

Robert Lemke

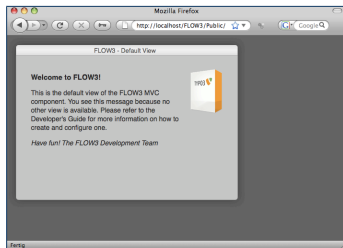
Getting into FLOW3

First steps within the new PHP framework

The development of FLOW3, the new basis for the next TYPO3 generation continues to advance. A first release for the use in individual projects is planned for this year. Intensive work is currently being done on the most important precondition for this, namely the development of the persistence layer and the content repository. This is reason enough to take a look at the development of FLOW3 and to hazard some first experiments.

Currently the code basis of FLOW3 [1] is undergoing several changes on a daily basis. To avoid disappointment for inexperienced users the project team has decided not to offer any complete packages for downloading. Thus the FLOW3 source-code is accessed via the sub-version repository [2].

Download and Installation



Following successful installation FLOW3 greets its developers with its default view.

Should something go wrong, the FLOW3 developers on the TYPO3 5.0 mailing list will gladly offer further assistance [4]. First create an index within your web root in which you would like to install FLOW3. Then download FLOW3 with the assistance of the sub-version client.

```

SHELL
$ mkdir /opt/local/www/FLOW3
$ svn checkout https://svn.typo3.org/FLOW3/Distribution/trunk /opt/local/www/FLOW3/
Ausgecheckt, Revision xxxx.
    
```

Listing 1

Since FLOW3 will write some files to the „Public/ index“, you need to ensure that this is also writable for other users who open FLOW3. On your test server you can do this with the following command:

```

SHELL
$ chmod -R 777 /opt/local/www/FLOW3/Public
    
```

Listing 2

Next, confirm that FLOW3 was successfully installed. Depending on how you have configured your web server, you should see the standard view of the MVC framework at the address <http://localhost/FLOW3/Public> (see illustration).

Configuring the Web-server

If FLOW3 is to be used in a productive environment later on (you should wait for an official release), we recommend that you set the root index of the web server or virtual host to the „Public“ index of FLOW3. This will have the result that all source code and files which will eventually belong in the file system will be accessible from outside, with the ex-

ception of „index.php“ and the resources reflected by FLOW3 into the open index. In addition, for the correct termination of virtual path names, you will need the web server module „mod_rewrite“ (Apache) or „ISAP_rewrite“ (ISS). FLOW3 comes with an htaccess file with the necessary settings for Apache.

Hello World

Before examining some of the mechanisms of FLOW3 in detail we will create a Hello World application with FLOW3. FLOW3 is a package-based system. Individual applications and extensions are bundled into packages. A package may contain several different files, including documents, translations, images and other resources and of course PHP classes. You can make a new package by creating a sub-index in the „Packages“ index. For their part, PHP classes are found in a sub-index called „Classes“. The file names correspond exactly to the class names.

Create a new PHP file and the corresponding indexes. The class will be called „F3_Demo_Controller_Default“:

```

SHELL
$ cd /opt/local/www/FLOW3/
$ mkdir -p Packages/Demo/Classes/Controller
$ touch Packages/Demo/Classes/Controller/F3_Demo_Controller_Default.php
    
```

Listing 3

You need not take these steps on the command line, rather you can take them in the comfort of your own development environment. Now fill the empty PHP file with a few lines of code which communicate the „Hello World“:

```

PHP
<?php
class F3_Demo_Controller_Default extends
F3_FLOW3_MVC_Controller_ActionController {
    public function defaultAction() {
        return 'Hello World!';
    }
}
    
```

Listing 4

When you (and the core developers) have done everything right you will obtain a copy of the requested edition at the address: <http://localhost/FLOW3/Public/Demo>.

A Question of Context

For different situations FLOW3 can define different configurations which are bundled into a so-called „Application Context“. Contexts which occur often in projects include productive application, development and (automated) testing. But different environments, such as numerous live servers or development calculators,

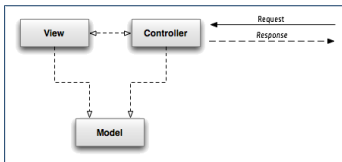
may require different configurations. FLOW3 knows the contexts „Production“, „Development“ and „Testing“ and comes with sensible settings for the various applications. Thus in the standard „Production“ context all caches are active and many program parts are optimized for good performance. During development many of these caches would get in the way, which is why many of them are switched off in the „Development“ context.

You can start the FLOW3 applications each time in different contexts by creating your own „index.php“. Currently the distribution contains a script for opening the „Development“ context: simply open the Address <http://localhost/>. FLOW3/Public/index_dev.php instead of the URL <http://localhost/FLOW3/Public/> and the configuration of the „Development“ context will be active.

When developing keep the development context steadily in mind; otherwise you will obtain unexpected results, since a lot of information about the various classes is stored in the „Production“ context.

Modelviewcontroller

Anyone listening to the conversations of PHP developers recently would have formed the impression that „Modelviewcontroller“ has become part of the standard vocabulary. While the MVC template has become one of the most important principles of neatly structured applications, there is hardly a design template that permits so many variations in its use. Thus the claim that „I use MVC“, does not say very much.



The design patten 'Model-View-Controller'

FLOW3 applies a so-called front controller, which takes up all requests – from the web or the command line. With various analysis and routing mechanisms it is then determined which controller will be finally responsible for answering the query.

Normally this will be an action controller (such as in our Hello World example) that supporting several actions which are implemented as methods in the controller class.

Depending on the action requested, the controller will select a suitable model. Of these three a model is the most natural object, one that will be reflected in the real world. Thus, for example, it may be a model of a client or an event. This model is then passed on by a controller to a view with the request to render a presentation customized to the model and the situation. Finally the controller returns the rendered view to the front controller in a reply.

