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:

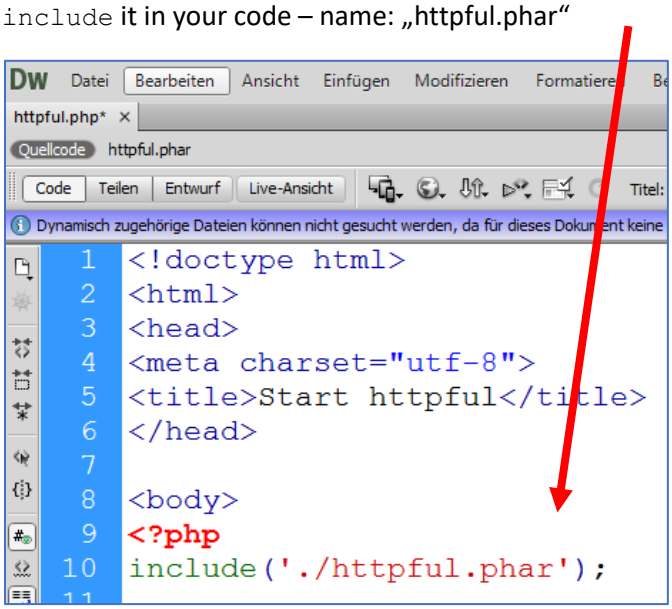
https://hotexamples.com/examples/httpful/Request/put/php-request-put-method-examples.html

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

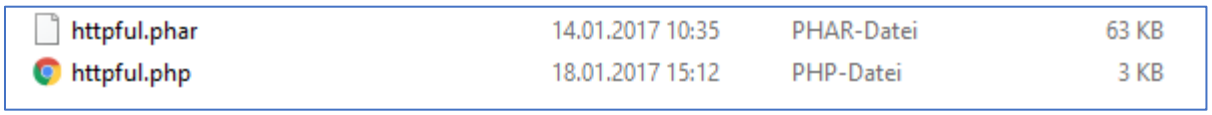
Arbeite in XAMPP

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“



Xampp-Ordner: httpful



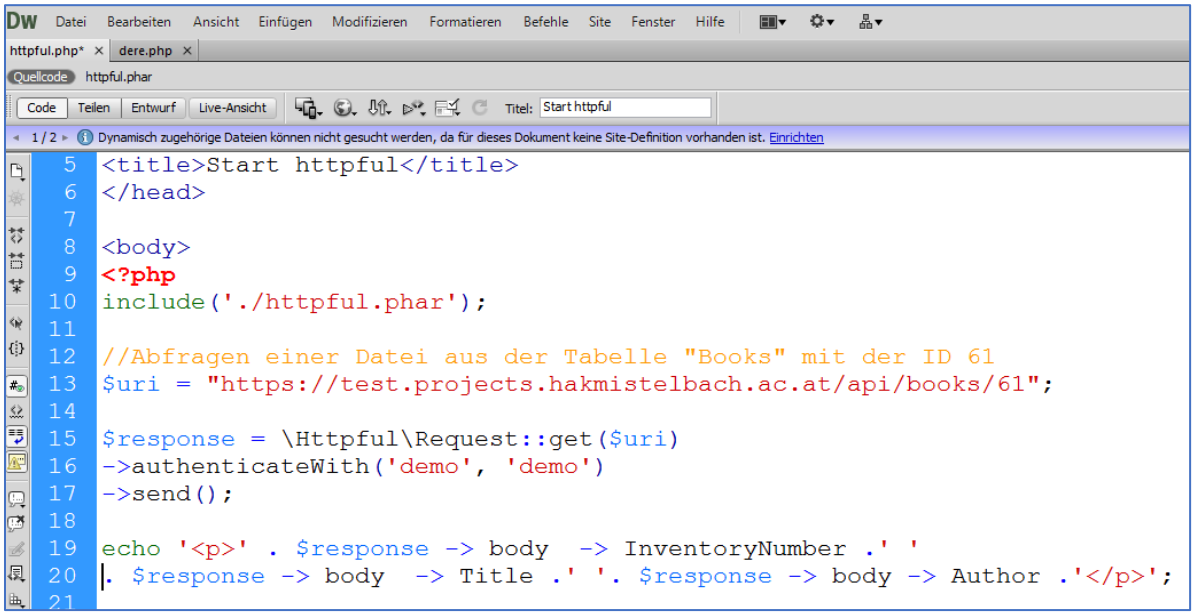
Info allgemein:

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:



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:

