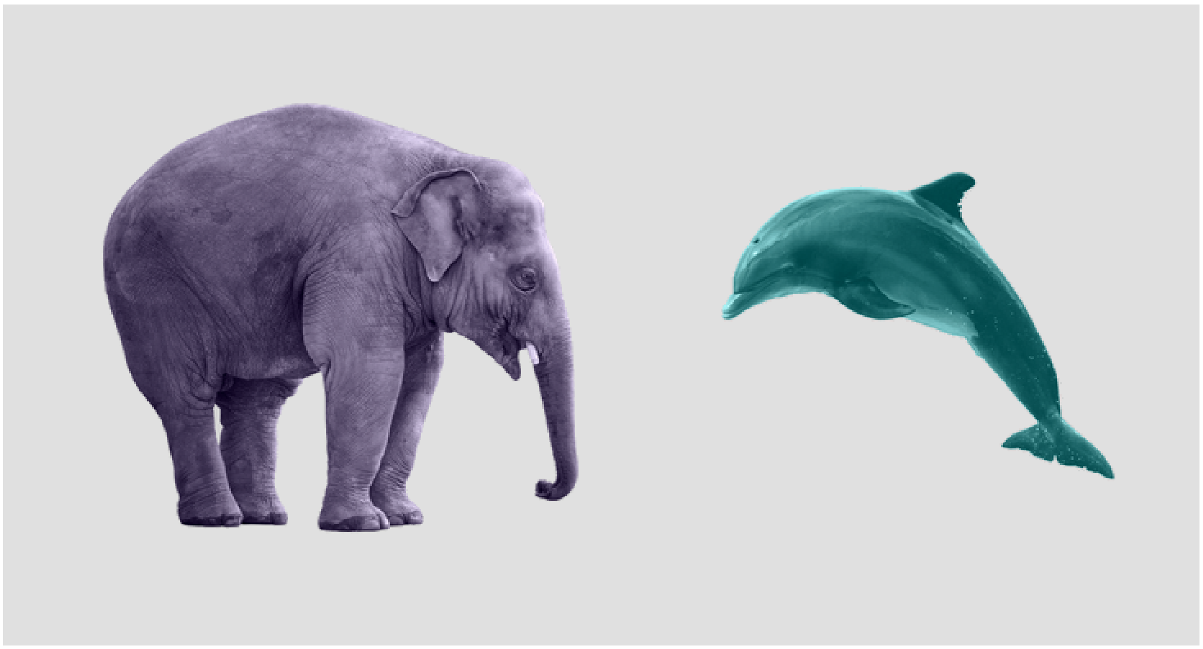
How to Document PHP and MySQL Project

Wojciech Mleczek 1 year ago Database Documentation · MySQL · PHP · Software Development



My Product Manager said "write a documentation"

For many of us it's the most annoying sentence we could ever hear. Project is just finished, everything works great, but then comes the moment when you have to document all the stuff you coded for the past months. Can it get worse than that? If you said "no, writing a documentation is the worst thing ever" then definitely this article is a must-read for you!

Applying those 3 principles leads to success

Before we start doing some magic I have to tell you about 3 gold principlels which every PHP programmer should follow:

This syntax is widely supported by many IDE's and tools - the only possible solution. In some cases you could use libraries which have defined custom tags (e.g. using Swagger about which I previously wrote an article).

```
/**
  * Add two numbers.
  *
  * @author Wojciech Mleczek
  * @param int $x
  * @param int $y
  * @return int Result of x+y operation.
  */
function add($x, $y)
{
    return $x + $y;
}
```

2. Document as you go

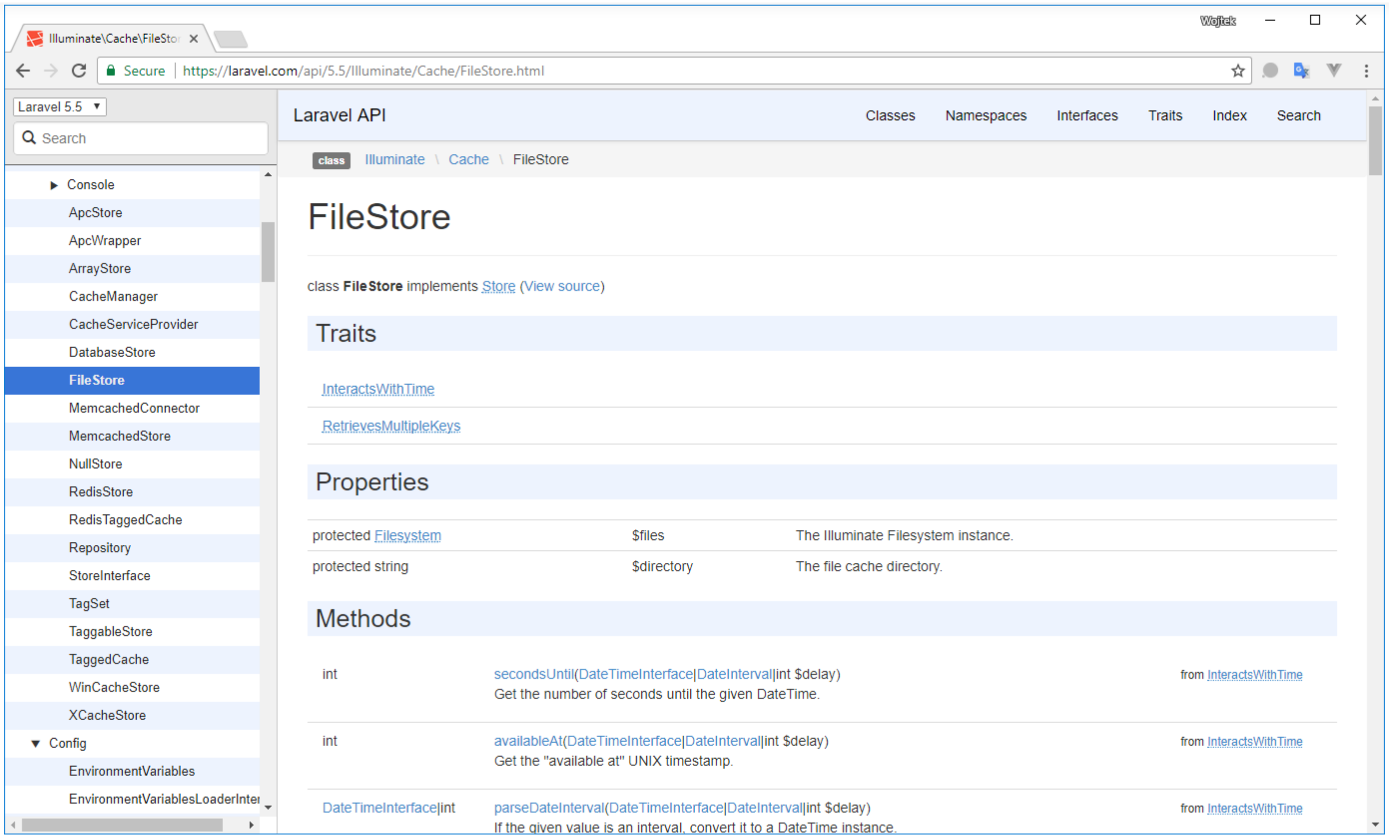
Whenever you think "this information should be in the documentation" you have to put it there immediately, otherwise you will probably never do this. The bigger the backlog in the documentation, the greater the unwillingness to write it.

