

Httpful

Httpful is a simple, chainable, readable PHP library intended to make speaking HTTP sane.

Or:

A Chainable, REST Friendly, PHP HTTP Client. A sane alternative to cURL.

Info: <http://phphttpclient.com/>

<https://www.bountysource.com/teams/httpful/issues>

Send off a GET request. Get automatically parsed JSON response. The library notices the JSON Content-Type in the response and automatically parses the response into a native PHP object.

Calls send JSON parameters and return JSON results.

Test-Umgebung:

<https://test.projects.hakmistelbach.ac.at/api/books>

<https://test.projects.hakmistelbach.ac.at/api/users>

<https://test.projects.hakmistelbach.ac.at/api/lendings>

Info:

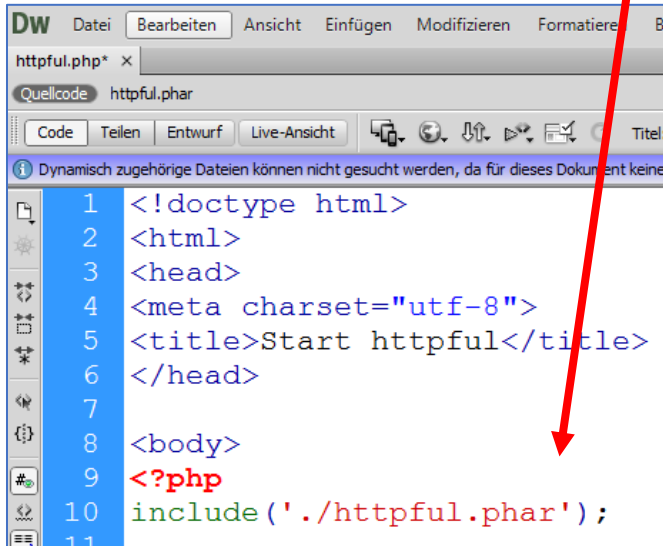
<https://hotexamples.com/examples/httpful/Request/put/php-request-put-method-examples.html>

<http://phphttpclient.com/docs/class-Httpful.Request.html>

Arbeite in XAMPP



Startet immer mit dem Einbinden der Bibliothek in „httpful.phar“

Httpful provides a PHP Archive file that includes the entire library. Simply download this file and include it in your code – name: „httpful.phar“



```
1 <!doctype html>
2 <html>
3 <head>
4 <meta charset="utf-8">
5 <title>Start httpful</title>
6 </head>
7
8 <body>
9 <?php
10 include('./httpful.phar');
11
```

Xampp-Ordner: httpful

 httpful.phar	14.01.2017 10:35	PHAR-Datei	63 KB
 httpful.php	18.01.2017 15:12	PHP-Datei	3 KB

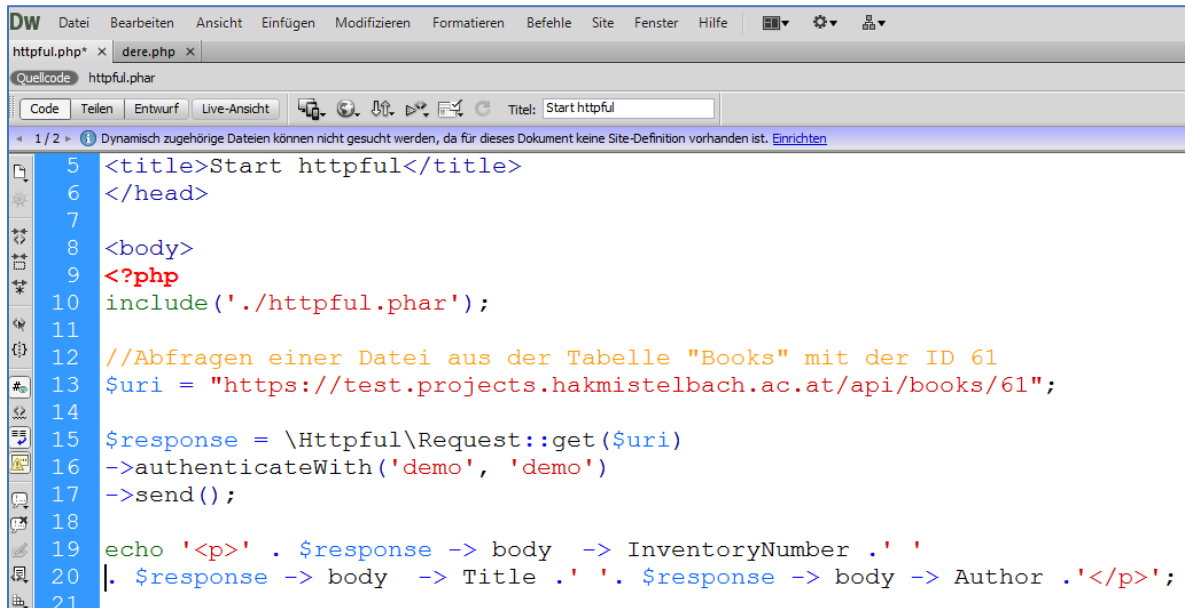
Info allgemein:

