Drupal – eigenes Modul erstellen

Jedes Drupal-Modul benötigt mindestens zwei Dateien in seinem Hauptverzeichnis:

`Die .info.yml-Datei:` Diese Datei enthält Metadaten über dein Modul (Name, Beschreibung, Abhängigkeiten usw.) und ist essenziell, damit Drupal dein Modul überhaupt erkennt.

`Die .module-Datei:` Diese Datei enthält den PHP-Code deines Moduls, in dem die Logik und die Funktionalität implementiert werden.

Schritt 1: Das Modulverzeichnis erstellen

Zuerst erstellen wir das Verzeichnis für unser Modul. Drupal-Module werden normalerweise im `modules/custom`-Verzeichnis deiner Drupal-Installation abgelegt.

Navigiere in deinem Drupal-Installationsverzeichnis zum Pfad web/modules.

Erstelle darin einen neuen Ordner namens custom.

Innerhalb des Ordners custom erstellst du einen weiteren Ordner für unser Modul. Nennen wir ihn `games_module`.

Dein Dateipfad sollte dann so aussehen:
deine_drupal_installation/web/modules/custom/games_module

Schritt 2: Die .info.yml-Datei erstellen

Navigiere in den gerade erstellten Ordner games_module.

Erstelle eine neue Datei mit dem Namen `games_module.info.yml.` Achte genau auf die Endung .info.yml!

Öffne diese Datei mit einem Texteditor (wie VS Code, Sublime Text, Notepad++ oder Atom) und füge den folgenden Inhalt ein:

```
name: 'Games Module'
type: module
description: 'Enables browser-based games and stores scores.'
core_version_requirement: '^10 || ^11'
package: 'Custom'
```

Lass uns kurz erklären, was die einzelnen Zeilen bedeuten:

- name: Das ist der Anzeigename deines Moduls, der in der Modul-Liste im Drupal-Backend erscheint.
- type: Definiert, dass es sich um ein module handelt.
- description: Eine kurze Beschreibung dessen, was dein Modul tut.
- core_version_requirement: Gibt an, mit welchen Drupal-Hauptversionen dein Modul kompatibel ist. ^10 || ^11 bedeutet, es ist kompatibel mit Drupal 10 und 11 (alle Unterversionen). Da du Drupal 11.1 nutzt, passt das perfekt.
- package: Dies ist eine optionale Gruppierungskategorie. "Custom" ist eine gute Wahl für selbst entwickelte Module, damit sie im Backend leichter zu finden sind.

Schritt 3: Die .module-Datei erstellen

Diese Datei wird den eigentlichen PHP-Code unseres Moduls enthalten.

Erstelle im selben Ordner games_module eine neue Datei mit dem Namen games_module.module. Auch hier ist die Endung wichtig!

Öffne diese Datei und füge zunächst nur die öffnende PHP-Markierung hinzu:

```
<?php


/**
 * @file
 * Contains games_module.module.
 */
```

<?php: Dies signalisiert dem Server, dass der folgende Code PHP ist.

/** ... */: Das ist ein DocBlock-Kommentar, der eine kurze Beschreibung der Datei enthält. Es ist gute Praxis, Code zu dokumentieren.

Schritt 4: Das Modul in Drupal aktivieren

Nachdem wir die Dateien erstellt haben, müssen wir Drupal mitteilen, dass es unser Modul gibt und es aktivieren.

1. Melde dich als Administrator in deiner Drupal-Installation an.

2. Navigiere zu **Erweitern** (oder **Extend** im Englischen) im Administrationsmenü. Das ist der Bereich, in dem du alle Module verwalten kannst.
3. Scrolle durch die Liste der Module oder nutze die Suchfunktion. Du solltest jetzt unter der Kategorie "Custom" (oder welche du bei `package` angegeben hast) dein Modul "Games Module" sehen.
4. Aktiviere das Kontrollkästchen neben "Games Module" und klicke dann am Ende der Seite auf **Installieren** (oder **Install**).

Wenn alles richtig gemacht wurde, siehst du eine Erfolgsmeldung, und das Modul ist jetzt aktiviert! Herzlichen Glückwunsch, du hast dein erstes Drupal-Modul erstellt und aktiviert!

Schritt 5: Eine einfache Seite für unser Spiel erstellen

Damit wir unser Spiel im Browser anzeigen können, brauchen wir eine Seite in Drupal, die es hostet. Dazu verwenden wir das Konzept eines Controllers und eines Routing-Eintrags.

Ein Routing-Eintrag definiert eine URL (z.B. /spiele/mein-erstes-spiel) und verknüpft sie mit einer Funktion (einer Methode in einem Controller), die den Inhalt für diese URL generiert.

Ein Controller ist eine PHP-Klasse, die die Logik zum Erzeugen von Inhalten für eine bestimmte Route enthält.

Schritt 5.1: Den Routing-Eintrag definieren

Zuerst müssen wir Drupal mitteilen, welche URL unser Spiel haben soll. Dafür erstellen wir eine neue Datei namens `games_module.routing.yml` im Hauptverzeichnis deines Moduls (web/modules/custom/games_module/).

