Generic PHP application installer

A discussion

Kore Nordmann

September 27, 2010

Arbit installer

- Web installer for PHP projects
- Design phase finished
  - Results at: `http://tracker.arbitracker.org/arbit/development_wiki/view/Installer`
  - Please verify that it would work for *your* application
  - Proof of concept in implementation phase

    `svn://arbitracker.org/arbit/projects/installer/trunk`
- Major feature of next arbit release

Target groups

- Shared-host users
- Dedicated-server users
- Administrators

Common use cases

- Download arbit installer package.
- Copy package to shared host.
- Call URL pointing to the installer.
- Enter security code, to get access to the installer.
- Enter the required data, check for required dependencies, click "Install".
- Have access to an usable arbit instance.

Requirenments

- General
- Security
- Check and verify system requirements
- Collecting configuration data
- Installation
- Interface

General requirements

- The installer should run on "any" installation and fail gracefully, if something is not available. This also means that the installer should work with PHP 5.1.4.
- Handling of dependent tasks. The configuration of a first task might require configuring another additional task. Example for arbit: The selected modules for a project will require configuring those modules.
- Provided as a single PHP file, or a single directory.
- Ability to upgrade existing installations.

Security requirements

- Require creation of file to proceed with installation
  - Or check for existence of a file the user has to create
  - Ask for install password that has been written in a non-webaccessible file on the server
- Lock installation tool to a single IP address (optional)
- Remove write rights from whole application directory after installation, if
- wished even from config file
  - Make installer unavailable after installation

Check and verify system requirements

- Minimum PHP version
    - PEAR libraries (optional)
    - PHP extensions
- safe_mode
- Session availability
- File upload settings (file_uploads, upload_max_filesize) and post_max_size, perhaps generating .htaccess with correct settings automatically
- Checking a list of required PHP functions, i.e. exec (disable_functions)
- Script time limit (max_execution_time)
- Allowed RAM size (memory_limit) (optional)
- Writability of directories and files, creating them if necessary (i.e. cache dirs)
- PHP ini settings

Collecting configuration data

- Database settings
  - Support for different database systems (optional)
  - Importing an SQL dump from file (optional)
  - Creating the database, possibly with a different user that has admin rights (optional)
  - Checking for connectivity
- Setting directories i.e. for cache files
- Setting all kinds of other configuration options (select from array of options, true/false, integer with ranges, email addresses etc.)
- Defining default config values
- Titles and longer descriptions for config options
- Having mandatory and optional settings. Optional ones are skippable.

Installation

- Works with fucked up setups
  - Different users for Webserver vs. FTP / SSH

Interface

- Skinnable (optional)
- Wizard-based on multiple pages (optional)
- Command line (CLI) support
- Translations (optional)

Design

- See code...

The end

- Open questions?
- Further remarks?
- Contact
  - Mail: <kore@qafoo.com>
  - Web: `http://kore-nordmann.de` / `http://qafoo.com`
  - Twitter: @koredn / @qafoo