HTML-Befehle, wie z.B. <p> immer in Anführungsstrichen, dann den Code anhängen mit dem Verkettungspunkt: echo '' . \$val1 -> KOID . '';

Authentifizierung:

```
->authenticateWith('username', 'password')
```

1) Erste Abfragen für genau einen Datensatz, der mit der ID-Nummer ausgewählt wird:



The screenshot shows a code editor window with the following content:

```
5 <title>Start httpful</title>
6 </head>
7
8 <body>
9 <?php
10 include('./httpful.phar');
11
12 //Abfragen einer Datei aus der Tabelle "Books" mit der ID 61
13 $uri = "https://test.projects.hakmistelbach.ac.at/api/books/61";
14
15 $response = \Httpful\Request::get($uri)
16 ->authenticateWith('demo', 'demo')
17 ->send();
18
19 echo '<p>' . $response -> body -> InventoryNumber .' '
20 |. $response -> body -> Title .' '. $response -> body -> Author .'</p>';
21
```

Code:

```
<!doctype html>
```

```
<html>
```

```
<head>
```

```
<meta charset="utf-8">
```

```
<title>Start httpful</title>
```

```
</head>
```

```
<body>
```

```
<?php
```

```
include('./httpful.phar');
```

```
//Abfragen einer Datei aus der Tabelle "Books" mit der ID 61
```

```
$uri = "https://test.projects.hakmistelbach.ac.at/api/books/61";
```

```
$response = \Httpful\Request::get($uri)
```

```
->authenticateWith('demo', 'demo')
```

```
->send();
```

```
echo '<p>' . $response -> body -> InventoryNumber .' '
```

```
|. $response -> body -> Title .' '. $response -> body -> Author .'</p>';
```

Zweiter Datensatz (ID 20), aber mit einem CSS-Style:

```
21
22 //Referenzen löschen - sonst müsste man einen neuen Namen vergeben
23 unset($uri);
24 unset($response);
25
26 //Abfragen einer Datei aus der Tabelle "Books" mit der ID 20 - mit Style
27 $uri = "https://test.projects.hakmistelbach.ac.at/api/books/20";
28
29 $response = \Httpful\Request::get($uri)
30 ->authenticateWith('demo', 'demo')
31 ->send();
32
33 echo '<p style="color:#F0F">' . $response -> body -> InventoryNumber .' '
34 . $response -> body -> Title .' ' . $response -> body -> Author .'</p>';
```

In Zeile 22-24 werden die Variablen gelöscht, damit man mit der gleichen Bezeichnung weiterarbeiten kann. Man könnte aber auch das Löschen weglassen und die Variablen neu benennen, z.B. „\$uri2“ und „\$response2“

Code:

