

PHP: forms

Key concepts

\$_GET, \$_POST, \$_REQUEST, \$_COOKIE, \$_SESSION, forms method='get' vs method='post', request, response, hidden input

1. \$_GET, \$_POST

Gegevens die ingevuld worden in een formulier worden in de voorgedefinieerde arrays \$_GET, \$_POST (en \$_REQUEST) geplaatst.

Wanneer in het formulier method = 'get' gekozen worden, dan worden de ingevulde waarden getoond in de URL in de browser. Deze waarden komen als value in de array \$_GET te staan, de key komt overeen met het attribuut name in het formulier.

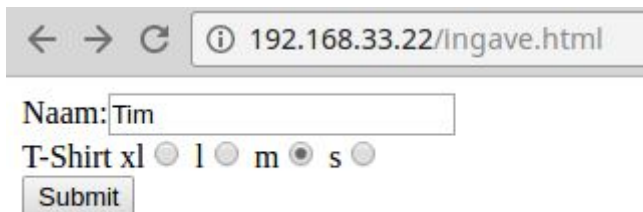
ingave.html

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <title>ingave</title>
</head>
<body>
  <form action="verwerking.php" method="get">
    Naam:<input type="text" name="naam"></input><br/>
    T-Shirt xl<input type="radio" name="tshirt" value="xl"/>
    l<input type="radio" name="tshirt" value="l"/>
    m<input type="radio" name="tshirt" value="m"/>
    s<input type="radio" name="tshirt" value="s"/><br/>
    <input type="submit">
  </form>
</body>
</html>
```

verwerking.php

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <title>verwerking</title>
</head>
<body>
  Naam:<?php print( $_GET['naam'] ); ?><br/>
  T-Shirt <?php print( $_GET['tshirt'] ); ?><br/>
</body>
</html>
```

De werking van het voorbeeld wordt hieronder getoond:



← → ↻ ⓘ 192.168.33.22/ingave.html

Naam:

T-Shirt xl ☐ l ☐ m ☒ s ☐



← → ↻ ⓘ 192.168.33.22/verwerking.php?naam=Tim&tshirt=m

Naam:Tim

T-Shirt m

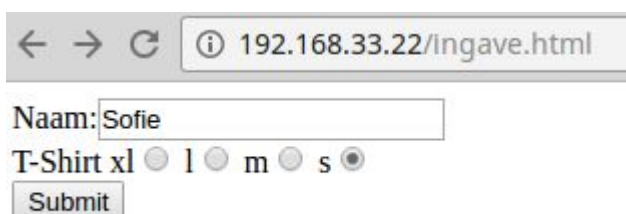
Wanneer we in ingave.html method="get" vervangen door method="post" en in verwerking.php \$_GET vervangen door \$_POST krijgen we de volgende situatie. De ingevulde gegevens worden niet getoond in de URL maar worden verstopt in de request body. Dit kan getoond worden via de Chrome developer tools (ctrl-shift-i) in Chrome, door op Network te klikken. Let op Form Data.

ingave.html

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <title>ingave</title>
</head>
<body>
  <form action="verwerking.php" method="post">
    Naam:<input type="text" name="naam"></input><br/>
    T-Shirt xl<input type="radio" name="tshirt" value="xl"/>
    l<input type="radio" name="tshirt" value="l"/>
    m<input type="radio" name="tshirt" value="m"/>
    s<input type="radio" name="tshirt" value="s"/><br/>
    <input type="submit">
  </form>
</body>
</html>
```

verwerking.php

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <title>verwerking</title>
</head>
<body>
  Naam:<?php print( $_POST['naam'] ); ?><br/>
  T-Shirt <?php print( $_POST['tshirt'] ); ?><br/>
</body>
</html>
```



← → ↻ ⓘ 192.168.33.22/ingave.html

Naam:

T-Shirt xl ☐ l ☐ m ☐ s ☒

The screenshot shows the Network tab of a web browser's developer tools. The selected request is 'verwerking.php'. The 'Form Data' section is highlighted with a red box, showing the following data:

- naam: Sofie
- tshirt: s

Waarden die meerdere keren ingevuld moeten worden, kunnen als een array in \$_GET en \$_POST geplaatst worden. In ingave.html kunnen meerdere vrienden ingegeven worden. Er wordt in het formulier gewerkt met de array vriend. In verwerking.php bevat \$_GET['vriend'] een array met alle ingevulde waarden.