3. DRY (Don't Repeat Yourself)

Automation is the basis of success, who likes to repeat exactly the same thing over and over again? This is best seen in the example of the class diagram we could prepare at the beginning of the project. If we don't have a tool that will synchronize our existing class diagram with code changes, then the diagram sooner or later will be out of date. Few people like to apply the same changes in 2 places, and even if someone do it then he can just forget to copy something or make a mistake while moving changes.

This type of documentation **can and should be fully automated**. Nowadays there are plenty of tools to generate a such documentation and their usage is as simple as possible. I'll show you how to use the FriendsOfPHP/sami generator which is used by Symfony and Laravel framework. If you're looking for alternatives to the Sami generator then you could check the phpDox or phpDocumenter.

Dataedo



FileStore

class **FileStore** implements Store (View source)

Traits

InteractsWithTime

RetrievesMultipleKeys

Properties

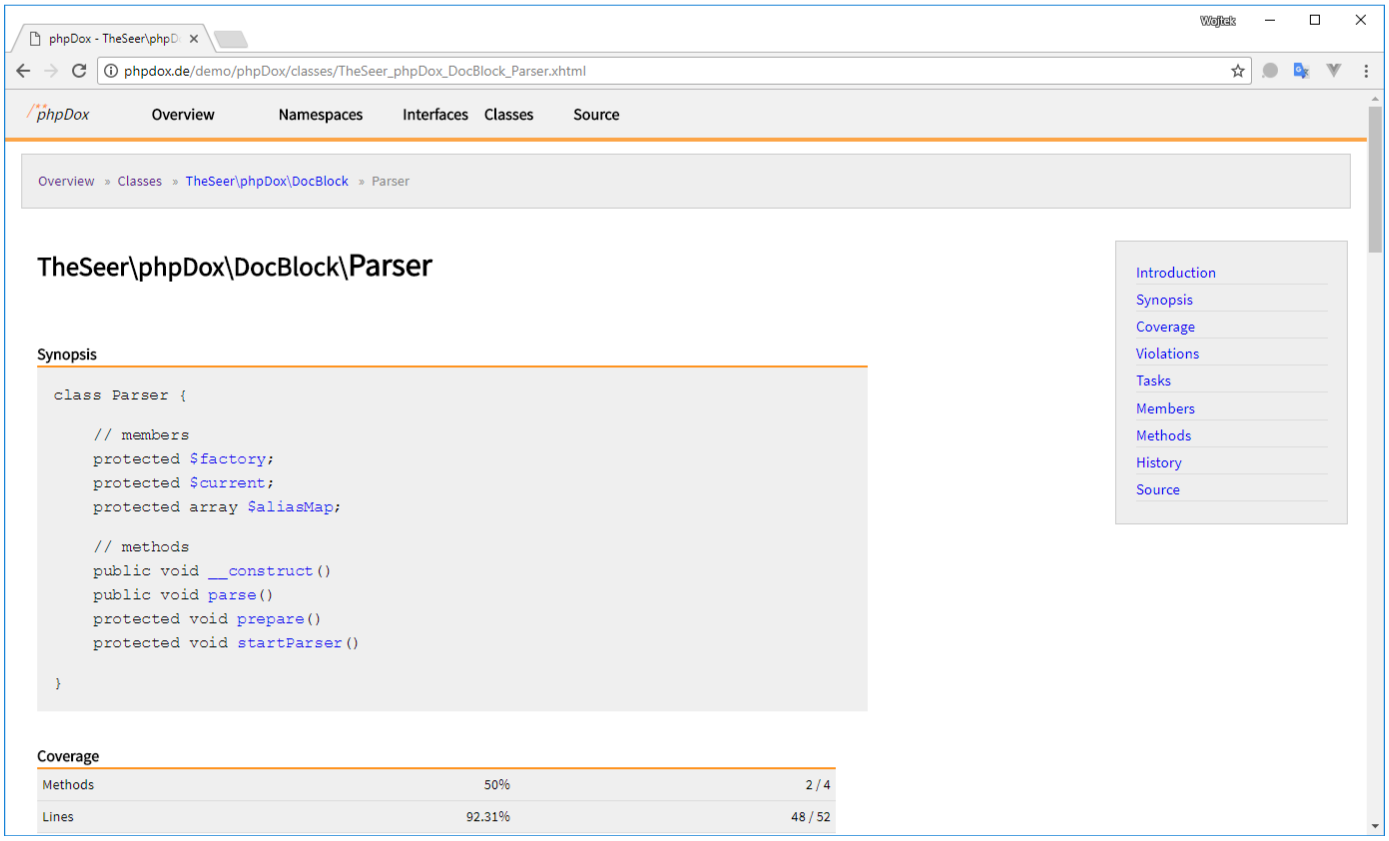
protected Filesystem	$files	The Illuminate Filesystem instance.
protected string	$directory	The file cache directory.