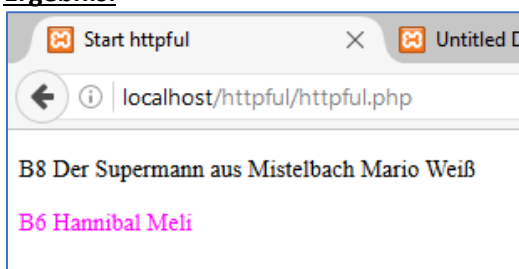
```
//Referenzen löschen - sonst müsste man einen neuen Namen vergeben
unset($uri);
unset($response);
```

```
//Abfragen einer Datei aus der Tabelle "Books" mit der ID 20 - mit Style
$uri = "https://test.projects.hakmistelbach.ac.at/api/books/73";
```

```
$response = \Httpful\Request::get($uri)
->authenticateWith('demo', 'demo')
->send();
```

```
echo '<p style="color:#F0F">' . $response -> body -> InventoryNumber .' '
. $response -> body -> Title .' ' . $response -> body -> Author .'</p>';
```

Ergebnis:



In Datenbank:

```
{
  "Bid": 20,
  "InventoryNumber": "B6",
  "Title": "Hannibal",
  "Author": "Meli",
  "Lended": false
},
```

2) Ausgabe mit Tabellenform:

Nummer	Titel
20	Hannibal

```
38 $uri2 = "https://test.projects.hakmistelbach.ac.at/api/books/20";
39
40 $response2 = \Httpful\Request::get($uri2)
41 ->authenticateWith('demo', 'demo')
42 ->send();
43
44 //Tabellenbeginn mit <table>
45 echo "<table border='1'>";
46
47 //Überschriften zwecks Übersichtlichkeit geteilt auf 2 Zeilen
48 echo "<tr> <td>Nummer</td> <td>Titel</td> </tr>";
49 echo ' <tr> <td> '. $response2 -> body -> Bid . ' </td> <td>'
50 . $response2 -> body -> Title . ' </td> </tr>';
51
52 //Ende mit </table>
53 echo "</table>";
54
55 //Referenzen löschen - sonst müsste man einen neuen Namen vergeben
56 unset($uri2);
57 unset($response2);
58
```

Code:

```
//Tabellenbeginn mit <table>
echo "<table border='1'>";
```

```
//Überschriften zwecks Übersichtlichkeit geteilt auf 2 Zeilen
echo "<tr> <td>Nummer</td> <td>Titel</td> </tr>";
echo ' <tr> <td> '. $response2 -> body -> Bid . ' </td> <td>'
. $response2 -> body -> Title . ' </td> </tr>';
```

```
//Ende mit </table>
echo "</table>";
```

3) Mehrere Elemente aus einer Datei abfragen mit foreach()

```
59 //Mehrere Elemente abfragen
60 $uri = "https://test.projects.hakmistelbach.ac.at/api/books";
61
62 $response = \Httpful\Request::get($uri)
63 ->authenticateWith('demo', 'demo')
64 ->send();
65
66 echo '<ul>';
67 foreach ($response->body as $val) {
68     echo '<li>' . $val -> Bid . '</li>';}
69 echo '</ul>';
70
71 echo '<ul>';
72 foreach ($response->body as $vall) {
73     echo '<li>' . $vall -> Title . '</li>';}
74 echo '</ul>';
75
```

Code:

```
echo '<ul>';
foreach ($response->body as $val) {
    echo '<li>' . $val -> Bid . '</li>';}
echo '</ul>';
```

- 11
- 18
- 20
- 61
- 63
- 64
- 72
- 74
- 76
- 89

- Servus
- Der Supermann aus Mistelbach
- Hannibal
- Der Supermann aus Mistelbach
- asdf
- um
- Der Fall des Lehrers
- Wer weiß das?
- Titel
-

3a)Inkl. Tabelle - Mehrere Elemente aus einer Datei abfragen mit foreach()

