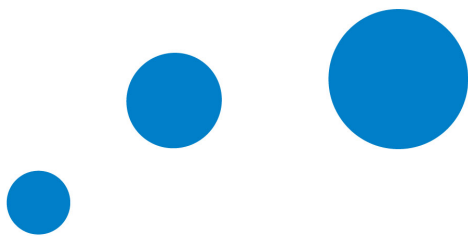


**FWP Shop**



**FWP** systems

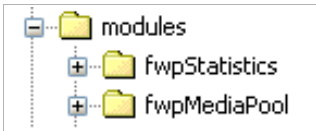
**Modul-Implementierung**

**Shop-Version 4.0**

## 1. Einführung

Module werden im FWP Shop im Verzeichnis modules/ angelegt. Im Verzeichnis modules/ wird ein neues Verzeichnis, entsprechend des Modulnamens angelegt. Der Name des Moduls muss camelCased sein, damit eine einheitliche Struktur gegeben ist.

Beispiel:

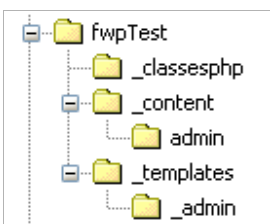


## 2. Verzeichnisstruktur

Jedes Modul besitzt 4 Hauptverzeichnisse, die angelegt werden müssen.

Verzeichnis	Verwendungszweck
_content/	In diesem Verzeichnis werden PHP-Dateien gespeichert, in denen die Logik für das Modul ausgeführt wird.
_classesphp/	In diesem Verzeichnis sind Klassen hinterlegt, die in den PHP-Dateien in _content genutzt werden können. Diese Unterteilung dient einer klaren Trennung der Komponenten.
_templates/	In diesem Verzeichnis werden alle Grafiken, CSS-Dateien usw. abgelegt, die für das Modul benötigt werden.
_templates/_admin/	In diesem Verzeichnis werden alle Templates abgelegt, die für die Darstellung des Moduls benötigt werden.

Im weiteren Verlauf dieser Dokumentation werden die einzelnen Schritte für ein besseres Verständnis anhand eines konkreten Beispiels (fwpTest) dargestellt. Nachdem Sie die beschriebene Verzeichnisstruktur angelegt haben, sollte diese so aussehen:



### 3. Entwicklung eines Moduls

Module sind anhand von \$\_GET-Parametern aufgebaut. Diese \$\_GET-Parameter (sPage, sAction) geben die Verzeichnisstruktur vor. Wie in der oben angezeigten Grafik zu sehen ist, wurde ein Verzeichnis namens „admin“ im Verzeichnis „\_content“ angelegt. Diese Verzeichnisebene stellt den \$\_GET-Parameter „sPage“ dar.

Wenn Sie im Verzeichnis „admin“ nun beispielsweise die Datei „test.inc.php“ anlegen, müssten Sie folgende URL aufrufen um die Datei „test.inc.php“ ausführen zu lassen:

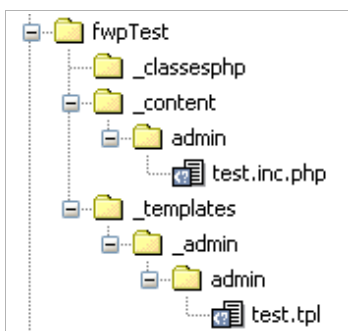
```
index.php?sMod=fwpTest&sPage=admin&sAction=test
```

Der Dateiname stellt den \$\_GET-Parameter „sAction“ dar. Die Verzeichnisse und Dateien können beliebig benannt werden, müssen aber den aufgerufenen \$\_GET-Parametern entsprechen.

### 4. Templates

Damit Sie Daten darstellen können, müssen Sie Templates anlegen. Wie bereits beschrieben liegen diese Templates im Verzeichnis „\_templates/\_admin/“. Die Verzeichnisstruktur ist hier gleichzusetzen mit dem der PHP-Logik. Legen Sie das Verzeichnis „\_templates/\_admin/admin/“ und die Datei „test.tpl“ im selben Verzeichnis an. Der Dateiname entspricht dem \$\_GET-Parameter „sAction“ und der Verzeichnisname wieder dem \$\_GET-Parameter „sPage“. Nach Anlage des Verzeichnisses und der Template-Datei können Sie den entsprechenden HTML-Code in die Template-Datei schreiben, um Ihr Modul darzustellen.

Die jetzige Verzeichnisstruktur sollte folgendermaßen aussehen:



## 5. Nutzung von Modul-Klassen

Um Klassen in den PHP-Dateien (\_content) zu nutzen, müssen diese geladen werden. Wenn Sie beispielsweise die Datei „test\_std.inc.php“ im Verzeichnis „\_classesphp“ anlegen und diese laden wollen, können Sie dies über folgenden Befehl innerhalb einer PHP-Datei machen:

```
load::classes('test', 'fwpTest');
```

Der erste Parameter ist der Klassenname, wobei das Suffix „\_std.inc.php“ nicht mit vorhanden sein darf. Der zweite Parameter ist der Modul-Name in dem Sie sich aktuell befinden. Nachdem die Klasse erfolgreich geladen wurde, kann diese verwendet werden.

## 6. Menüpunkt im Administrationsbereich anlegen

Damit Ihr Modul im Menü des Administrationsbereiches erscheint, müssen Sie im Verzeichnis „\_classesphp“ die Datei „adminInterface\_std.inc.php“ anlegen. Der Inhalt der Datei sollte anhand des „fwpTest“-Beispiels folgendermaßen aussehen:

```
<?php

load::interfaces('adminInterface_iface');

class fwpTestAdminInterface implements adminInterface_iface {
    public function modifyMenu() {
        $oMenu = backendMenu::getInstance();
        $oMenu->appendItem(new backendMenuItem(
            'fwpTest', // Modulname
            'Test-Modul', // Beschriftung für den Link im Menü
            'index.php?sMod=fwpTest&sPage=admin&sAction=test' // URL
        ),
            'fwpAdmin' // Knoten in den der neue Menüpunkt eingetragen wird
        );
    }

    public function autoload() {}
}

?>
```

Diese Klasse wird automatisch geladen und ausgeführt. Der Inhalt der Datei sorgt dafür, dass ein neuer Menüpunkt im Administrationsbereich angezeigt wird (Siehe Kommentare).

Der zweite Parameter der Methode „appendItem“, in diesem Fall der String „fwpAdmin“, gibt den Knoten in der Menüstruktur an, in welchem der neue Menüpunkt erscheinen soll. Den passenden Knoten für Ihre Bedürfnisse entnehmen Sie bitte der Datei „\_config/\_xml/\_admin/menu.xml“ Ihrer Shop-Installation. Die korrekten Namen können Sie den XML-Knoten „sInternalName“ entnehmen.

## **7. Sicherheit**

Nicht jedes Modul darf von jedem Shop-Mitarbeiter aufgerufen bzw. genutzt werden. Die Zugriffsrechte auf Ihr neu entwickeltes Modul können Sie über die Rechteverwaltung konfigurieren. Für genauere Informationen lesen Sie bitte die Dokumentation zur Rechteverwaltung, die Sie ebenfalls auf unserer Website finden.