Methods

int	secondsUntil(DateTimeInterface|DateInterval|int $delay) Get the number of seconds until the given DateTime.	from InteractsWithTime
int	availableAt(DateTimeInterface|DateInterval|int $delay) Get the "available at" UNIX timestamp.	from InteractsWithTime
DateTimeInterface|int	parseDateInterval(DateTimeInterface|DateInterval|int $delay) If the given value is an interval, convert it to a DateTime instance.	from InteractsWithTime

phpDox:

http://phpdox.de/demo/phpDox/classes/TheSeer_phpDox_DocBlock_Parser.xhtml



TheSeer\phpDox\DocBlock\Parser

Synopsis

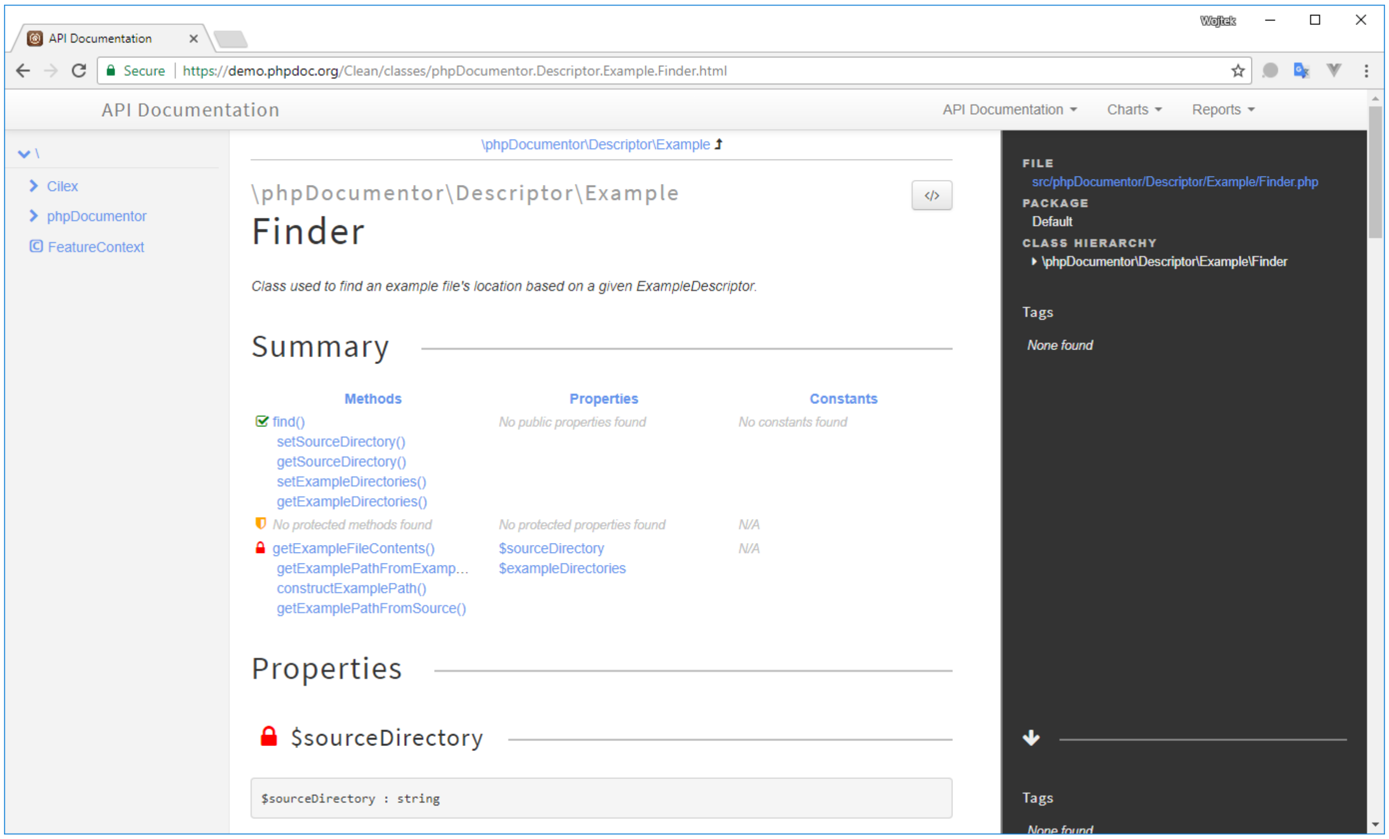
```
class Parser {

    // members
    protected $factory;
    protected $current;
    protected array $aliasMap;

    // methods
    public void __construct()
    public void parse()
    protected void prepare()
    protected void startParser()

}
```

Coverage

Methods	50%	2 / 4
Lines	92.31%	48 / 52

phpDocumentor:

https://demo.phpdoc.org/Clean/classes/phpDocumentor.Descriptor.Example.Finder.html



\phpDocumentor\Descriptor\Example

\phpDocumentor\Descriptor\Example

Finder

Class used to find an example file's location based on a given ExampleDescriptor.

Summary

Methods	Properties	Constants
find() setSourceDirectory() getSourceDirectory() setExampleDirectories() getExampleDirectories()	*No public properties found*	*No constants found*
No protected methods found	*No protected properties found*	*N/A*
getExampleFileContents() getExamplePathFromExamp... constructExamplePath() getExamplePathFromSource()	$sourceDirectory $exampleDirectories	*N/A*

Properties

$sourceDirectory

```
$sourceDirectory : string
```

FILE
src/phpDocumentor/Descriptor/Example/Finder.php
PACKAGE
Default
CLASS HIERARCHY
\phpDocumentor\Descriptor\Example\Finder

Tags
None found

Tags
None found

Ok, it's time to do some work in practise. Our example will be based on the Deplink project, but you could also choose almost any project from the GitHub trending PHP repositories or your project on which you're currently working on.