Nummer	Titel
11	Servus
18	Der Supermann aus Mistelbach
20	Hannibal
61	Der Supermann aus Mistelbach
63	asdf
64	um

```
82 //Mehrere Elemente abfragen mit Tabelle
83 $uri = "https://test.projects.hakmistelbach.ac.at/api/books";
84
85 $response = \Httpful\Request::get($uri)
86 ->authenticateWith('demo', 'demo')
87 ->send();
88
89 echo "<table border='1'>";
90 echo "<tr><td>Nummer</td><td>Titel</td></tr>";
91 echo '<tr><td>';
92 foreach ($response->body as $val) {echo '<p>' . $val -> Bid . '</p>';}
93 echo '</td>';
94
95 echo '<td>';
96 foreach ($response->body as $val1) {echo '<p>' . $val1 -> Title . '</p>';}
97 echo '</td></tr>';
98 echo "</table>";
99
```

3b)Tabelle – aber schöner – eigene Zellen

lfd.Nr.	ID-Nummer	Titel	Autor
1	11	MorganZoo	Hansi
2	73	Der JavaScript Ritter	Lukas Grafinger2
3	84	Der Supermann aus Mistelbach	Mario Weiß

Die „foreach“ steht am Beginn vor allen „echo“s:

```
foreach($response->body as $val) { }
```

```
118 ▾ foreach($response->body as $val) { //hier beginnt die Abfrage
```

```

106 //Mehrere Elemente abfragen mit Tabelle aber schöner
107 $uri = "https://test.projects.hakmistelbach.ac.at/api/books";
108
109 $response = \Httpful\Request::get($uri)
110 ->authenticateWith('demo', 'demo')
111 ->send();
112
113 echo "<table border='1'>";
114 echo "<tr><td>lfid.Nr.</td><td>ID-Nummer</td><td>Titel</td>
115 <td>Autor</td></tr>"; //das ist die Headline im Kopf der Tabelle
116
117 $i = 1; //benötigt für die fortlaufende Nummerierung
118 foreach($response->body as $val) { //hier beginnt die Abfrage
119 echo '<tr><td>' . $i . '</td>'; //fortlaufende Nummerierung
120 $i++;
121
122 echo '<td>';
123 echo '<p>' . $val -> Bid . '</p>';
124 echo '</td>';
125 echo '<td>';
126     echo '<p>' . $val -> Title . '</p>';
127 echo '</td>';
128 echo '<td>';
129     echo '<p>' . $val -> Author . '</p>';
130 echo '</td></tr>';
131 }
132 echo "</table>";

```

Code:

```

echo "<table border='1'>";
echo "<tr><td>lfid.Nr.</td><td>ID-Nummer</td><td>Titel</td>
<td>Autor</td></tr>"; //das ist die Headline im Kopf der Tabelle

```

```

$i = 1; //benötigt für die fortlaufende Nummerierung
foreach($response->body as $val) { //hier beginnt die Abfrage
echo '<tr><td>' . $i . '</td>'; //fortlaufende Nummerierung
$i++;

```

```

echo '<td>';
echo '<p>' . $val -> Bid . '</p>';
echo '</td>';
echo '<td>';
    echo '<p>' . $val -> Title . '</p>';
echo '</td>';
echo '<td>';
    echo '<p>' . $val -> Author . '</p>';
echo '</td></tr>';
}
echo "</table>";

```


4)Neuanlage in „books“

```
3     "Bid": 11,  
4     "InventoryNumber": "B1",  
5     "Title": "Servus",  
6     "Author": "Hansi",  
7     "Lended": false
```

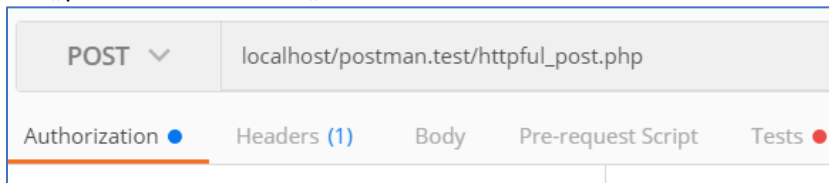
Namen in der Datenbank:

