SensioLabs

Symfony

Hacking & Extending Symfony2
SF2C3

The Profiler

The Profiler

Symfony comes with the Profile object that collects data at every HTTP request.

It uses a set of DataCollector objects to gather information and exposes a web interface to display them nicely.

Data Collector	Meaning
`time`	Collects data about timers
`security`	Collects data about the security context
`request`	Collects data about the current HTTP request
`memory`	Collects data about the memory consumption
`logger`	Collects the recorded logs in the logger
`exception`	Collects data when uncaught exceptions are raised
`event`	Collects data on triggered events
`config`	Collects data on the execution platform configuration

Additional data collectors

Data Collector	Meaning
doctrine	Collects executed SQL queries with Doctrine ORM
propel	Collects executed SQL queries with Propel ORM
mongodb	Collects executed Mongo queries with Doctrine ODM
message	Collects sent emails with Swiftmailer

Creating a Data Collector

The Data Collector

A data collector is an object that collects data and must implement the DataCollectorInterface interface.

The DataCollectorInterface interface

```php
namespace Symfony\Component\HttpKernel\DataCollector;

use Symfony\Component\HttpFoundation\Request;
use Symfony\Component\HttpFoundation\Response;

interface DataCollectorInterface
{
    public function collect(
        Request $request,
        Response $response,
        \Exception $exception = null
    );

    public function getName();
}
```

The abstract DataCollector class

```php
namespace Symfony\Component\HttpKernel\DataCollector;

abstract class DataCollector implements DataCollectorInterface, \Serializable
{
    protected $data;

    public function serialize()
    {
        return serialize($this->data);
    }

    public function unserialize($data)
    {
        $this->data = unserialize($data);
    }
}
```

Creating a data collector

```php
namespace Acme\Bundle\ComposerBundle\DataCollector;

use Symfony\Component\HttpKernel\DataCollector\DataCollector;
use Symfony\Component\HttpFoundation\Request;
use Symfony\Component\HttpFoundation\Response;
use Acme\Bundle\ComposerBundle\Composer;

class ComposerDataCollector extends DataCollector
{
    private $composer;

    public function __construct(Composer $composer)
    {
        $this->composer = $composer;
    }

    public function getName()
    {
        return 'composer';
    }
}
```

Creating a data collector

```
class ComposerDataCollector extends DataCollector
{
    public function collect(
        Request $request,
        Response $response,
        \Exception $exception = null
    )
    {
        $this->data['packages'] = $this->composer->getPackages();
    }

    public function getPackages()
    {
        return $this->data['packages'];
    }
}
```

The Composer service class

```
namespace Acme\ComposerBundle;

class Composer
{
    private $file;

    public function __construct($composerLockFile)
    {
        $this->file = realpath($composerLockFile);
    }

    public function getPackages()
    {
         // ...
    }
}
```

```php
class Composer
{
    public function getPackages()
    {
        // read composer.lock file
        $config = json_decode(file_get_contents($this->file));

        $packages = array();
        foreach ($config->packages as $package) {
            $packages[] = array(
                'package' => $package->name,
                'license' => $package->license,
                'version' => $package->version,
            );
        }

        return $packages;
    }
}
```

Enabling the data collector

```
<?xml version="1.0" ?>
<container>
    <services>
        <service id="acme_composer.data_collector.composer"
            class="Acme\[...]\DataCollector\ComposerDataCollector"
            public="false">
            <argument type="service" id="acme_composer.composer"/>
            <tag name="data_collector"/>
        </service>
        <service id="acme_composer.composer"
            class="Acme\ComposerBundle\Composer"
            public="false">
            <argument>%kernel.root_dir%/../composer.lock</argument>
        </service>
    </services>
</container>
```

Inspecting the profiler in functional tests

```php
class ComposerControllerTest extends WebTestCase
{
    public function testIndex()
    {
        $client = static::createClient();
        $client->request('GET', '/configuration');

        $profile   = $client->getProfile();
        $collector = $profile->getCollector('composer');

        $this->assertLessThan(30, count($collector->getPackages()));
    }
}
```

The Web Profiler Application

The Web Profiler application

The profiler application enabled in development environment is basically just a nice web interface for the Profile object.

The Profiler application

When registering a new data collector, it's also possible to attach a template to render to extend the Profiler and WDT panels.