Open command line and create new directory dedicated for this sample:

```
mkdir sami-sample && cd sami-sample
```

Clone your project (in my case it's an example project from GitHub repository):

```
git clone https://github.com/deplink/deplink
```

```
curl -O http://get.sensiolabs.org/sami.phar
```

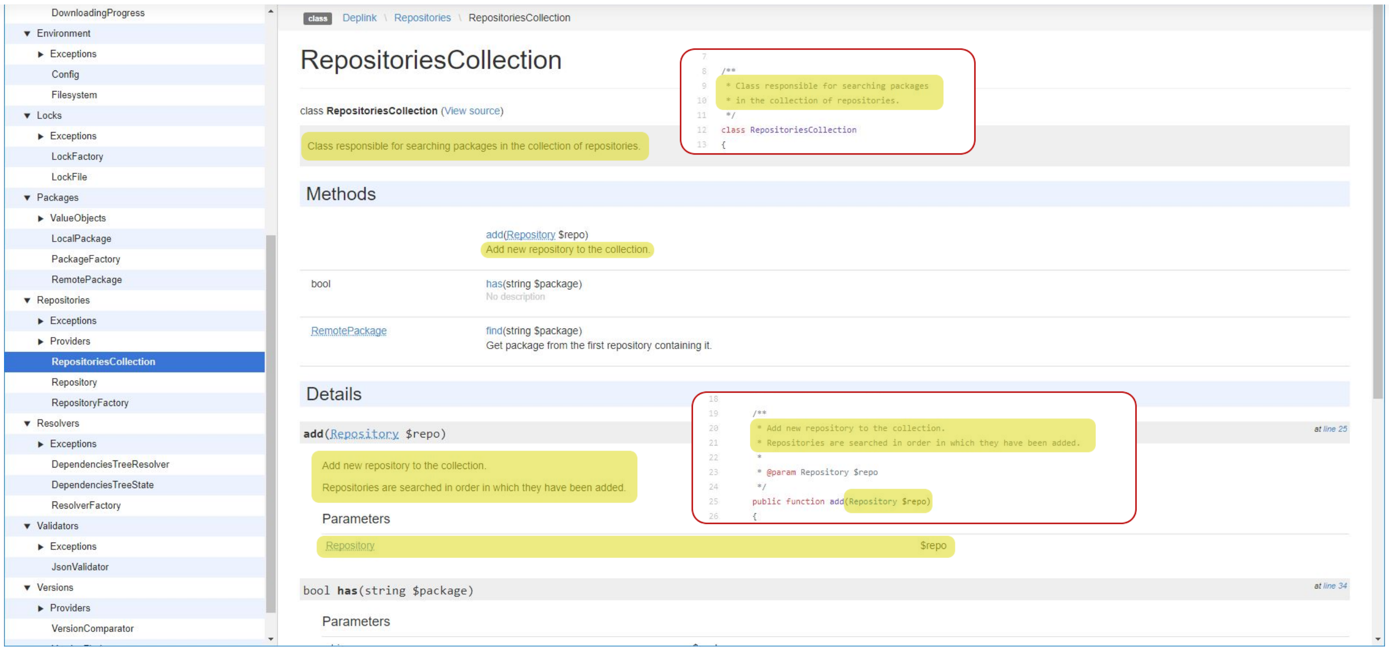
Write basic Sami configuration in the `sami-sample/config.php` file:

```php
<?php

// Point where Sami should search for PHP source code.
$srcDir = 'deplink/src';

return new Sami\Sami($srcDir);
```

Now, let's generate documentation from our source code:

```
php sami.phar update config.php
```

Done. To view it open `build/index.html`and browse objects in the navigation pane on the left. Sample class view:



DownloadingProgress
▼ Environment
▶ Exceptions
Config
Filesystem
▼ Locks
▶ Exceptions
LockFactory
LockFile
▼ Packages
▶ ValueObjects
LocalPackage
PackageFactory
RemotePackage
▼ Repositories
▶ Exceptions
▶ Providers
RepositoriesCollection
Repository
RepositoryFactory
▼ Resolvers
▶ Exceptions
DependenciesTreeResolver
DependenciesTreeState
ResolverFactory
▼ Validators
▶ Exceptions
JsonValidator
▼ Versions
▶ Providers
VersionComparator

class Deplink \ Repositories \ RepositoriesCollection

RepositoriesCollection

```
/**
 * Class responsible for searching packages
 * in the collection of repositories.
 */
class RepositoriesCollection
{
```

class **RepositoriesCollection** (View source)

Class responsible for searching packages in the collection of repositories.

Methods

	add(Repository $repo) Add new repository to the collection.
bool	has(string $package) No description
RemotePackage	find(string $package) Get package from the first repository containing it.

Details

```
add(Repository $repo)
```

at line 25

```
    /**
     * Add new repository to the collection.
     * Repositories are searched in order in which they have been added.
     *
     * @param Repository $repo
     */
    public function add(Repository $repo)
    {
```

Add new repository to the collection.

Repositories are searched in order in which they have been added.