<https://test.projects.hakmistelbach.ac.at/api/books>

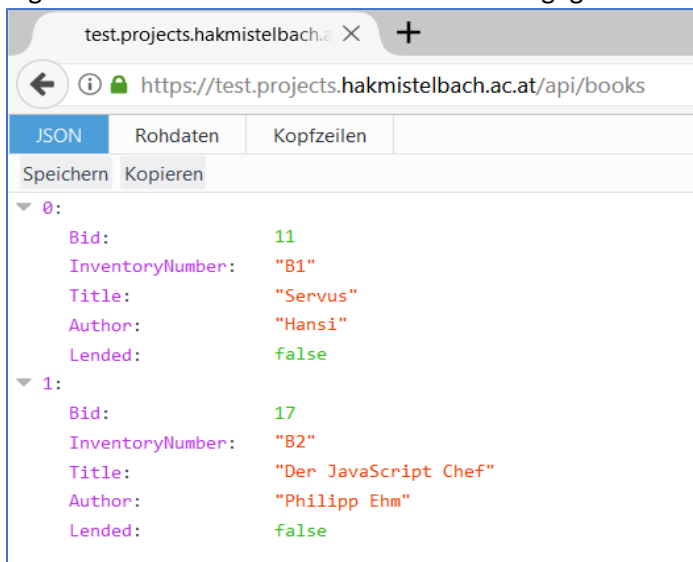
Die „Bid“ nicht anschreiben, da sie automatisch vergeben wird: speichern unter „httpful_post.php“

```
1 <?php  
2 include('httpful.phar');  
3  
4 $uri = "https://test.projects.hakmistelbach.ac.at/api/books";  
5  
6 $response = \Httpful\Request::post($uri)  
7 ->sendJson()  
8 ->authenticateWith('demo', 'demo')  
9 ->body('{  
10     "InventoryNumber": "B2",  
11     "Title": "Der JavaScript Chef",  
12     "Author": "Philipp Ehm",  
13     "Lended": false  
14 }')  
15 ->send();  
16 ?>
```

Mit „postman“ und einer „POST“-Methode an die Datenbank senden:



Ergebnis: wenn die URI direkt im Browser eingegeben wird



Mögliche Fehlerquelle: falsche Bezeichnung z.B. „Autor“ statt „Author“

5) Mit Formulareingabe führt zu einem Eintrag in die Datenbank:

Neue Dateien: „formular_insert.html“ und „formular_insert.php“

Formular mit den <input>-Feldern und der passenden „action“ auf die nachfolgende PHP-Site:

Der <boolean>-Wert bei „lended“ (=ausgeborgt“) sollte hier noch „false“ sein, da das Buch ja noch nicht ausgeborgt sein kann. Der „Boolean“-Wert verträgt auch keine andere Eingabe als „true“ oder „false“.



The screenshot shows a web browser window with the title 'Mail_Formular'. The address bar shows 'localhost/httpful/formular_insert.html'. The main content is a form titled 'Formular bitte ausfüllen'. The form contains the following fields and a button:

