Composer Packages

CJ Harries

1/19/15

Composer is a dependency manager for PHP. It allows developers to automatically load and use external code without manually moving and updating files. It is the core of many PHP frameworks, including Laravel.

With curl:

```
curl -sS https://getcomposer.org/installer | php
```

Without curl:

```
php -r "readfile('https://getcomposer.org/installer');" | php
```

Options are here.

With decent operating systems:

```
mv composer.phar /usr/local/bin/composer
```

(Note: on some versions of OSX you might also need to run `mkdir -p /usr/local/bin` before the above will work)

Without decent operating systems:

```
google.com
```

Easy way:

```
composer --help
```

Hard way:

```
php composer.phar --help
```

`composer.json` contains all of the information your project needs to describe itself and load dependencies. It's similar to RequireJS's `main.js` or Android's Gradle system. Your project needs one, your library needs one, and your external dependencies come with one. It lives in the root directory. If it's not there, Composer can't find it.

```
{
  "name": "sntai/core",
  "version": "0.0.00",
  "description": "Core AI functionality",
  "type": "library",
  "license": "Proprietary-SNT",
  "keywords": ["ai"],
  "authors": [
    {
      "name": "CJ Harries",
      "email": "cjharries@sntmedia.com"
    }
  ],
  "autoload": {
    "psr-4": {
      "SNTAI\\Core\\" : "src"
    }
  }
```

```json
{
  "name": "sntai/apifin",
  "version": "0.0.00",
  "description": "API AI FIN",
  "type": "library",
  "license": "Proprietary-SNT",
  "keywords": ["ai"],
  "repositories": [
    {
      "type": "vcs",
      "url": "https://github.com/passfail-cjharries/sntai_core"
    }
  ],
  "require": {
    "sntai/core": "dev-master"
  }
}
```

Like any configuration file, `composer.json` gives you the option to define

- Project information (author, version, license, etc.)
- Exports and configuration (autoloading, standards, unit tests, etc.)
- Dependencies (other composer packages, VCS, local files, etc.)

You don't have to ctrl+c/ctrl+v code.

Make a new directory for your project and organize your files in a way that makes sense, with a composer.json in the root directory.

```
/sntai_core
  /src
    AiCache.php
  /views
    SomeHtml.html
  /tests
    testAiCache.php
composer.json
```

Add source and tests to `composer.json` (along with any other exports)

```
{
  ...
  "autoload": {
    "psr-4": { "SNTAI\\Core\\": "src" }
  },
  "autoload-dev": {
    "psr-4: { "SNTAIT\\Core\\Tests\\" : "tests" }
  },
  ...
}
```

Name and version (if you like that sort of thing) it

```
{
    ...
    "name": "sntai/core",
    "version": "0.0.01",
    ...
}
```

(Optional) Alias the branches you want to create through `git`

```
{
  ...
  "extra": {
    "branch-alias": {
      "prod": "development"
    }
  },
  ...
}
```

Do the git thing

```
~/sntai_core$ git init
~/sntai_core$ git remote add origin https://
    github.com/passit/sntai_core.git
~/sntai_core$ git push -u origin master
```

You can also add your package to Packagist, but you probably shouldn't do that with proprietary code.

Navigate to the root directory and make a new `composer.json` with any package information you want to add.

```
// You've seen this before
```

You'll probably want to update your .gitignore to add the vendor directory. If you don't, you'll pollute your repos with external code.

```
vendor
/vendor/
vendor/*
# You have the power
```

Composer is incredibly powerful in that it can pull from Packagist, local sources, VCS, and basically anything you try to. Composer can even connect to your own private repos with some fiddling.

Packagist is the easiest to use.

```
{
  ...
  "require": {
    "monolog/monolog": "1.0.*"
  },
  ...
}
```

git isn't too hard either. Notice the dev- in dev-master.

```
{
  ...
  "repositories": [
    {
      "type": "vcs",
      "url": "https://github.com/passit/sntai_core"
    }
  ],
  "require": {
    "sntai/core": "dev-master"
  },
  ...
}
```

Private repos work as well (with some tweaking and potentially some ssh-fu). Notice the `dev-` in `dev-master`.

```
{
  ...
  "repositories": [
    {
      "type": "vcs",
      "url": "git@github.com:passit/
          sntai_core.git"
    }
  ],
  "require": {
    "sntai/core": "dev-master"
  },
  ...
}
```

Navigate to the root directory of your project.
First time:

```
~/sntai_core$ composer install
```

Any other time:

```
~/sntai_core$ composer update
```

Hard way:

```
~/sntai_core$ php composer.phar install/update
```

- Using Composer from the command line is very powerful.
- The `composer.json` schema has tons of options.
- The internet has some nice things to say about Composer.

- Composer docs
- StackOverflow
- Google