ingave.html

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>ingave</title>
</head>
<body>
    <form action="bevestig.php" method="post">
        Vrienden:<br/>
        <input type="text" name="vriend[]"><br/>
        <input type="text" name="vriend[]"><br/>
        <input type="text" name="vriend[]"><br/>
        <input type="text" name="vriend[]"><br/>
        <input type="text" name="vriend[]"><br/>
        <input type="submit">
    </form>
</body>
</html>
```

verwerking.php

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>verwerking</title>
</head>
<body>
<?php
foreach ( $_GET['vriend'] as $index => $vriend ) {
    print( $index. ': '. $vriend . '<br/>' );
}
?>
</body>
</html>
```

Vrienden:

Tim
Sofie
Nele
Joris
An

Submit

0: Tim
1: Sofie
2: Nele
3: Joris
4: An

Let bij de screenshot van verwerking.php in de browser op de urlencoding in de URL: [wordt geëncodeerd als %5B ,] wordt geëncodeerd als %5D.

2. Hidden input

Waarden die van het ene formulier doorgegeven moeten worden naar een ander formulier kunnen in een hidden input geplaatst worden. In onderstaand voorbeeld wordt in ingave.html een naam ingegeven waarna de gebruiker naar bevestig.php gestuurd wordt. De waarde voor naam wordt in bevestig.php in een hidden input geplaatst zodanig dat deze waarde ook beschikbaar is in verwerking.php.

ingave.html

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <title>ingave</title>
</head>
<body>
  <form action="bevestig.php" method="get">
    Naam:<input type="text" name="naam"/><br/>
    <input type="submit">
  </form>
</body>
</html>
```

bevestig.php

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>bevestig</title>
</head>
<body>
    De ingegeven waarde voor naam is
    <?php print($_GET['naam']);?>.
    <form action="verwerking.php" method="get">
        <input type="hidden" name="naam"
            value="<?php print($_GET['naam']);?>" />
        <input type="submit" value="bevestig" />
    </form>
    <a href="ingave.html">terug</a>
</body>
</html>
```

verwerking.php

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>verwerking</title>
</head>
<body>
    De bevestigde waarde voor naam is <?php
print($_GET['naam']);?>.
</body>
</html>
```

← → ↻ ⓘ 192.168.33.22/Ingave.html

Naam:

← → ↻ ⓘ 192.168.33.22/bevestig.php?naam=Tim

De ingegeven waarde voor naam is Tim.

[terug](#)

← → ↻ ⓘ 192.168.33.22/verwerking.php?naam=Tim

De bevestigde waarde voor naam is Tim.

3. Scheiding van logica en presentatie

In alle voorbeelden van dit hoofdstuk wordt de logica en de presentatie op dezelfde locatie behandeld. Het volgende voorbeeld gaat hier nog een stap in verder: in hetzelfde bestand wordt er afhankelijk of er een waarde voor naam ingegeven is de verwerking van de ingave of het formulier voor de ingave getoond.

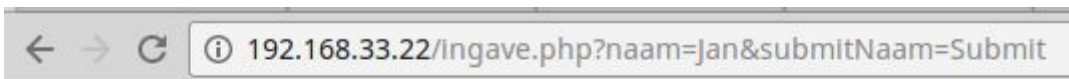
Later in de cursus zullen we zien dat logica, presentatie (en ook toegang tot databanken) best gescheiden gehouden worden. Maar voor de oefeningen van de inleidende hoofdstukken is dit nog niet belangrijk.

ingave.php

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <title>verwerking</title>
</head>
<body>
<?php
if (isset($_GET['naam'])) {
    print('Ingevoerde waarde voor naam:' . $_GET['naam']);
} else { ?>
    <form action="ingave.php" method="get">
        Naam:<input type="text" name="naam"/><br/>
        <input type="submit" name="submitNaam">
    </form>
<?php } ?>
</body>
</html>
```



Naam:



Ingevoerde waarde voor naam:Jan

4. Cookies en sessions

Cookies bevatten gegevens die door de server naar de browser gestuurd worden, waar ze bewaard worden voor later gebruik. Een typisch voorbeeld van het gebruik van cookies is het bewaren van keuzes bij het bezoeken van een website. Een gebruiker kiest bijvoorbeeld de eerste keer dat hij een site bezoekt Nederlands als taal. Deze waarde wordt vervolgens bewaard in een cookie en de gebruiker wordt doorverwezen naar de Nederlandse pagina's.