- Inventarnummer z.B. B3:
- Buchtitel:
- Autor:
- Ausgeborgt:
- Abschicken

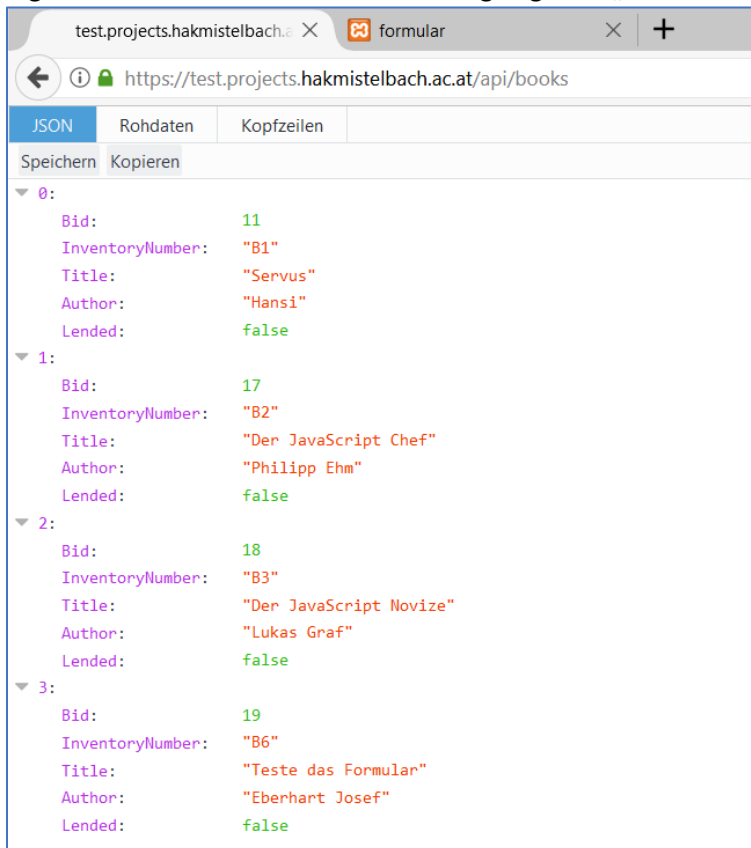
```
8 <body>
9 <h1>Formular bitte ausfüllen</h1>
10 <form action = "formular_insert.php" method = "post" >
11   <p>Inventarnummer z.B. B3: <input name = "ivnr"
12     type="text" size = "5" > </p>
13   <p>Buchtitel: <input name = "title" type="text" size = "50" > </p>
14   <p>Autor: <input name = "author" type="text" size = "50"> </p>
15   <p>Ausgeborgt: <input name = "lended" type="text"
16     size = "7" placeholder="false" > </p>
17 <br>
18 <input type="submit" value="Abschicken">
19 </form>
20
21 </body>
```

```
15 //Neuer Eintrag eines Buches in "book" mit Formular und $_POST-Übernahme
16 $uri = "https://test.projects.hakmistelbach.ac.at/api/books";
17
18 $response = \Httpful\Request::post($uri)
19   ->sendsJson()
20   ->authenticateWith('demo', 'demo')|
21   ->body('{
22     "InventoryNumber": "'.$_POST['ivnr'].'",
23     "Title": "'.$_POST['title'].'",
24     "Author": "'.$_POST['author'].'",
25     "Lended": "false"
26   }');
27 ->send();
28
```

Code:

```
$response = \Httpful\Request::post($uri)
->sendJson()
->authenticateWith('demo', 'demo')
->body('{
    "InventoryNumber": "'.$_POST['ivnr'].'",
    "Title": "'.$_POST['title'].'",
    "Author": "'.$_POST['author'].'",
    "Lended": "false"
}')
->send();
```

