# Nanbando Documentation

### *Release 0.1*

**Johannes Wachter**

November 26, 2016

Nanbando is a simple application to automate website backups. It provides an elegant way to extend and configure the backup parts. Nanbando has built-in support for various storage's and provides easy to use sync and fetch operations. It was built with modularity, extensibility and simplicity in mind.

# Contents

## 1.1 Installation

To install the application simply download the executable and move it to the global bin folder.

```
wget http://nanbando.github.io/core/nanbando.phar
wget http://nanbando.github.io/core/nanbando.phar.pubkey
chmod +x nanbando.phar
mv nanbando.phar /usr/local/bin/nanbando
mv nanbando.phar.pubkey /usr/local/bin/nanbando.pubkey
```

After first installation you can update the application with a built-in command.

```
nanbando self-update
```

**Note:** The executable is signed with a OpenSSL private key. This ensures the origin of the build.

Check the configuration state of your application by using the command `nanbando check`.

## 1.2 Usage

Before we can start to create backup-projects we have to configure the local and remote storage. This global configuration will be shared for all projects of the executing user. The configuration will be written in the file `~/.nanbando.yml`.

```
nanbando:
    storage:
        local_directory: "%home%/nanbando/local"
        remote_service: filesystem.remote

oneup_flysystem:
    adapters:
        remote:
            local:
                directory: "%home%/nanbando/remote"

    filesystems:
        remote:
            adapter: remote
```

```
            alias: filesystem.remote
        plugins:
            - filesystem.list_files
```

---

**Note:** In the configuration you can use the parameter `%home%` which points to the home directory of the current user.

---

The application contains a simple directory backup plugin which we will use in this simple usage example. To start a new backup goto the root directory of your website and create a file named `nanbando.json` which contains the configuration and later also the dependencies for this backup-project.

```
{
    "name": "application",
    "backup": {
        "data": {
            "plugin": "directory",
            "parameter": {
                "directory": "path/to/data/directory"
            }
        }
    },
    "require": {
    }
}
```

After you have created this file you can run following command to configure the local installation with the given configuration. If you have added requirements to the configuration the application will install them into the folder `.nanbando`.

```
php nanbando.phar reconfigure
php nanbando.phar backup
```

The second command will create a new backup zip in the local folder `~/nanbando/local/application/<date>.zip`.

After this steps you can do following steps:

- `php nanbando.phar restore` - restore a local backup

- `php nanbando.phar push` - push backups to remote storage

- `php nanbando.phar fetch` - fetch a backup on a different machine to restore it there

## 1.3 Configuration

The configuration is devided into two parts - global (optional) and project configuration.

---

**Warning:** After changing configuration please run command `reconfigure` to be sure that the configuration will be used for recreating the symfony container.

---

### 1.3.1 Global Configuration

The global congfiguration is placed in the user home directory. This will be used for all projects used by the user. Put this configuration into `~/.nanbando.yml`.

---

```
nanbando:
    storage:
        local_directory: "%home%/nanbando"
        remote_service: filesystem.remote

oneup_flysystem:
    adapters:
        remote:
            local:
                directory: "%home%/nanbando/remote"

    filesystems:
        remote:
            adapter: remote
            alias: filesystem.remote
            plugins:
                - filesystem.list_files
```

**Note:** The configuration documentation for the `oneup_flysystem` can be found on github OneupFlysystemBundle.

For nanbando you have to define the local directory, where the backup command can place the backup archives, and the remote filesystem-service which can be configured in the `oneup_flysystem` extension.

By default the `local_directory` will be set to `%home%/nanbando` and the `remote_service` will be `null`. This leads to local backups will work out of the box but all commands (`fetch`, `push`) which needs the remote-storage will be disabled.

## 1.3.2 Local Configuration

The local configuration contains the name, backup configuration and the additional Plugins.

```
{
    "name": "application",
    "parameters": {
        "directory": "path/to/data/directory"
    },
    "backup": {
        "data": {
            "plugin": "directory",
            "parameter": {
                "directory": "%directory%"
            }
        }
    },
    "require": {
    }
}
```

The `backup` section can contain as much parts as needed. Each plugin can provide its own `parameter` structure.

**Note:** The section `parameters` can be used to define global parameters which can be used in the plugin configuration. To import files place them in the `imports` array. This can be used to reuse the symfony-application parameter.

## 1.4 Plugins

Nanabando was written with extensibility in mind. To keep the core as small as possible only one plugin is included in the application. But nanbando also provides optional plugins which can be installed by each backup-project.

### 1.4.1 Usage

You can use a plugin by adding it to the `nanbando.json` file. There you can configure it like other composer projects in the require section of the file.

```
{
    "name": "application",
    "backup": {
        "su_standard": {
            "plugin": "mysql",
            "parameter": {
                "username": "root",
                "database": "your_database"
            }
        }
    },
    "require": {
        "nanbando/mysql": "dev-master"
    }
}
```

To install this plugin run the `reconfigure` command. It will install the plugin with the embedded-composer and reconfigure the local application. After that you can run the `backup` command to backup the database and `restore` to restore the database.

### 1.4.2 Available Plugins

This list of plugins is currently quite small but there should be more plugins soonish.

- `nanbando/mysql` - This plugin backups your mysql-database with the `mysqldump` command
- `nabando/jackrabbit` - This plugin will backups your jackrabbit data by exporting into xml

## 1.5 Cookbook

The Cookbook covers some applications and how nanbando can be used to backup the data of this applications.

### 1.5.1 How to backup a Sulu application?

You can use the following configuration to backup the application using jackrabbit as phpcr data storage.

```
{
    "name": "test-application",
    "imports": [
        "app/config/parameters.yml"
    ],
    "parameters": {
        "jackrabbit_uri": "http://localhost:8080/server/"
```

```
    },
    "backup": {
        "uploads": {
            "plugin": "directory",
            "parameter": {
                "directory": "uploads"
            }
        },
        "database": {
            "plugin": "mysql",
            "parameter": {
                "username": "%database_user%",
                "password": "%database_password%",
                "database": "%database_name%"
            }
        },
        "cmf": {
            "plugin": "jackrabbit",
            "parameter": {
                "jackrabbit_uri": "%jackrabbit_uri%",
                "path": "/cmf"
            }
        },
        "versions": {
            "plugin": "jackrabbit",
            "parameter": {
                "jackrabbit_uri": "%jackrabbit_uri%",
                "path": "/jcr:versions"
            }
        }
    },
    "require": {
        "nanbando/mysql": "dev-master",
        "nanbando/jackrabbit": "dev-master"
    }
}
```

If you use mysql as data storage for phpcr you can remove the `cmf` and `versions` part of the backup.

```
{
    "name": "test-application",
    "imports": [
        "app/config/parameters.yml"
    ],
    "backup": {
        "uploads": {
            "plugin": "directory",
            "parameter": {
                "directory": "uploads"
            }
        },
        "database": {
            "plugin": "mysql",
            "parameter": {
                "username": "%database_user%",
                "password": "%database_password%",
                "database": "%database_name%"
            }
        }
```

```
    },
    "require": {
        "nanbando/mysql": "dev-master"
    }
}
```

# Indices and tables

- genindex
- modindex
- search