The profiler template format

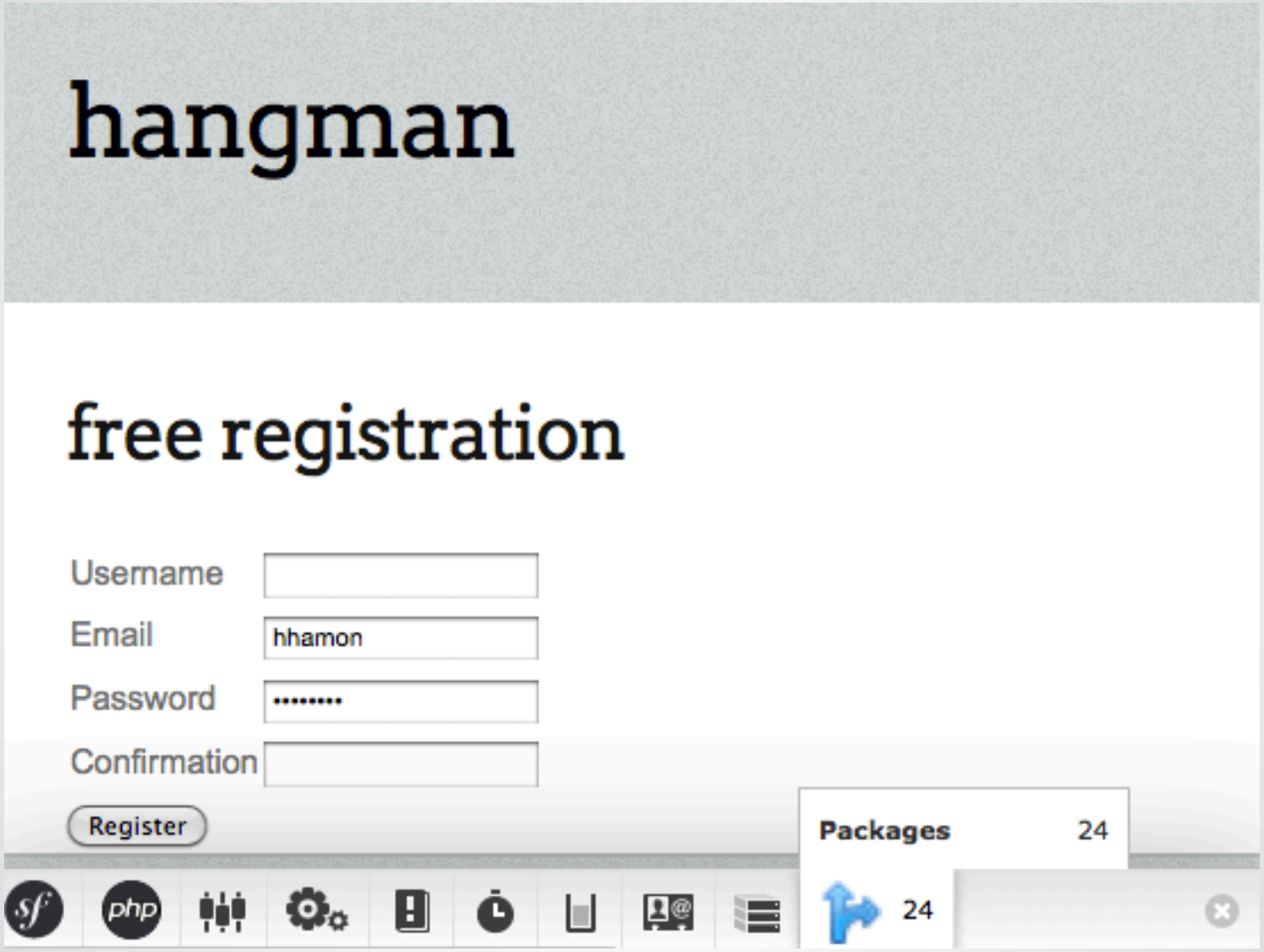
```
{% extends 'WebProfilerBundle:Profiler:layout.html.twig' %}

{% block toolbar %}
    {# the web debug toolbar content #}
{% endblock %}

{% block head %}
    {# if the web profiler panel needs some specific JS or CSS files #}
{% endblock %}

{% block menu %}
    {# the menu content #}
{% endblock %}

{% block panel %}
    {# the panel content #}
{% endblock %}
```

Linking a data collector to a template

```
<service id="acme_composer.data_collector.composer"
    class="Acme\[...]\DataCollector\ComposerDataCollector"
    public="false">

    <argument type="service" id="acme_composer.composer"/>

    <tag name="data_collector"
       template="AcmeComposerBundle:Collector:composer"
       id="composer"
       priority="1000"
    />

</service>
```

