cambio en el modelo de la base entonces realiza un rollback (migration down)

Commands

Yii provides a set of migration command line tools that allow you to:

- create new migrations;
- apply migrations;
- revert migrations;
- re-apply migrations;
- show migration history and status.

All these tools are accessible through the command php yii migrate.

COMMANDS

- migrate/create Creates a new migration.

- migrate/history Displays the migration history.

-migrate/up (default) Upgrades the application by applying new migrations.

- migrate/down Downgrades the application by reverting old migrations.

- migrate/mark Modifies the migration history to the specified version.

- migrate/new Displays the un-applied new migrations.

- migrate/redo Redoes the last few migrations.

- migrate/to Upgrades or downgrades till the specified version.

To see the detailed information about individual sub-commands, enter:

yii help <sub-command>

Especificación

- Esta herramienta está integrada en yii2.
- All these tools are accessible through the command
  - php yii migrate
- Al crear la primera migración se incorpora la carpeta migrations al proyecto, la cual contendrá las migraciones que se describirán con el siguiente formato:
  - m<YYMMDD_HHMMSS>_<Name>
    - Where:
    - <YYMMDD_HHMMSS> refers to the UTC datetime at which the migration creation command is executed.
    - <Name> is the same as the value of the name argument that you provide to the command.

- Además se incluirá en la base de datos una tabla denominada migration donde se guardará el historial local de las migraciones aplicadas.
    - The table will be automatically created the first time this command is executed, if it does not exist. You may also manually

      create it as follows:CREATE TABLE migration (

      version varchar(180) PRIMARY KEY,

      apply_time integer

)

- **Transnational Migrations**

  While performing complex DB migrations, it is important to ensure each migration to either succeed or fail as a whole so that the database can maintain integrity and consistency. To achieve this goal, it is recommended that you enclose the DB operations of each migration in a transaction.
  An even easier way of implementing transactional migrations is to put migration code in the safeUp() and safeDown() methods. These two methods differ from up() and down() in that they are enclosed implicitly in a transaction. As a result, if any operation in these methods fails, all prior operations will be rolled back automatically.

```
class m150101_185401_create_news_table extends Migration
{
    public function safeUp()
    {
        $this->createTable('news', [
            'id' => $this->primaryKey(),
            'title' => $this->string()->notNull(),
            'content' => $this->text(),
        ]);

        $this->insert('news', [
            'title' => 'test 1',
            'content' => 'content 1',
        ]);
    }

    public function safeDown()
    {
        $this->delete('news', ['id' => 1]);
        $this->dropTable('news');
    }
}
```

Sintaxis de las migrations

Creando tabla usuario:

yii migrate/create **usuario**

```
<?php
use yii\db\Migration;
class m150101_185401_usuario extends Migration
{
    public function up()
    {

        //se ejecuta al realizar el migrate/create
    }
    public function down()
    {   //se ejecuta al hacer el migrate/down
        echo "m101129_185401_create_news_table cannot be reverted.\n";
        return false;
    }
}
```

```
$this->createTable('example_table', [
  'id' => $this->primaryKey(),
  'name' => $this->string(64)->notNull(),
  'type' => $this->integer()->notNull()->defaultValue(10),
  'description' => $this->text(),
  'rule_name' => $this->string(64),
  'data' => $this->text(),
  'created_at' => $this->datetime()->notNull(),
  'updated_at' => $this->datetime(),
]);
```

- También puede crearse una migración de la siguiente manera:
  - To create table fields right away, specify them via **--fields** option.

yii migrate/create NewTable --fields="author_id:integer:notNull:foreignKey(user),

category_id:integer:defaultValue(1):foreignKey,

title:string,body:text"

Ejemplo Completo:

yii migrate/create create_post_table --fields="title:string,body:text"
Generates

```
class m150811_220037_create_post_table extends Migration
{
    public function up()
    {
        $this->createTable('post', [
            'id' => $this->primaryKey(),
            'title' => $this->string(),
            'body' => $this->text(),
        ]);
    }

    public function down()
    {
        $this->dropTable('post');
    }

}
```

TIPOS DE DATOS

- pk: an auto-incremental primary key type, will be converted into "int(11) NOT NULL AUTO_INCREMENT PRIMARY KEY"
- string: string type, will be converted into "varchar(255)"
- text: a long string type, will be converted into "text"
- integer: integer type, will be converted into "int(11)"
- boolean: boolean type, will be converted into "tinyint(1)"
- float: float number type, will be converted into "float"
- decimal: decimal number type, will be converted into "decimal"
- datetime: datetime type, will be converted into "datetime"
- timestamp: timestamp type, will be converted into "timestamp"
- time: time type, will be converted into "time"
- date: date type, will be converted into "date"
- binary: binary data type, will be converted into "blob"