Parameters

Repository	$repo

```
bool has(string $package)
```

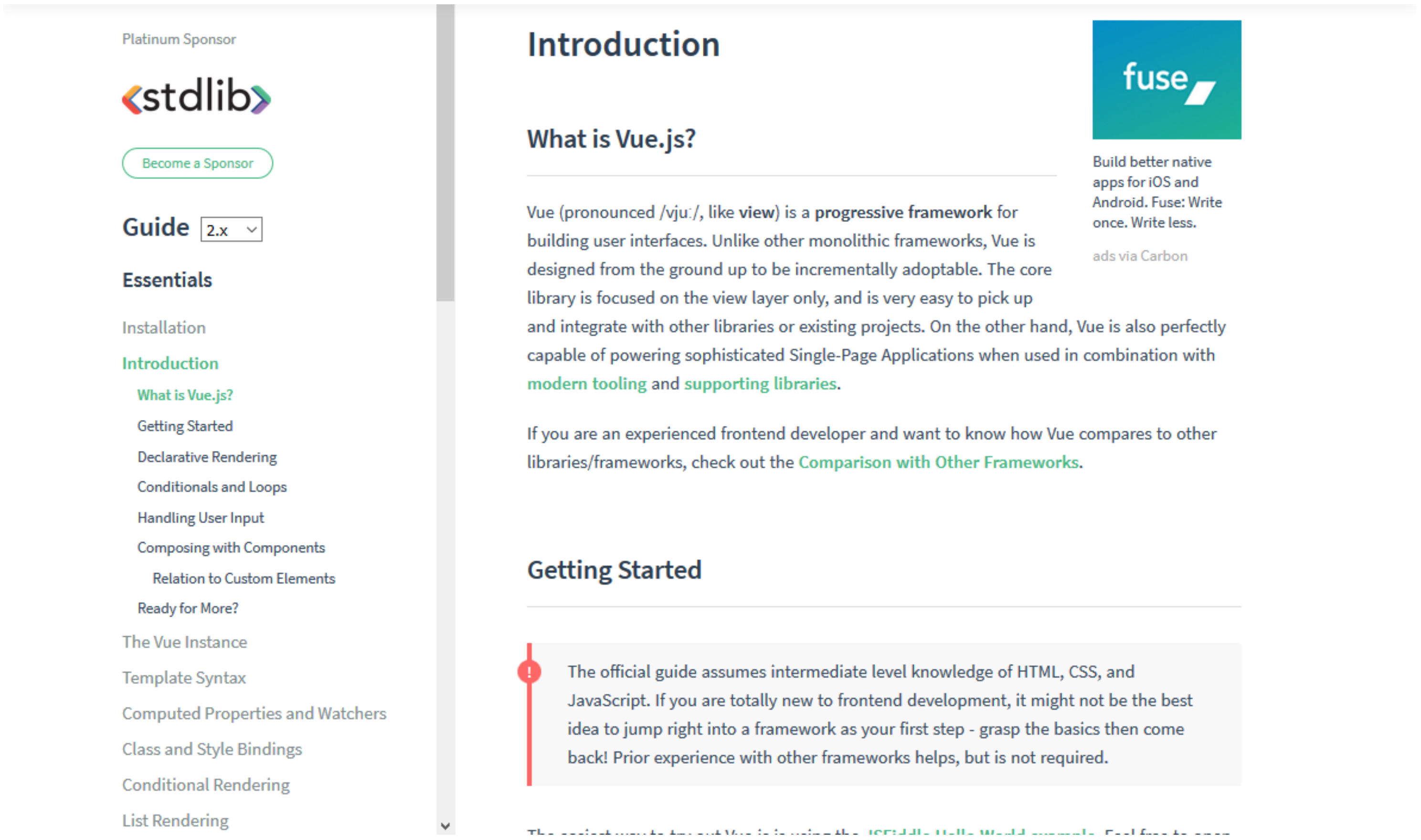
at line 34

Parameters

Generating wiki documentation

In my opinion this kind of documentation is the most important for most of projects (not only PHP related), but can be only partially automated (our task is to write the content). Let's have look at sample documentations online:

Vue Documentation



Platinum Sponsor

stdlib

Become a Sponsor

Guide 2.x

Essentials

- Installation
- Introduction
  - What is Vue.js?
  - Getting Started
  - Declarative Rendering
  - Conditionals and Loops
  - Handling User Input
  - Composing with Components
    - Relation to Custom Elements
  - Ready for More?
- The Vue Instance
- Template Syntax
- Computed Properties and Watchers
- Class and Style Bindings
- Conditional Rendering
- List Rendering

Introduction

What is Vue.js?

Vue (pronounced /vjuː/, like **view**) is a **progressive framework** for building user interfaces. Unlike other monolithic frameworks, Vue is designed from the ground up to be incrementally adoptable. The core library is focused on the view layer only, and is very easy to pick up and integrate with other libraries or existing projects. On the other hand, Vue is also perfectly capable of powering sophisticated Single-Page Applications when used in combination with modern tooling and supporting libraries.

If you are an experienced frontend developer and want to know how Vue compares to other libraries/frameworks, check out the Comparison with Other Frameworks.

