Php - Beispielhafte Anfrage

Beschreibung:

Dieses PHP-Skript ruft Sensordaten von der SmartDog API v2 ab.

Die API liefert Sensordaten für einen bestimmten Zeitraum im JSON-Format zurück.

Programmiersprache: PHP

API-URL: `https://apiv2.smart-dog.eu/index.php`

Code:

```
<?php

function getSensorData() {
    $url = "https://apiv2.smart-dog.eu/index.php";

    $requestData = [
        "action" => "getSensorData",
        "apikey" => "6641d282073d76b625987af5141d3e2a",
        "SensorID" => "551941",
        "UTC_TIMESTAMP_FROM" => "1739867939",
        "UTC_TIMESTAMP_TO" => "1739877939"
    ];

    $options = [
        "http" => [
            "header"  => "Content-Type: application/json\r\n",
            "method"  => "POST",
            "content" => json_encode($requestData)
        ]
    ];

    $context  = stream_context_create($options);
    $response = file_get_contents($url, false, $context);
```

```
    if ($response === FALSE) {
        die("Fehler beim Abrufen der Sensordaten");
    }

    $data = json_decode($response, true);
    echo "<pre>" . print_r($data, true) . "</pre>";
}

getSensorData();

?>
```

Beschreibung der Funktion:

1. API-Request vorbereiten:
    - Erstellt eine Anfrage an die API mit den Parametern:
        - `action`: `"getSensorData"`
        - `apikey`: `"6641d282073d76b625987af5141d3e2a"`
        - `SensorID`: `"551941"`
        - `UTC_TIMESTAMP_FROM`: `"1739867939"`
        - `UTC_TIMESTAMP_TO`: `"1739877939"`
2. Datenabruf mit `file_get_contents()`:
    - Sendet eine `POST`-Anfrage mit JSON-Daten an die API.
    - Nutzt `stream_context_create()` zum Setzen der `Content-Type`-Header.
    - Falls die Antwort fehlschlägt (`FALSE`), gibt das Skript eine Fehlermeldung aus.
3. Antwortverarbeitung:
    - Dekodiert die JSON-Antwort mit `json_decode()`.
    - Gibt die Sensordaten formatiert (`print_r()`) als HTML-`<pre>`-Block aus.

Beispiel für eine erfolgreiche API-Antwort:


```
{
   "sensor_id": 551941,
   "valid": 1,
   "datasets": {
      1739868002.000000: {
         "DATA": "251",
         "TIMESTAMP_UTC": 1739868002,
         "TIMESTAMP_LOCAL": 1739871602
      },
      1739868302.000000: {
         "DATA": "257",
         "TIMESTAMP_UTC": "1739868302.000000",
         "TIMESTAMP_LOCAL": 1739871902
      }
   }
}
```


Alternative mit cURL (empfohlen für größere API-Anfragen):

```
<?php

function getSensorData() {
   $url = "https://apiv2.smart-dog.eu/index.php";

   $requestData = [
      "action" => "getSensorData",
      "apikey" => "6641d282073d76b625987af5141d3e2a",
      "SensorID" => "551941",
      "UTC_TIMESTAMP_FROM" => "1739867939",
```

```
        "UTC_TIMESTAMP_TO" => "1739877939"
    ];

    $ch = curl_init($url);
    curl_setopt($ch, CURLOPT_RETURNTRANSFER, true);
    curl_setopt($ch, CURLOPT_HTTPHEADER, ["Content-Type: application/json"]);
    curl_setopt($ch, CURLOPT_POST, true);
    curl_setopt($ch, CURLOPT_POSTFIELDS, json_encode($requestData));

    $response = curl_exec($ch);

    if (curl_errno($ch)) {
        die("Fehler beim Abrufen der Sensordaten: " . curl_error($ch));
    }

    curl_close($ch);

    $data = json_decode($response, true);
    echo "<pre>" . print_r($data, true) . "</pre>";
}

getSensorData();

?>
```

Diese Variante verwendet `cURL`, was für größere API-Anfragen besser geeignet ist.

Einsatz in einer HTML-Seite:

```
<button onclick="location.reload();">Sensordaten abrufen</button>
<pre><?php getSensorData(); ?></pre>
```

Beim Klicken auf den Button `"Sensordaten abrufen"` wird die PHP-Funktion ausgeführt, und die Daten erscheinen im `<pre>`-Tag.

```

```