Plug-Ins

About F3 Plug-Ins

Plug-ins are nothing more than autoloaded classes that use framework built-ins to extend F3's features and functionality. If you'd like to contribute, leave a note at the Fat-Free Discussion Area (https://groups.google.com/forum/#!forum/f3-framework) hosted by Google Groups or tell us about it in the FreeNode `#fatfree` IRC channel. Someone else might be involved in a similar project. The framework community will appreciate it a lot if we unify our efforts.

Image

CAPTCHA Images

There might be instances when you want to make your forms more secure against spam bots and malicious automated scripts. F3 provides a `captcha()` method to generate images with random text that are designed to be recognizable only by humans.

```
$img = new Image();
$img->captcha('fonts/CoolFont.ttf',16,5,'SESSION.captcha_code');
$img->render();
```

This example generates a random image based on your desired TrueType font. The `fonts/` folder is a subfolder within application's `UI` path. The second parameter indicates the font size (a 2x magnification process is performed, i.e. a size of 16 will produce an image of around 32 pixels height). The third parameter defines the quantity of hexadecimal characters to generate, valid values between minimum of 4 and maximum of 13.

The last parameter represents a F3 variable name. Use it to store the string equivalent of the CAPTCHA image, in order to compare it with the user input. To make the string reload-safe, we specified a session variable: `SESSION.captcha_code` which maps to `$_SESSION['captcha_code']`, which you can use later to verify whether the input element in the form submitted matches this string.

Image Processing

The image plugin also provides additional processing features for scale, crop and overlay images, or adjusting brightness, contrast and many more. Please have a look to the Image Class API reference (/image) for additional features description.

Log

See how easy it is to create a custom Logger to save all your interesting application events.

```
$logger = new \Log('app-events.log');
$logger->write('User John logged in.');
```

Markdown

Convert your favorite Markdown (Wikipedia) (https://en.wikipedia.org/wiki/Markdown) text to HTML.

```
$filePath = 'content/readme.md';
$fileContent = $f3->read($filePath); // read file contents

echo \Markdown::instance()->convert($fileContent);
```

Web

Grabbing Data from Another Site

We've covered almost every feature available in the framework to run a stand-alone Web server. For most applications, these features will serve you quite well. But what do you do if your application needs data from another Web server on the network? F3 has the Web plugin to help you in this situation:

```
$web=new Web;
$request=$web->request('http://www.google.com/');
// another way to do it:
$request=Web::instance()->request('http://www.google.com/');
```

This simple example sends an HTTP request to the page located at www.google.com and stores it in the `$request` PHP variable. The `request()` method returns an array containing the HTTP response such that `$request['headers']` and `$request['body']` represent the response headers and body, respectively. We could have saved the contents using the F3::set command, or echo'ed the output directly to the browser. Retrieving another HTML page from the net may not have any practical purpose. But it can be particularly useful in ReSTful applications, like querying a CouchDB server.

```
$host='localhost:5984';
$web->request($host.'/_all_dbs'),
$web->request($host.'/testdb/',array('method'=>'PUT'));
```

You may have noticed that you can pass an array of additional options to the `request()` method:

```
$web->request(
    'https://www.example.com:443?'.
    http_build_query(
        array(
            'key1'=>'value1',
            'key2'=>'value2'
        )
    ),
    array(
        'header'=>array(
            'Accept: text/html,application/xhtml+xml,application/xml',
            'Accept-Language: en-us'
        ),
        'follow_location'=>FALSE,
        'max_redirects'=>30,
        'ignore_errors'=>TRUE
    )
);
```

If the framework variable `CACHE` is enabled, and if the remote server instructs your application to cache the response to the HTTP request, F3 will comply with the request and retrieve the cached response each time the framework receives a similar request from your application, thus behaving like a browser.

Fat-Free will use whatever means are available on your Web server for the `request()` method to run: PHP stream wrappers (`allow_url_fopen`), cURL module, or low-level sockets.

Handling File Downloads

F3 has a utility for sending files to an HTTP client, i.e. fulfilling download requests. You can use it to hide the real path to your download files. This adds some layer of security because users won't be able to download files if they don't know the file names and their locations. Here's how it's done:

```
$f3->route('GET /downloads/@filename',
    function($f3,$args) {
        // send() method returns FALSE if file doesn't exist
        if (!Web::instance()->send('/real/path/'.$args['filename']))
            // Generate an HTTP 404
        $f3->error(404);
    }
);
```

Remoting and Distributed Applications

The `request()` method can also be used in complex SOAP or XML-RPC applications, if you find the need for another Web server to process data on your computer's behalf - thus harnessing the power of distributing computing. W3Schools.com has an excellent tutorial on SOAP. On the other hand, TutorialsPoint.com gives a nice overview of XML-RPC.

more Plug-Ins

There are many more Plug-Ins available for F3. Have a look into their API Reference to get more information about them.

- Audit (audit)
- Auth (auth)
- Basket (basket)
- Image (image)
- Log (log)
- Markdown (markdown)
- SMTP (smtp)
- Template (template)
- Test (test)
- Web (web)
- Geo (geo)
- OpenID (openid)
- PingBack (pingback)
- Google Static Maps API v2 (google-static-maps)

Still haven't found what you're looking for? Maybe you'll find it in the User Contributed Plug-Ins (development#user-plugins) section.

← 5. Databases (databases) 7. Optimization → (optimization)