fuse

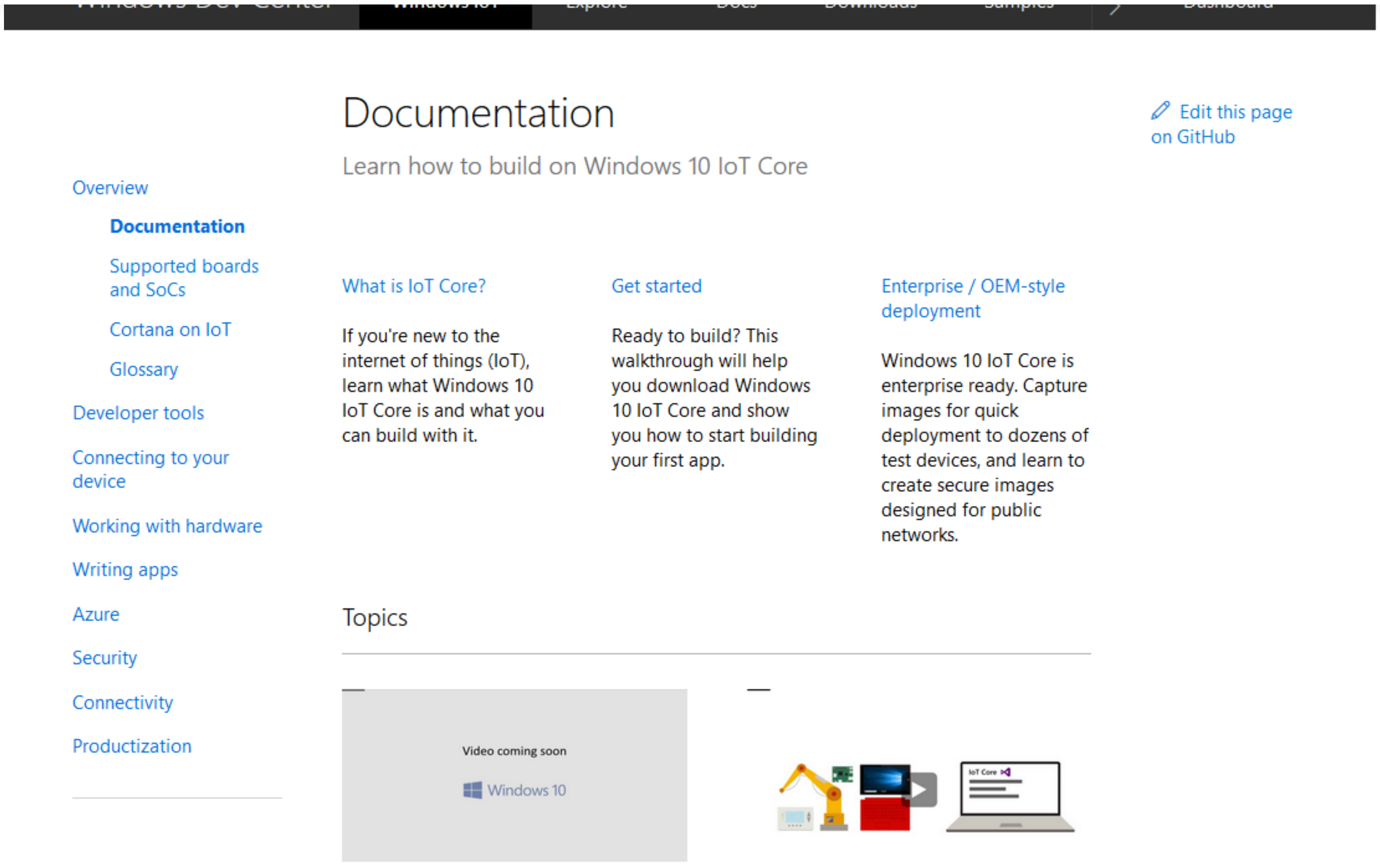
Build better native apps for iOS and Android. Fuse: Write once. Write less.

ads via Carbon

Getting Started

The official guide assumes intermediate level knowledge of HTML, CSS, and JavaScript. If you are totally new to frontend development, it might not be the best idea to jump right into a framework as your first step - grasp the basics then come back! Prior experience with other frameworks helps, but is not required.

Windows 10 IoT Core Official Website



Documentation

Learn how to build on Windows 10 IoT Core

Edit this page on GitHub

- Overview
  - **Documentation**
  - Supported boards and SoCs
  - Cortana on IoT
  - Glossary
- Developer tools
- Connecting to your device
- Working with hardware
- Writing apps
- Azure
- Security
- Connectivity
- Productization

What is IoT Core?

If you're new to the internet of things (IoT), learn what Windows 10 IoT Core is and what you can build with it.

Get started

Ready to build? This walkthrough will help you download Windows 10 IoT Core and show you how to start building your first app.

Enterprise / OEM-style deployment

Windows 10 IoT Core is enterprise ready. Capture images for quick deployment to dozens of test devices, and learn to create secure images designed for public networks.

Topics

Do you see any similarities? Both pages are a markdown file on GitHub as evidenced by the message at the bottom of the page:

Microsoft:

> Edit this page on GitHub

Vue:

> Caught a mistake or want to contribute to the documentation? Edit this page on Github!

Using this solution you can make that whenever you edit markdown file and push a commit then a new documentation will be generated and uploaded to your web server. It's an amazing possibility to eliminate repeatable work, reduce human errors and gain time for programmers to work on the other stuff.

Create and go to the new working directory:

```
mkdir hexo-sample && cd hexo-sample
```

Install Hexo CLI globally (Node.js with NPM required):

```
npm install hexo-cli -g
```

Init new Hexo project in current directory:

```
hexo init .
```

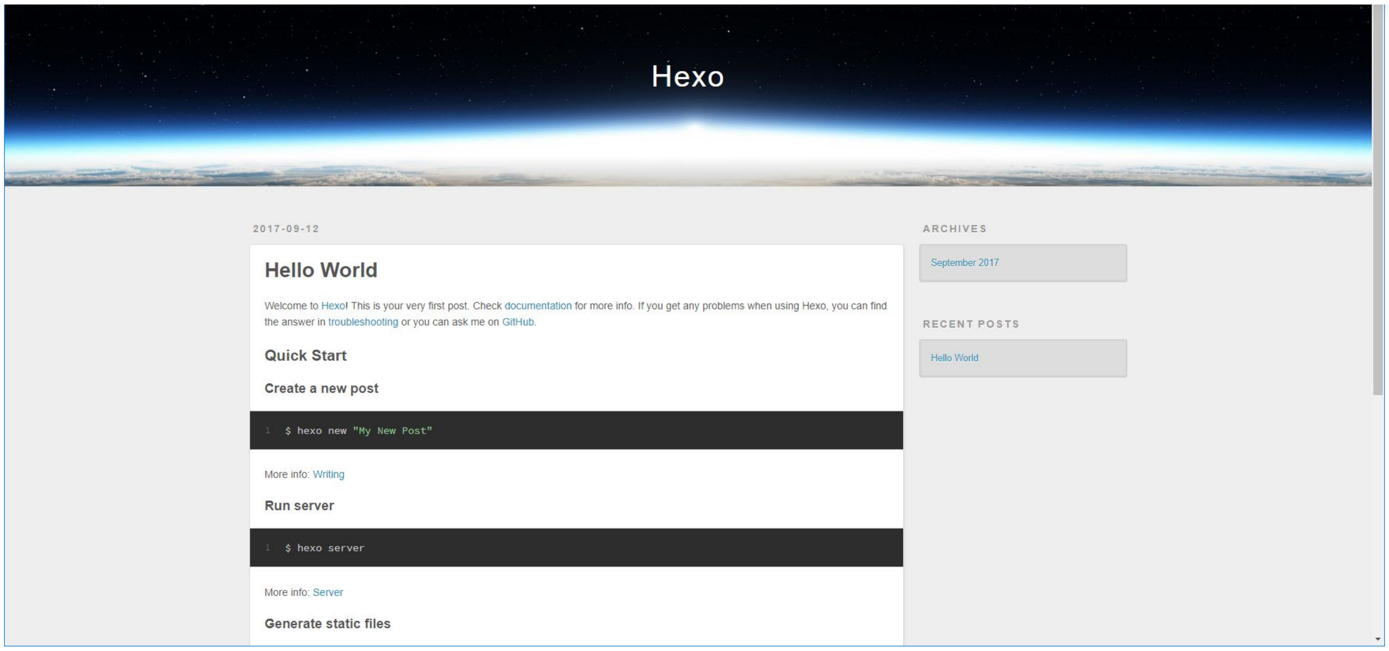
Run server and go to **http://localhost:4000/** to see your new empty project:

```
hexo server
```