Füge den folgenden Inhalt in die Datei ein:

```
games_module.game_page:
  path: '/spiele/mein-erstes-spiel'
  defaults:
    _controller: '\Drupal\games_module\Controller\GameController::gamePage'
    _title: 'Mein erstes Browser-Spiel'
  requirements:
    _permission: 'access content'
```

Schauen wir uns an, was hier passiert:

- games_module.game_page: Dies ist der eindeutige interne Name unserer Route. Es ist gute Praxis, den Modulnamen als Präfix zu verwenden.
- path: '/spiele/mein-erstes-spiel': Das ist die URL, unter der unser Spiel erreichbar sein wird (z.B. deine-drupal-seite.de/spiele/mein-erstes-spiel).
- defaults:: Hier definieren wir, was passieren soll, wenn die URL aufgerufen wird.
    - _controller: '\Drupal\games_module\Controller\GameController::gamePage': Das ist der wichtigste Teil. Er sagt Drupal: "Wenn diese URL aufgerufen wird, rufe die Methode gamePage in der Klasse GameController auf, die sich im Namespace Drupal\games_module\Controller befindet."
    - _title: 'Mein erstes Browser-Spiel': Dies ist der Titel der Seite, der im Browser-Tab und in der Seite selbst angezeigt wird.
- requirements:: Hier legen wir fest, welche Berechtigungen ein Benutzer benötigt, um diese Seite aufrufen zu können.
    - _permission: 'access content': Dies ist eine Standard-Drupal-Berechtigung, die besagt, dass jeder Benutzer, der Inhalte sehen darf (was fast jeder angemeldete und anonyme Benutzer ist), diese Seite aufrufen kann. Für ein öffentliches Spiel ist das passend.

Schritt 5.2: Den Controller erstellen

Jetzt müssen wir die Controller-Klasse erstellen, die die Methode gamePage enthält.

Erstelle im Verzeichnis deines Moduls (web/modules/custom/games_module/) einen neuen Ordner namens `src`.

Innerhalb von src erstellst du einen weiteren Ordner namens `Controller`.

Im Ordner Controller erstellst du eine neue Datei namens `GameController.php`.

Dein Dateipfad sollte dann so aussehen: deine_drupal_installation/web/modules/custom/games_module/src/Controller/GameController.php

Füge den folgenden Inhalt in die Datei GameController.php ein:

```
<?php

namespace Drupal\games_module\Controller;

use Drupal\Core\Controller\ControllerBase;

/**

 * Provides a controller for the game page.
```

```
 */
class GameController extends ControllerBase {
  /**
   * Returns a render array for the game page.
   *
   * @return array
   * A simple render array.
   */
  public function gamePage() {
    return [
      '#markup' => '<h2>Willkommen zu unserem ersten Spiel!</h2><p>Hier wird bald unser Spiel erscheinen.</p>',
    ];
  }
}
```

Erklärung des Codes:

- `namespace Drupal\games_module\Controller;`: Dies definiert den **Namespace** der Klasse. Er muss exakt zu dem passen, was wir im `routing.yml` angegeben haben, damit Drupal die Klasse finden kann.
- `use Drupal\Core\Controller\ControllerBase;`: Wir importieren die Basisklasse `ControllerBase`. Es ist gute Praxis, von dieser Klasse zu erben, da sie nützliche Funktionen für Controller bereitstellt.
- `class GameController extends ControllerBase { ... }`: Dies ist unsere Controller-Klasse.
- `public function gamePage() { ... }`: Das ist die Methode, die wir in unserem `routing.yml`-Eintrag (`_controller`) angegeben haben. Wenn die URL aufgerufen wird, wird diese Methode ausgeführt.
- `return [ '#markup' => '...', ];`: Controller-Methoden in Drupal geben normalerweise ein sogenanntes **Render-Array** zurück. Das ist eine PHP-Array-Struktur, die Drupal Anweisungen gibt, wie Inhalte gerendert (angezeigt) werden sollen.
    - `'#markup'`: Dies ist ein häufig verwendeter Schlüssel in Render-Arrays, um einfaches HTML oder Text auszugeben.

Schritt 5.3: Drupal den Cache leeren

Nachdem wir neue .yml-Dateien (wie routing.yml) und neue Klassen hinzugefügt haben, muss Drupal seinen internen Cache leeren, damit es die neuen Definitionen erkennt.

- Melde dich als Administrator in deiner Drupal-Installation an.
- Navigiere zu Konfiguration (oder Configuration).
- Wähle unter dem Abschnitt "Entwicklung" Leistung (oder Performance).
- Klicke auf die Schaltfläche Alle Caches leeren (oder Clear all caches).

Schritt 5.4: Testen der neuen Seite

Nachdem der Cache geleert wurde, kannst du die neue Seite in deinem Browser aufrufen.

Öffne deinen Webbrowser.

Gib die URL zu deiner Drupal-Seite ein, gefolgt von dem Pfad, den wir definiert haben: http://deine-drupal-seite.de/spiele/mein-erstes-spiel (bei mir lokal: localhost:8081/spiele/mein-erstes-spiel)

Du solltest jetzt eine Seite mit der Überschrift "Mein erstes Browser-Spiel" und dem Text "Willkommen zu unserem ersten Spiel! Hier wird bald unser Spiel erscheinen." sehen.