Cookies worden aangemaakt via de functie `setcookie`. Deze functie wordt meestal met drie argumenten gebruikt. Bijvoorbeeld de code

```
setcookie ('taal', 'NL',time() + 24 * 60 *60 );
```

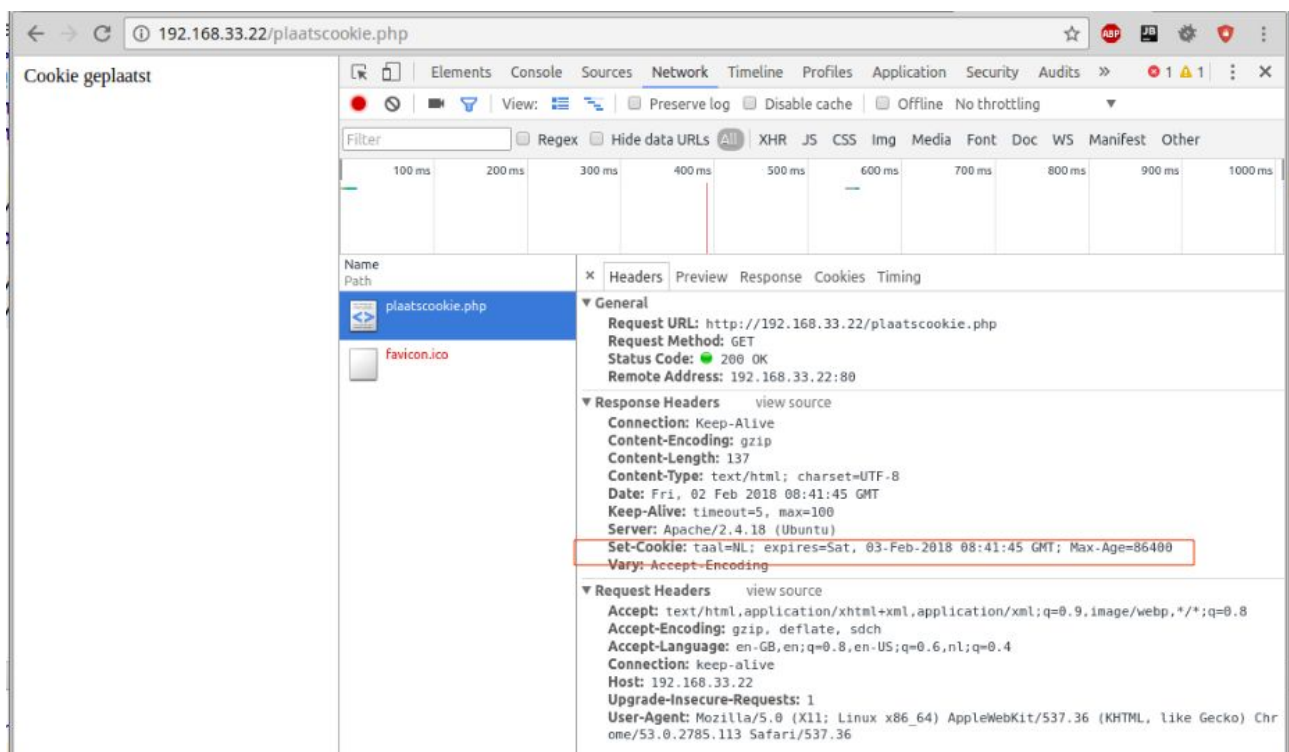
maakt een cookie aan met naam `taal` en inhoud `'NL'`, het cookie blijft 1 dag (= 24 *60*60 seconden) geldig. Het aanroepen van `setcookie` moet altijd gebeuren in het begin van een script, voor er gegevens naar de client verstuurd worden.

In de onderstaande screenshot wordt de communicatie tussen server en browser getoond wanneer `plaatscookie.php` geopend wordt. In de response staat de regel die de opdracht bevat om de cookie in de browser te plaatsen.