Métodos

- execute(): executing a SQL statement
- insert(): inserting a single row
- batchInsert(): inserting multiple rows
- update(): updating rows
- delete(): deleting rows
- createTable(): creating a table
- renameTable(): renaming a table
- dropTable(): removing a table
- truncateTable(): removing all rows in a table
- addColumn(): adding a column
- renameColumn(): renaming a column
- dropColumn(): removing a column
- alterColumn(): altering a column
- addPrimaryKey(): adding a primary key
- dropPrimaryKey(): removing a primary key
- addForeignKey(): adding a foreign key
- dropForeignKey(): removing a foreign key
- createIndex(): creating an index
- dropIndex(): removing an index
- addCommentOnColumn(): adding comment to column
- dropCommentFromColumn(): dropping comment from column
- addCommentOnTable(): adding comment to table
- dropCommentFromTable(): dropping comment from table

OPERACIONES COMUNES

- **Add Column**

  If the migration name is of the form add_xxx_column_to_yyy_table then the file content would contain addColumn and dropColumn statements necessary.
  yii migrate/create add_position_column_to_post_table --fields="position:integer"

  generates

```
class m150811_220037_add_position_column_to_post_table extends Migration
{
    public function up()
    {
        $this->addColumn('post', 'position', $this->integer());
    }
    public function down()
    {
        $this->dropColumn('post', 'position');
    }
}
```

Method Details

- **Creates a primary key column.**
  - public yii\db\ColumnSchemaBuilder **primaryKey** ($length = nul)
- **Creates a boolean column.**
  - public yii\db\ColumnSchemaBuilder **boolean** ( )
- **Creates a timestamp column.**
  - public yii\db\ColumnSchemaBuilder **timestamp** ($precision = null)
- **Creates a string column.**
  - public yii\db\ColumnSchemaBuilder **string** ($length = null)
- **Creates a time column.**
  - public yii\db\ColumnSchemaBuilder **time** ($precision = null)
- **Creates a float column.**
  - public yii\db\ColumnSchemaBuilder **float** ($precision = null)
- **Creates an integer column.**
  - public yii\db\ColumnSchemaBuilder **integer** ($length = null)
- **Creates a double column.**
  - public yii\db\ColumnSchemaBuilder **double** ($precision = null)
- **Creates a date column.**
  - public yii\db\ColumnSchemaBuilder **date** ( )
- **Creates a datetime column.**
  - public yii\db\ColumnSchemaBuilder **dateTime** ($precision = null)
- **Creates a char column.**
  - public yii\db\ColumnSchemaBuilder **char** ($length = null)
- **Creates a decimal column.**
  - public yii\db\ColumnSchemaBuilder **decimal** ($precision = null, $scale = null)

- ## Foreign keys

Since 2.0.8 the generator supports foreign keys using the foreignKey keyword.
yii migrate/create create_post_table
--fields="author_id:integer:notNull:foreignKey(user),category_id:integer:defaultValue(1):foreignKey,title:string,body:text"
generates

```
/**
 * Handles the creation for table `post`.
 * Has foreign keys to the tables:
 *
 * - `user`
 * - `category`
 */
class m160328_040430_create_post_table extends Migration
{

    public function up()
    {
        $this->createTable('post', [
            'id' => $this->primaryKey(),
            'author_id' => $this->integer()->notNull(),
            'category_id' => $this->integer()->defaultValue(1),
            'title' => $this->string(),
            'body' => $this->text(),
        ]);

        // creates index for column `author_id`
        $this->createIndex(
            'idx-post-author_id',
            'post',
            'author_id'
        );

        // add foreign key for table `user`
        $this->addForeignKey(
            'fk-post-author_id',
            'post',
            'author_id',
            'user',
            'id',
            'CASCADE'
        );

        / creates index for column `category_id`
        $this->createIndex(
            'idx-post-category_id',
            'post',
            'category_id'
        );

        // add foreign key for table `category`
        $this->addForeignKey(
            'fk-post-category_id',
            'post',
            'category_id',
            'category',
            'id',
            'CASCADE'
        );
    }
```

Incorporamos las migraciones y ahorremos dolores de cabeza.

FIN

enlaces

http://www.yiiframework.com/doc/guide/1.1/en/database.migration