Ergebnis: Einfach im Browser öffnen: angelegt hier „B6“

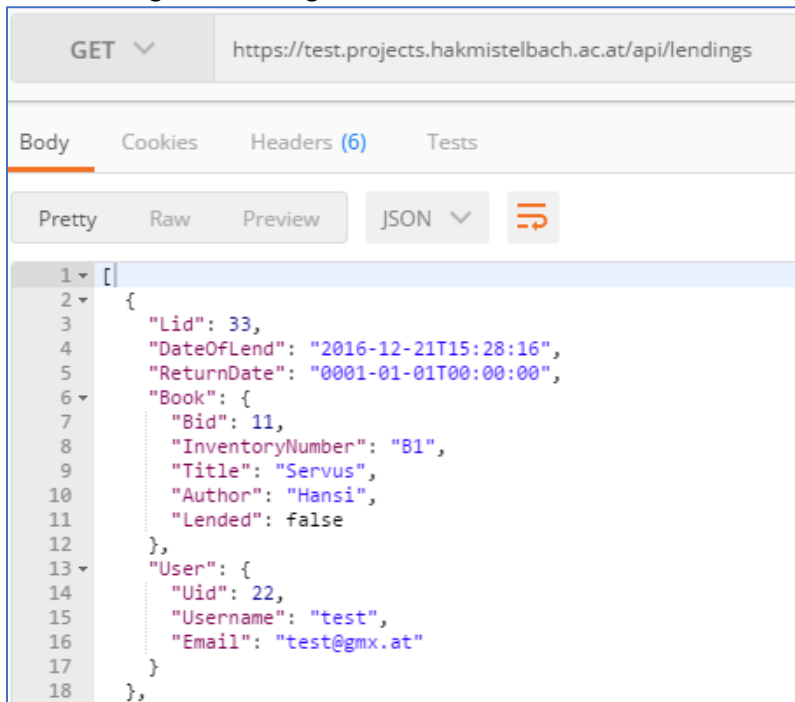


The screenshot shows a web browser window with the address bar displaying `https://test.projects.hakmistelbach.ac.at/api/books`. The page content is a JSON array of four book objects. The browser's developer tools are open, showing the JSON response in a tree view. The fourth object in the array has an `InventoryNumber` of `"B6"`.

Index	Bid	InventoryNumber	Title	Author	Lended
0	11	"B1"	"Servus"	"Hansi"	false
1	17	"B2"	"Der JavaScript Chef"	"Philipp Ehm"	false
2	18	"B3"	"Der JavaScript Novize"	"Lukas Graf"	false
3	19	"B6"	"Teste das Formular"	"Eberhart Josef"	false

6) Eintrag in Datenbank mit Formular, aber Fixeintrag des Datums

Die Zwischentabelle zwischen „books“ und „user“ (=Ausborger) heißt „lendings“ und nimmt neben dem Ausborge-Datum folgende Daten auf:



```
GET https://test.projects.hakmistelbach.ac.at/api/lendings

Body Cookies Headers (6) Tests

Pretty Raw Preview JSON

1 [
2   {
3     "Lid": 33,
4     "DateOfLend": "2016-12-21T15:28:16",
5     "ReturnDate": "0001-01-01T00:00:00",
6     "Book": {
7       "Bid": 11,
8       "InventoryNumber": "B1",
9       "Title": "Servus",
10      "Author": "Hansi",
11      "Lended": false
12    },
13    "User": {
14      "Uid": 22,
15      "Username": "test",
16      "Email": "test@gmx.at"
17    }
18  },
19 ]
```

Die „formular_insert2.php“:

```
15 //Neuer Eintrag eines Verleihs in "lendings" für TEST NEEEEUUU
16 $uri = "https://test.projects.hakmistelbach.ac.at/api/lendings";
17
18 $response = \Httpful\Request::post($uri)
19 ->sendJson()
20 ->authenticateWith('demo', 'demo')
21 ->body('{
22     "DateofLend": "2016-12-21T15:28:16",
23     "ReturnDate": "2016-12-23T15:28:16",
24     "Book": {
25         "InventoryNumber": "'.$_POST['ivnr'].'",
26         "Title": "'.$_POST['title'].'",
27         "Author": "'.$_POST['author'].'",
28         "Lended": "false"
29     },
30     "User": {
31         "Uid": 23,
32         "Username": "Huber",
33         "Email": "hubsi@gmx.at"
34     }
35 }')
36 ->send();
```

Bzw. Fixe Eingabe des „Users“

```
    },
    "User": {
        "Uid": 22,|
        "Username": "test",
        "Email": "test@gmx.at"
    }
}')
->send();
```

Die HTML-Seite: formular_insert2.html:

```
 8 <body>
 9 <h1>Formular bitte ausfüllen</h1>
10 <form action = "formular_insert2.php" method = "post" >
11   <p>Inventarnummer z.B. B3: <input name = "ivnr"
12     type="text" size = "5" > </p>
13   <p>Buchtitel: <input name = "title" type="text" size = "50" > </p>
14   <p>Autor: <input name = "author" type="text" size = "50"> </p>
15   <p>Ausgeborgt: <input name = "lended" type="text"
16     size = "7" placeholder="false" > </p>
17   <p>USER:</p>
18   <p>Uid: <input name = "Uid" type="number" size = "9"> </p>
19   <p>Username: <input name = "Username" type="text" size = "20"> </p>
20   <p>E-Mail:<input name = "Email" type="text" size = "50"> </p>
21 <br>
22 <input type="submit" value="Abschicken">
23 </form>
```