Set-Cookie:taal=NL; expires=Sat, 03-Feb-2018 08:41:45 GMT; Max-Age=86400

plaatscookie.php

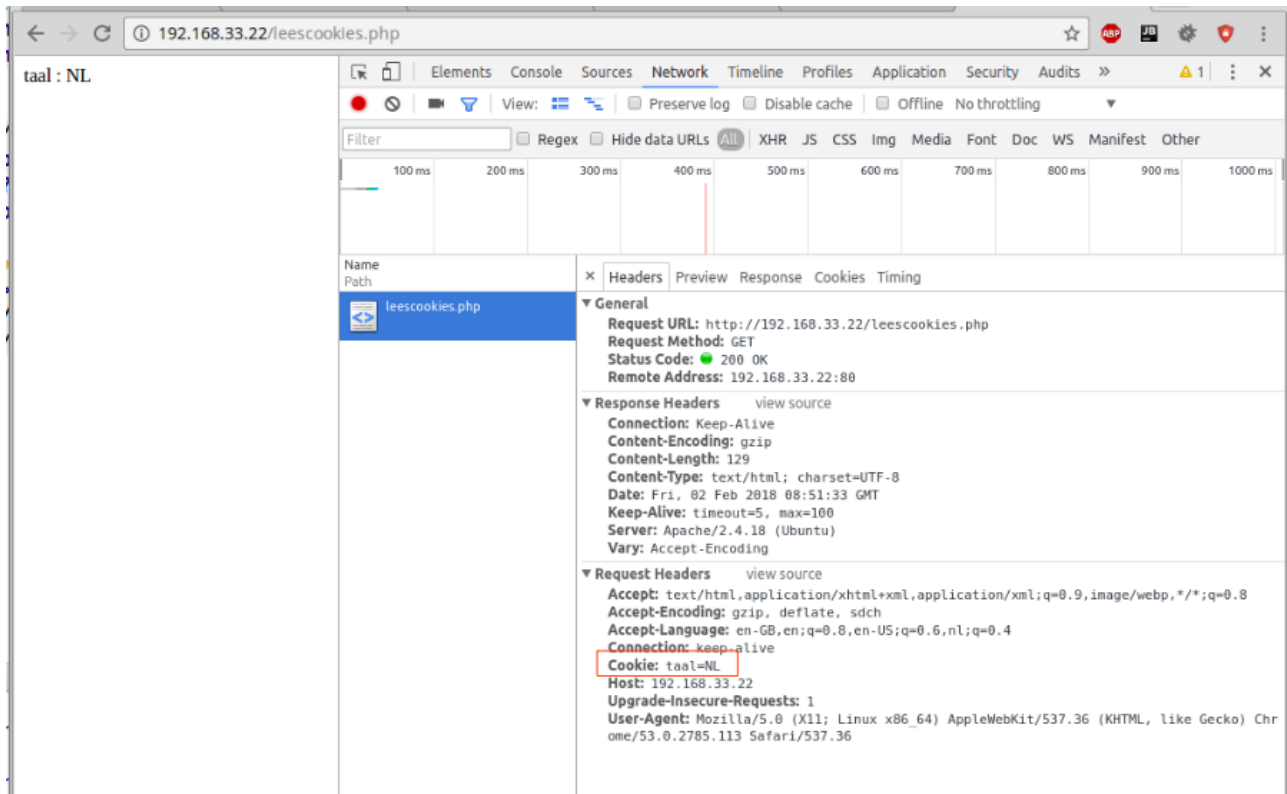
```
<?php
setcookie ('taal', 'NL',time() + 24 * 60 *60 );
?>
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>plaatscookie</title>
</head>
<body>
    Cookie geplaatst
</body>
</html>
```



De waarden van een cookie kunnen gelezen worden via de rij `$_COOKIE`. In leescookies worden alle key value paren afgedrukt. Enkel de waarde voor taal afdrukken kan natuurlijk ook via de regel `print ($_COOKIE['taal']);`. Zoals getoond in de onderstaande screenshot stuurt de browser in de request alle cookies mee naar de server.

leescookies.php

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <title>leescookie</title>
</head>
<body>
<?php
foreach($_COOKIE as $key => $value){
    print( $key. ' : '. $value );
}
?>
</body>
</html>
```



Cookies kunnen worden verwijderd door een tijd in het verleden te kiezen:

```
setcookie ("user", "", time() - 3600);
```

Bij cookies worden alle gegevens bewaard bij de client. Bij sessies worden de gegevens bewaard op de server, maar krijgt de client nog wel een cookie met daarin het session_id om de juiste gegevens te kunnen ophalen.

Een sessie wordt gestart via het commando

```
session_start();
```

Gegevens worden bewaard en uitgelezen via de array \$_SESSION:

```
$_SESSION['name'] = 'Sofie';
```

```
print ( $_SESSION['name'] );
```

Zoals getoond in onderstaande screenshot worden de gegevens op de server bewaard in de map /var/lib/php/sessions/.

```
vagrant@webadv:~$ sudo su -  
root@webadv:~# cat /var/lib/php/sessions/sess_ljsbnh7iq9rgkca962lo324fa1  
name|s:5:"Sofie";root@webadv:~#
```