Or simply build project without running server. This command generates HTML files which you can upload to your server:

```
hexo generate
```

After every modification to markdown files you can freely use `generate` command to update the HTML files.



2017-09-12

Hello World

Welcome to Hexo! This is your very first post. Check documentation for more info. If you get any problems when using Hexo, you can find the answer in troubleshooting or you can ask me on GitHub.

Quick Start

Create a new post

```
$ hexo new "My New Post"
```

More info: Writing

Run server

```
$ hexo server
```

More info: Server

Generate static files

ARCHIVES

September 2017

RECENT POSTS

Hello World

The page content you see is located in a `hexo-sample\source\_posts\hello-world.md` file. Content of this file is in markdown with frontmatter header. Markdown is a simplified language that enables creating formatted HTML documents using plain text. Try to edit it, markdown syntax and front matter basic information could be helpful to get started. Have fun while exploring new possibilities of the Hexo static HTML generator!

You can be a bit confused that this is an example of the blog. Don't be afraid, it's possible to change it to standard wiki articles grouped in categories, but you have to do it manually by modyfing theme. Once you prepare a theme you can reuse it in any project and focus only on content.

Generating documentation of database

Database documentation is often missing for PHP application. The reasons could be a few:

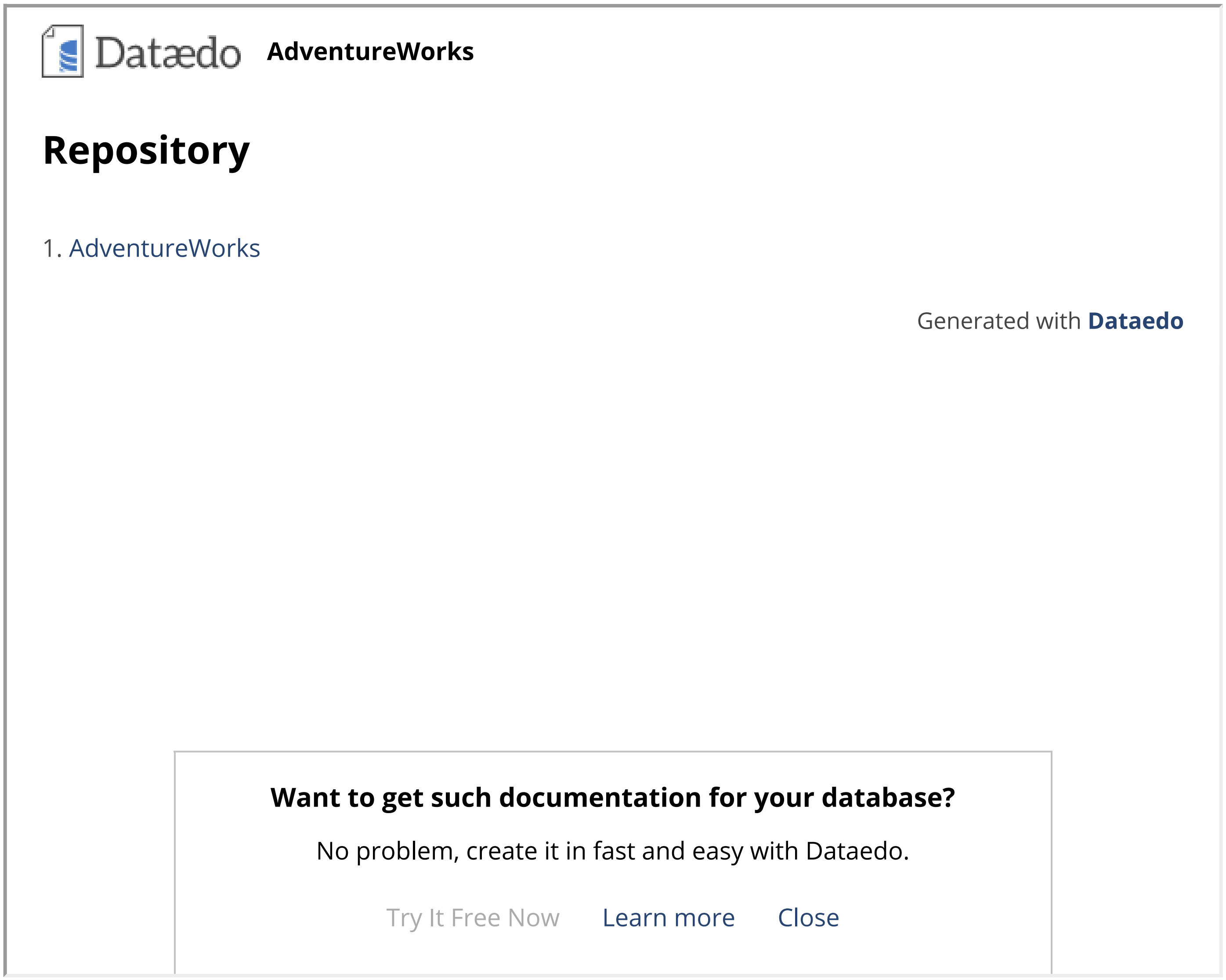
- we've documented ORM models,
- we've documented PHP migrations,
- we think that columns are self-describing,
- or simply lack of time

to do it without all this unnecessary introductions.