The Web Debug Toolbar

```
{% block toolbar %}

    {% set icon %}
        <span>
            <img width="13" height="28" alt="Composer"
src="...[]..."/>
            <span>{{ collector.packages|length }}</span>
        </span>
    {% endset %}
    {% set text %}
        <div class="sf-toolbar-info-piece">
            <b>Packages</b>
            <span>{{ collector.packages|length }}</span>
        </div>
    {% endset %}
    {% include '@WebProfiler/Profiler/toolbar_item.html.twig'
       with { 'link': profiler_url } %}

{% endblock %}
```


hangman
free registration
Username
Email
hhamon
Password
Confirmation
Register
Packages
24
24

The Composer Panel

```
{% block menu %}
<span class="label">
    <span class="icon"><img src="
FyZQBBZG9iZSBJbWFnZVJlYWR5ccllPAAAAjBJREFUeNqEU89rE0EUfjM7+8NNKGINhCRQhF6
8CetFNFAEwZM/
wGPOigchf0ovAb3nICrosVpaBLGXutJDQTE0WqQxFpt2001kd2dmfbPJrgmx9sGwu9+879v3v
nlD4jiGK41voOk6MMaAUHoTMGIpVzjnIKIIVI4UInmqiMIQoiAAjnsMpqOGxPr4/
TyuJpwSkwI1/GP93vWSoz5erHdSof
+K0JSMpdZvV4tOr8dBrVvXio7C1N6pAoLzqurn2eu2a1kxmGYMz9+0XYUle9izGHnwBFdtpgU
EH6gENOpDEEiFJEZhXJZSJgZSSoFqmmrPgZGZzUxAOTomgCKofYWhoffxVMoIXxRSlrEauHPj
gvNq9Ws9FUkEhr6flfQ7FHAchIlgPmc8KlfOFixbnyOUmMihm6OeLFUXnNW1lvLnV1ZBGu1BC
FzGUKrMzbPCucr3AWd2RME2sAUC4DECXkwzzozAITVAqMRiaWEQxaSohUfeTme/c
+gPecjFpbtXnbcv37uEkGWNsZUZAU8z4YsnwWY6KXTbe63tnQ1MbgAhW
+jJ2ubT9YTMDGPCxLGBKnxiQKBjC/3uOY+PnzYOTXuI7h8gWR2piynvmK43p46RTwgEaK6tE
+hvf97HQWrgOrByOUiP+5+jPFlBiALIB3/
v51AEwVaCYYtn8vmT74K6cRmAZKIB9I99gTPgpbjySTfNEwRwOsbhdpYfj94IceEvDuXFRbBs
G7q7u1MCfwQYALhRPOgzaWSbAAAAAElFTkSuQmCC" alt="Composer" /></span>
    <strong>Composer</strong>
</span>
{% endblock %}
```

```
{% block panel %}
    <h2>Installed Composer packages</h2>
    <table>
        <thead>
            <tr>
                <th scope="col">Package</th>
                <th scope="col">License</th>
                <th scope="col">Version</th>
            </tr>
        </thead>
        <tbody>
            {% for package in collector.packages %}
            <tr>
                <td>{{ package.package }}</td>
                <td>{{ package.license|join(', ') }}</td>
                <td>{{ package.version }}</td>
            </tr>
            {% endfor %}
        </tbody>
    </table>
{% endblock %}
```

Installed packages

Package name	License	Version
doctrine/annotations	MIT	v1.1.2
doctrine/cache	MIT	v1.3.0
doctrine/collections	MIT	v1.1
doctrine/common	MIT	v2.4.1
doctrine/dbal	MIT	v2.4.1
doctrine/doctrine-bundle	MIT	v1.2.0
doctrine/inflector	MIT	v1.0
doctrine/lexer	MIT	v1.0
doctrine/orm	MIT	v2.4.1
incenteev/composer-parameter-handler	MIT	v2.1.0
jdorn/sql-formatter	MIT	v1.2.16
kriswallsmith/assetic	MIT	v1.1.2

Training Department

SensioLabs Training

92-98 Boulevard Victor Hugo

92 115 Clichy Cedex

FRANCE

Phone: +33 140 998 211

Email: training@sensiolabs.com

symfony.com - trainings.sensiolabs.com