BsBuilder Documentation

Release 0.1.x

Tabaré Caorsi

June 04, 2012

CONTENTS

Contents:

CHAPTER

ONE

INTRODUCTION

1.1 What is BsBuilder?

BsBuilder is an automated build system which main purpose is deploying PHP applications with easy and in a consistent manner. It could be of course be used to build anything else but currently it's main purpose is to 'build' PHP applications.

BsBuilder not based or derived from any tool in particular but it takes ideas from all over the place: phing, ant, maven, gnumake, qmake (QT), and so on and so on. So thanks to all of those great tools.

1.2 A little bit of history repeating

Here at BinarySputnik we upload site updates **many** times a week (even many times a day) thus making the task of creating the tar files used for uploading a really dull, repetitive task.

Ever worse, we *made mistakes* while doing this, *"Forgot such or such file"*, *"Overwrote config file"*, and so on. This had to stop!

1.3 What do you mean by building PHP apps?

Ok, your mama may have told you that php apps are not build, that for the most of if, she is right. The thing is that you still need to update production servers, track version, upload big files (or small ones), etc.

We call the process of creating the "uploadable" files *build* and the files themselves *packages*.

1.4 What's next?

You can start looking around the code to figure out how BsBuilder works or you can go on and read: *Getting Started*

1.5 What the future holds

In future versions we plan to add:

1. Better error handling
2. More command line options

3. Better native support of the `bsbuild` executable
4. More tasks
5. Even more tasks
6. And a full automated distribution and deployment system

Stay tuned!

CHAPTER

TWO

SETTING BSBUILDER UP

2.1 System Requirements

BsBuilder requires:

1. PHP 5.3 > (CLI)
2. And that's it :)

Some tasks may require:

1. A tar executable in the PATH
2. A bzip executable in the PATH
3. A 7zr executable in the PATH
4. lib cUrl for PHP

2.2 Installing

Grab a copy from: *Downloads* and put is somewhere you can reach it. Make the `bsbuild` file executable (Linux only)

We are working to make this much easier and compatible with both Windows and Linux.

Don't get me wrong, BsBuilder works just fine wherever but you may have to run it as: `php bsbuild`.

CHAPTER

THREE

GETTING STARTED

3.1 XML

BsBuilder can only do what you tell it to, and to do that you have to *speak* XML.

The basic XML structure of you project's build file should include:

- One root element called `project`
- Any number of elements elements such as `property`
- At least one `target` element with any number of `task` sub elements

3.2 A Very Simple Example

We are going to build a project called, yes you know it, hello_world.

```xml
<?xml version="1.0" encoding="UTF-8"?>

<project name='hello_world' default='all'>

   <target name='all'>
      <tasks>
            <mkdir dirname='./build/all' />

            <copy source='.' dest='./build/all'>

            </copy>

            <package strategy='tar_bz2' name='hellod_workd.tar.bz2' dest='./build/all'  />

            <echo text='Package save as: ./build/all/hellod_workd.tar.bz2' />
      </tasks>
   </target>

   <target name='clean'>
      <tasks>
            <delete dirname='./build' />
      </tasks>
   </target>
</project>
```

First we define the project and it's name. And the most important part there is to define a `default` target which allows us to run bsbuild without arguments.

Then for this example we define two targets: `default` and `clean`.

The `default` target it a simple *copy all* and compress build. We add the `copy` task and instruct it to copy everything from the current directory to the build directory. We can *ignore* some files and patterns but we'll talk about that in a minute.

Then we have the `package` task which in this case, compresses everything from the `dest` folder into a file named `hellod_workd.tar.bz2`.

3.3 Run BsBuilder

To run bs builder to build the example project we just run: `bsbuild` without arguments from the folder where we have the build.conf.xml file.

CHAPTER

FOUR

COMMON TASKS

4.1 Property

The property task defines a property (kind of like a variable) that can be used later within the build file.

Note that all properties are *global*.

4.1.1 Example

```
<property name='basename' value='my-package' />
<property name='tarfile' value='${basename}.tar.bz2' />
<property name='sevenzip' value='${basename}.7z' />
<property name='target_dir' value='./build' type='prompt' />
<property name='passw' type='password' />
```

4.1.2 Attributes

Name	Type	Description	Default	Required
name	String	The of the property to export	No default value	True
value	String	The value for the property	No default value	True
type	String	The strategy to use. View Strategies	plain	False

4.1.3 Strategies

The strategies dictate the behavior of the task.

The ones available are `plain`, `prompt` and `password`.

`plain` just sets *value* to the property called *name*.

`prompt` asks the user for a value for the property *name* and if provided sets the property to that value, otherwise sets the property to *value*.

`password` is the same as prompt but instead of the actual text stars (`*`) are displayed.

NOTE: `password` works only on nix systems.

4.2 Echo

The echo task just prints a message to the screen.

4.2.1 Example

```
<echo text='' />
<echo text='' color='yellow' />
```

4.2.2 Attributes

Name	Type	Description	Default	Required
text	String	The text to print	No default value	True
color	String	The color to use. View colors	None	False

4.2.3 Colors

The available colors to use are:

1. black
2. dark_gray
3. blue
4. light_blue
5. green
6. light_green
7. cyan
8. light_cyan
9. red
10. light_red
11. purple
12. light_purple
13. brown
14. yellow
15. light_gray
16. white

4.3 Copy

This tasks handles the copy of one file or directory from one source to a destination. Right now it only supports *ignoring* certain files or patterns. In the future it will support filesets.