An example of the incorporation of views and the attendant conventions for the naming of indexes and classes may be found in the FLOW3 handbook [5].

Arguments und Validation

In the end almost every application wants to communicate with the user and process his or her entries. Online this normally occurs through the exchange of GET or POST parameters which are sent via formulas. However, the high number of security bulletins demonstrates that this is often a weak point of web-based applications: unfiltered entries through Cross-Site Scripting (XSS) or SQL injection can throw up weighty security issues. FLOW3 normally denies access to GET and POST parameters and only passes on filtered versions to the controller. All expected arguments must be pre-registered and unknown parameters are ignored. In addition, the expected data type is defined for each argument so that validation can occur in the foreground. Try it by changing the code in the Hello World example as follows:

PHP

```
F3_Demo_Controller_Default.php
<?php
class F3_Demo_Controller_Default extends
F3_FLOW3_MVC_Controller_ActionController {
    public function initializeController() {
        $this->arguments->addNewArgument('firstName', 'Text');
    }
    public function defaultAction() {
        return (string)$this->arguments['firstName'];
    }
}
?>
```

Listing 5

When you open the address http://localhost/FLOW3/Public/index_dev.php/Demo?firstName=Robert in your browser you should be able to see the output „Robert“ (providing that you installed FLOW3 as described above).

As you can see from the listing, the argument ‘firstName“ is registered and declared as „Text“. Have a go with other data types (eg „Number“ or „EmailAddress“) and experiment with different parameter values.

Of course the definition of arguments is just a first step, as the validation presented here is only part of a mighty filter and validation framework, which supports not only the built in validator but also your own. A number of such validators may be linked together and used again so that more complex controls become possible. But alongside validation, authentication and authorization are important aspects of the development of web applications.

Security Aspects

Andreas Förthner is currently developing a comprehensive security framework. What makes the FLOW3 solution special is that security functions are implemented and directed centrally, rather than having to be considered at many different places as previously.

For your action controller this means that you will no longer have to check whether there is a registered user who is part of a group which allows him or her to perform the function requested. You can develop your action method without having to write a single line of security-relevant code. Instead the security framework will incorporate the necessary checks into your code with FLOW3’s AOP function (aspect-oriented programming). With this approach you will also have the option of retrospectively protecting program parts which were not intended for protection by the author. Similarly the theme SQL injection does not appear – because no more SQL is being used.

Persistence

Surely the most eagerly awaited FLOW3 feature has been the persistence framework and the content repository belonging to it. For only when this has been completed will it be possible to store and recall files in a simple manner. But how do you access your files when there is no more SQL? The answer is not at all: the files come to you. FLOW3 offers special support for domain-driven design (see T3N 11) which allows the developer to concentrate fully on the domain, ie on the business logic. A FLOW3 developer therefore thinks and designs with objects rather than database tables – surely no easy transition but one that is well worthwhile. Take the example of a blog entry. A PHP class representing a blog entry could look like this:

```

PHP
/**
 * @entity
 */
class F3_Blog_Domain_Post {
    /**
     * @var string
     */
    protected $title;

    /**
     * @var DateTime
     */
    protected $date;

    /**
     * @var array
     * @reference
     */
    protected $tags = array();

    /**
     * Setter for title
     *
     * @param string $title
     * @return void
     */
    public function setTitle($title) {
        $this->title = $title;
    }

    /**
     * Adds a tag to the post
     * @param F3_Blog_Domain_Tag $tag
     * @return void
     */
    public function addTag(F3_Blog_Domain_Tag $tag) {
        $this->tags[] = $tag;
    }
}
?>

```

Listing 6

Of course there will be further setter and getter methods as well as additional functions such as „publish()“ which will allow a blog entry to be published. In accordance with domain-driven design we gather our blog entries into a blog-entry repository, because that’s precisely what our PostRepository is:

```

PHP
/**
 * @repository
 */
class F3_Blog_Domain_PostRepository {

    /**
     * @var array
     * @reference
     */
    protected $posts = array();

    public function add(F3_Blog_Domain_Post $post) {
        $this->posts[] = $post;
    }

    public function remove($post) { ... }
    public function findByDate($date) { ... }
}

```

```

...
}

```

Listing 7

This repository serves to retrieve blog entries at a later stage: to this end there are corresponding „find*()“ methods which return posts following a particular criterion. So far so good: this is not a very exciting code. But that’s all you need to save blog entries. FLOW3 recognizes that your blog entry is an object that can persist (through the @entity annotation at the beginning of the class). But that alone will not ensure that your new post object will persist too. You still have to add it to a suitable repository – the PostRepository – by opening its „add()“ method. FLOW3 recognizes all repositories by their „@repository“ annotation and ensures that its entries are saved and restored when you need them.

At the End of the Year

Right now that’s the answer to the question when FLOW3 can be used for individual projects. Until then the development team will spend many sleepless nights grappling with intricacies for the simple reason that they can hardly wait to work with it themselves.

Links and Literature

 [Softlink 2176](#)

- [1] FLOW3 Website: <http://flow3.typo3.org>
- [2] Subversion Repository: <http://svn.typo3.org/FLOW3/Distribution/trunk>
- [3] Tipps zur Installation: <http://flow3.typo3.org/download/>
- [4] TYPO3-5.0 Mailing list: <http://tinyurl.com/t3v5list>
- [5] MVC-Dokumentation: <http://flow3.typo3.org/documentation/reference/mvc-framework/>

THE AUTHOR



Robert Lemke is a co-founder of the TYPO3 Association and the spiritual founder of the FLOW3 framework. He is heading the development of TYPO3 5.0 and is especially fond of clean source-code. Robert lives in Lübeck with his wife Heike and their espresso machine Vibiemme.