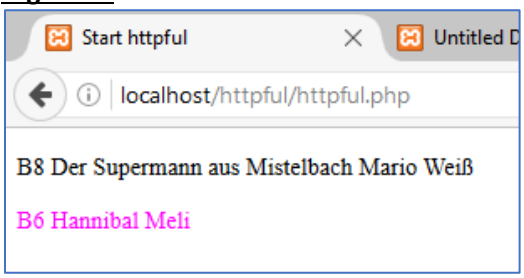
```
21
22 //Referenzen löschen - sonst müsste man einen neuen Namen vergeben
23 unset($uri);
24 unset($response);
25
26 //Abfragen einer Datei aus der Tabelle "Books" mit der ID 20 - mit Style
27 $uri = "https://test.projects.hakmistelbach.ac.at/api/books/20";
28
29 $response = \Httpful\Request::get($uri)
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“

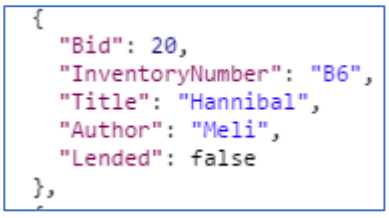
Code:

```
//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:



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);
```

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()

```php
//Mehrere Elemente abfragen
$uri = "https://test.projects.hakmistelbach.ac.at/api/books";

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

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

echo '<ul>';
foreach ($response->body as $val1) {
    echo '<li>' . $val1 -> Title  .'</li>';}
echo '</ul>';
```

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>";
```

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
```

```php
//Mehrere Elemente abfragen mit Tabelle aber schöner
$uri = "https://test.projects.hakmistelbach.ac.at/api/books";

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

echo "<table border='1'>";
echo "<tr><td>lfd.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>";
```

Code:

```
echo "<table border='1'>";
echo "<tr><td>lfd.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“

Namen in der Datenbank:

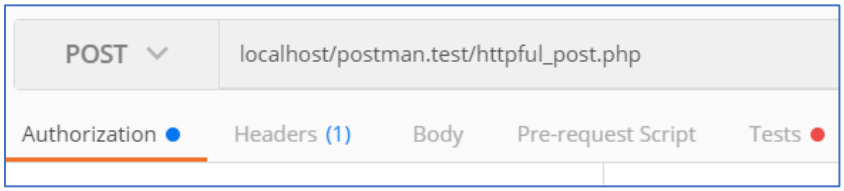
```
    "Bid": 11,
    "InventoryNumber": "B1",
    "Title": "Servus",
    "Author": "Hansi",
    "Lended": false
```

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

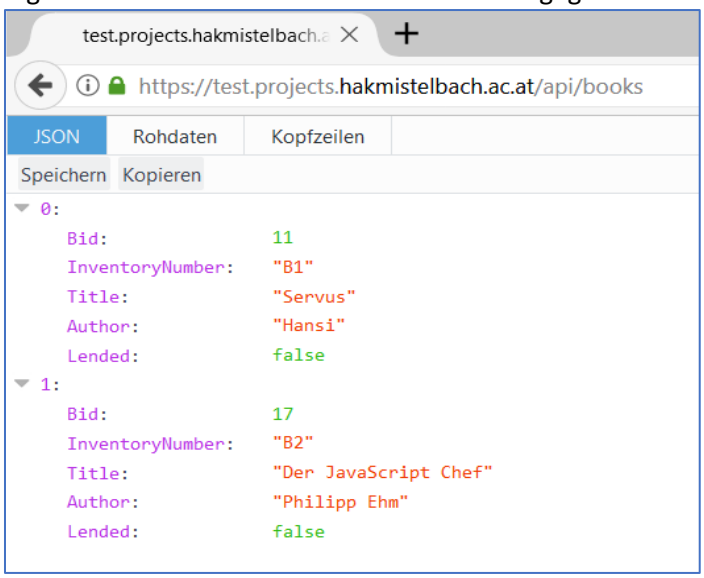
Die „Bid“ nicht anschreiben, da sie automatisch vergeben wird: speichern unter „httpful_post.php“

```php
<?php
include('httpful.phar');

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

$response = \Httpful\Request::post($uri)
->sendsJson()
->authenticateWith('demo', 'demo')
->body('{
    "InventoryNumber": "B2",
    "Title": "Der JavaScript Chef",
    "Author": "Philipp Ehm",
    "Lended": false
}')
->send();
?>
```

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“.