4.3.1 Example

```
<copy strategy='php' source='.' dest='./build/all' />
<copy source='./importantscript.php' dest='./build/all' />
<copy source='.' dest='${basedir}/all'>
      <ignore name='./test' />
      <ignore name='./build.conf.xml' />
      <ignore name='./.build_data' />
      <ignore name='./.gitignore' />
      <ignore name='./.git' />
</copy>
<copy source='.' dest='${basedir}/all'
      memory='on' memory_file='dist.mem'>
      <ignore name='./test' />
      <ignore name='./build.conf.xml' />
      <ignore name='./.build_data' />
      <ignore name='./.gitignore' />
      <ignore name='./.git' />
</copy>
```

4.3.2 Attributes

Name	Type	Description	Default	Required
source	String	The source file or directory to copy	No default value	True
dest	String	The destination directory	No default value	True
strategy	String	The strategy to use. View Strategies	php	False
memory	String	Either `on` or `off`	off	False
memory_file	String	Where to save the *memory*	No default value	False

4.3.3 Strategies

The strategies dictate the behavior of the task.

Right now the only one available is `php` which handles the copy using only PHP. In the future we will add more and you will have the ability to create your own.

4.3.4 Memory

The copy task has a special attribute named `memory`. If you turn this `on` prior copying the system will do 2 things: first it will check if the `memory_file` exists and if yes load it, and then it will check weather each file's m5d checksum is different from those saved in the `memory_file` and **only** copy those files that have actually changed from last build.

After copying it will save the updated `memory_file`.

This allows us to build a project, upload it, and if needed re build and upload only the difference from the latest build.

4.4 Replace

This tasks allows you to replace a portion of text of patter in one file.

4.4.1 Example

```
<replace value='db.password=1234'
         new_value='db.password=super_secret_password'
         file='./dist/all/configs/application.ini' />
<replace value='db.password=1234'
         new_value='db.password=super_secret_password'
         file='./dist/all/configs/application.ini.tpl'
         new_file='./dist/all/configs/application.ini' />
```

4.4.2 Attributes

Name	Type	Description	Default	Required
value	String	The value to search for	No default value	True
new_value	String	The string to replace `value` with	No default value	True
file	String	The file in which to perform this operation	No default value	True
new_file	String	The filename where the replaced content will be saved	Same value as file	False
strategy	String	The strategy to use. View Strategies	simple_replace	False

NOTE: if `file` is not found a warning is displays but nothing happens.

4.4.3 Strategies

The strategies dictate the behavior of the task.

Right now the only one available is `simple_replace` which uses php `str_replace` to do the replacing.

4.5 Package

This tasks packages the application.

4.5.1 Example

```
<package strategy='tar_bz2' name='file.tar.bz2' dest='./dist/all'  />
<package strategy='7z' name='file.7z' dest='./dist/all'  />
```

4.5.2 Attributes

Name	Type	Description	Default	Required
name	String	The filename for the package	No default value	True
dest	String	The destination directory. Which is the same as the source of the package	No default value	True
strategy	String	The strategy to use. View Strategies	tar_bz2	False

NOTE: if `file` is not found a warning is displays but nothing happens.

4.5.3 Strategies

The strategies dictate the behavior of the task.

You can user either `tar_bz2` or `7z`.

The `tar_bz2` strategy uses the system's tar binary to create a `.tar.bz2` package.

The `7z` strategy uses the system's 7zr binary to create a `.7z` package.

4.6 Version

This task increments a version number composed of MAJOR.MINOR.BUILD in the following manner:

if type is `build` only the BUILD part is incremented. If type is `major` only the MAJOR and BUILD parts are incremented. And finally, if type is `minor` only the MAJOR and MINOR parts are incremented.

You can pass an optional attribute instructing the task to export such version *number* as a property.

You also have to set a filename to save the version number.

4.6.1 Example

```
<version type='build' file='version.txt' property='version' />
```

4.6.2 Attributes

CHAPTER
FIVE

EXTRA TASKS

5.1 GitHub Upload

This task allows you to upload a given file to a GitHub repo *Downloads* area.

5.1.1 Example

```
<github_upload username='your_username'
               password='your_password'
               repository='your_repo'
               file='/path/to/file.tar.bz2' />

<github_upload username='${github_username}'
               password='${github_password}'
               repository='your_repo'
               file='${file}' />
```

5.1.2 Attributes

Name	Type	Description	Default	Required
username	String	The github usermane to use	No default value	True
password	String	The github password to use	No default value	True
repository	String	The github repository to upload to	No default value	True
file	String	The file to upload to github	No default value	True

5.1.3 Recommendations

It is highly recommended to use a `password` property task to prompt the user for the password instead of just writing it in the xml config file. Both ways are ok though.

CHAPTER

SIX

BSBUILDER COMMAND LINE

Right now the only support argument is the target name and a special *pseudo* target namd.

If you wish to run the default target just run

```
bsbuilder
```

If you wish to run on particular target called `foo` just run:

```
bsbuilder foo
```

And if you wish to see a list of available targets and ther (optional) description:

```
bsbuilder help
```

CHAPTER

SEVEN

DOWNLOADS

7.1 Latests Development Version

To get the most up to date version of BsBuilder you can clone our github repo from: `git://github.com/p1r0/BsBuilder.git`

7.2 Tarball

Latest version is: 0.1.39. And you can download it from Here

CHAPTER

EIGHT

LICENSE

BsBuilder is made by BinarySputnik and license under the GNU Lesser General Public License, version 3.0 (LGPL-3.0).

You may use it for whatever you want!

CHAPTER

NINE

INDICES AND TABLES

- *genindex*
- *modindex*
- *search*