Download, install and run Dataedo. If you have any issues with one of the steps then please click on it and follow instruction contained therein. You shouldn't have any problems to download and install Dataedo as well as creating first file repository (which is simply a file that will be used to store documentation for your database).

For us the **"3. Connect To Your Database"** and **"7. Export To HTML (Pro feature)"** section in the "How to Document MySQL Database in 5 Minutes with Dataedo [Free Tool]" article will be important. Follow the steps in both of this sections to produce HTML like below one:

Open sample documentation



Dataedo **AdventureWorks**

Repository

1. AdventureWorks

Generated with **Dataedo**

Want to get such documentation for your database?

No problem, create it in fast and easy with Dataedo.

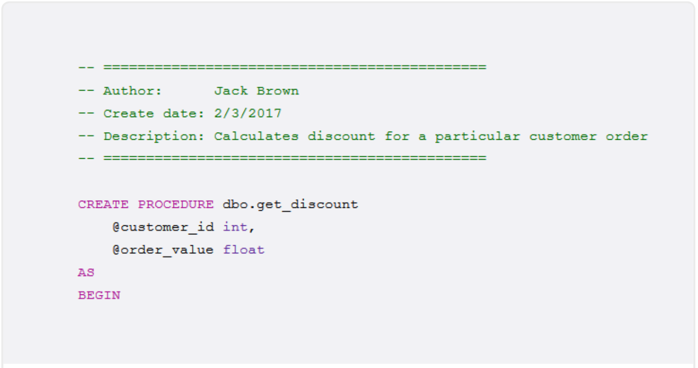
Maybe at least I convinced you that documenting code doesn't have to be so terrible. I keep my fingers crossed that next time it's up to you to surprise the boss saying that the documentation is ready. Good luck!

There are no comments. Click here to write the first comment.

Recommendations

PRODUCTS AND NEWS

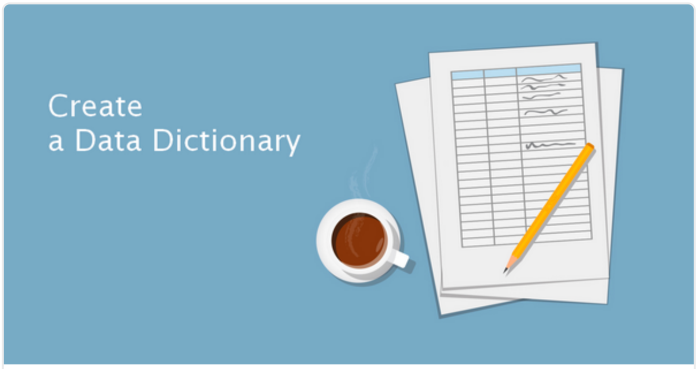
How to document REST API project written in PHP using Swagger and Dataedo



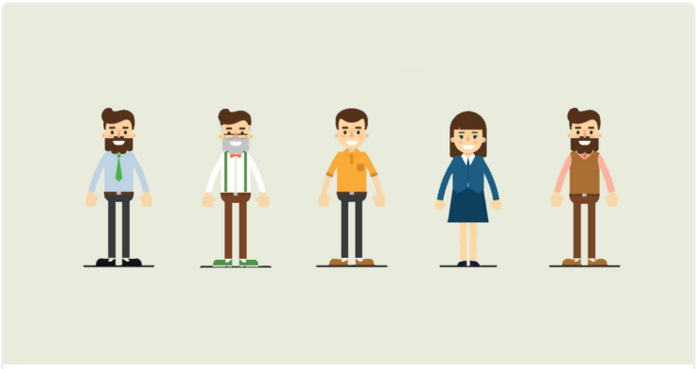
```
-- ============================================
-- Author:      Jack Brown
-- Create date: 2/3/2017
-- Description: Calculates discount for a particular customer order
-- ============================================

CREATE PROCEDURE dbo.get_discount
    @customer_id int,
    @order_value float
AS
BEGIN
```

How to Document Stored Procedures and Functions in Database

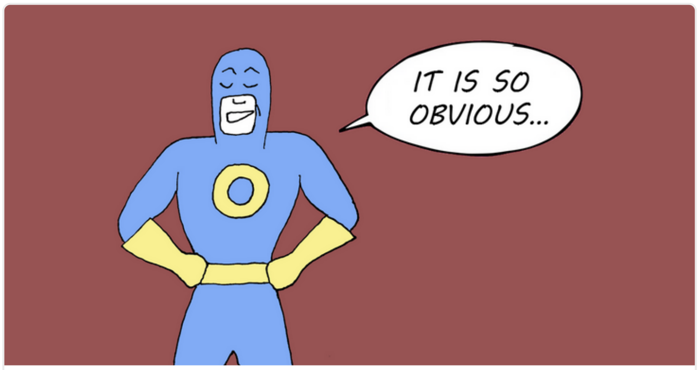
Extended Properties for Database Documentation - Good or Bad?



Create a Data Dictionary with Dataedo



Who do You Document Databases For



Captain Obvious' Guide to Column Descriptions - Data Dictionary Best Practices

Dataedo

Subscribe our blog and knowledge base

Get new articles and tips on modeling, designing and documenting databases to your inbox.

Subscribe