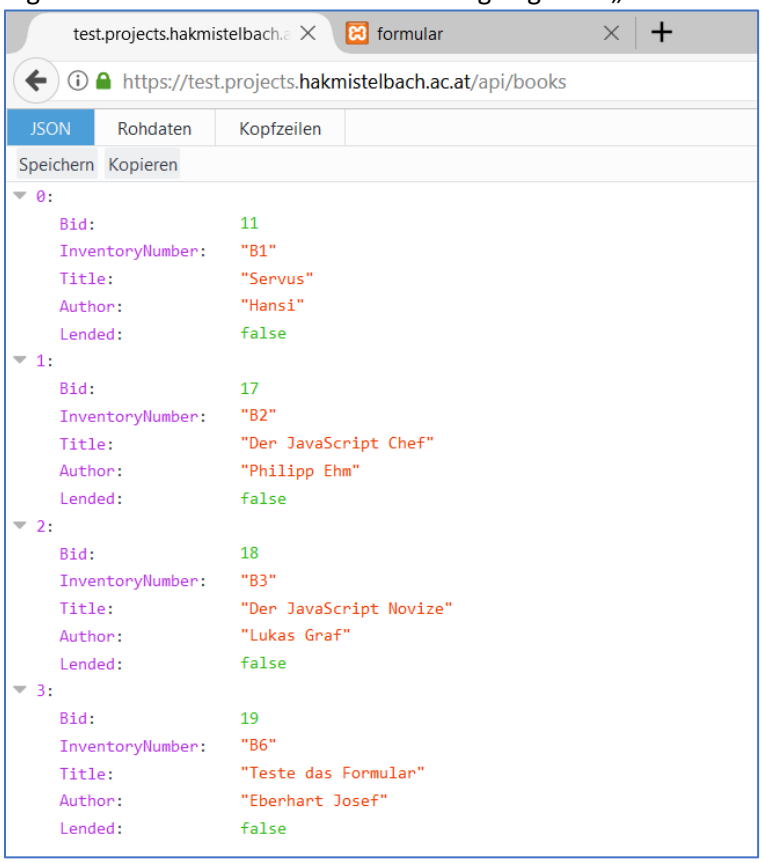

```
<body>
<h1>Formular bitte ausfüllen</h1>
<form action = "formular_insert.php" method = "post" >
  <p>Inventarnummer z.B. B3: <input name = "ivnr"
     type="text" size = "5" > </p>
  <p>Buchtitel: <input name = "title" type="text" size = "50" > </p>
  <p>Autor: <input name = "author" type="text" size = "50"> </p>
  <p>Ausgeborgt: <input name = "lended" type="text"
     size = "7" placeholder="false" > </p>
<br>
<input type="submit" value="Abschicken">
</form>

</body>
```

```
//Neuer Eintrag eines Buches in "book" mit Formular und $_POST-Übernahme
$uri = "https://test.projects.hakmistelbach.ac.at/api/books";

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

Code:

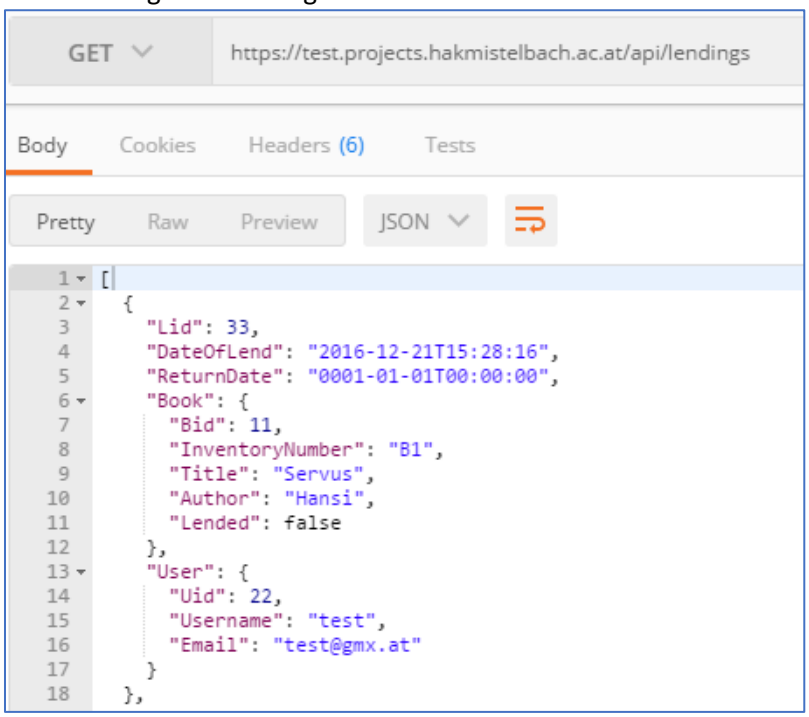
```
$response = \Httpful\Request::post($uri)
->sendsJson()
->authenticateWith('demo', 'demo')
->body('{
        "InventoryNumber": "'.$_POST['ivnr'].'",
        "Title": "'.$_POST['title'].'",
        "Author": "'.$_POST['author'].'",
        "Lended": "false"
 }')
->send();
```

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



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:



Die „formular_insert2.php“:

```php
//Neuer Eintrag eines Verleihs in "lendings" für TEST NEEEUUUU
$uri = "https://test.projects.hakmistelbach.ac.at/api/lendings";

$response = \Httpful\Request::post($uri)
->sendsJson()
->authenticateWith('demo', 'demo')
->body('{
    "DateofLend": "2016-12-21T15:28:16",
    "ReturnDate": "2016-12-23T15:28:16",
    "Book": {
        "InventoryNumber": "'.$_POST['ivnr'].'",
        "Title": "'.$_POST['title'].'",
        "Author": "'.$_POST['author'].'",
        "Lended": "false"
    },
    "User": {
        "Uid": 23,
        "Username":  "Huber",
        "Email":  "hubsi@gmx.at"
    }
  }')
->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>
```