4.2 Calculator+ deel 2 (cookies)

1. In deze les leren we:
2. onze class gaan we uitbreiden
3. we leren wat stateless (en statefull) is
4. we leren met cookies te werken: creëren, uitlezen en debuggen
5. we gaan een string omzetten met explode zoals we dat eerder hebben gedaan in de oefeningen.
6. we gaan alle elementen van een array afdrukken in een loop
7. we gaan het gemiddelde berekenen van een array

We hebben een calculator die bestaat uit (1) een form, (2) een resultaten pagina en (3) een class.

Stel dat we nu een serie getallen willen invoeren en dat we het gemiddelde willen uitrekenen. Hoe kunnen we dit dan doen?

We kunnen ons object gebuiken en een array gebruiken waar we elke keer een element aan toe voegen. Dit hebben we geoefend in de opgaven die we eerder hebben gemaakt. Weet je nog dat we de functie array_push gebruikte?

We zouden in de lijst van het form een functie kunnen maken die we 'avg' noemen. Hiermee kunnen we dan het gemiddelde berekenen. We willen dit als volgt maken:

Elke keer als we een getal invoeren en de functie `avg` kiezen dan wordt het getal aan een array toegevoegd en wordt het gemiddelde van alle getallen berekend.

Dus

- je voert 5 in met de nieuwe functie avg en drukt op submit,
- je ziet als resultaat 5 en het gemiddelde is 5
- je voert nu 15,
- je ziet als resultaat 5,15 en het gemiddelde 10
- je voert nu 1 in en je ziet als reultaat 5,15,1 en het gemiddelde 7

Een webserver is stateless:

> *Stateless is the polar opposite of stateful, in which any given response from the server is **independent of any sort of state**.*

Dit betekent dat de webserver geen dingen onthoud, althans niet vanzelf. Elke keer als je de webserver iets vraagt weet hij niets van de vorige keer dat je er was. En toch weten webserver zoals Facebook precies wie jij bent en wat jij allemaal hebt gedaan. Hoe kan dat?

Dat komt omdat Facebook (en andere sites) deze gegevens opslaat en koppelt aan jouw als gebruiker. je logt aan en Facebook weet dan wie je bent. Facebook zoekt dan alle gegevens die bij jou horen op en gebruikt die.

> Vraag: hoe weet een site als Facebook bij elke keer alsje wat vraagt wie jij bent; je stuurt immers niet elke keer je userID en password mee?

Dus - terug naar ons gemiddelde- als we willen dat er een array wordt opgebouwd dan moeten we de status van het array bijhouden. Dit kan met een database maar dan moeten we ook een log-in maken. In een database moeten de gegevens immers aan jouw worden gekoppeld. Waarom? Nou stel dat er gelijktijdig met jou een andere gebruiker ook een gemiddelde wil berekenen, en je zou de gegevens niet per gebruiker opslaan, dan zouden jullie elkaar gegevens gebruiken.

Als dit verhaal niet duidelijk is, vraag dan om uitleg. Dit is een belangrijk onderwerp om te begrijpen en om inzicht te krijgen hoe een webserver werkt.

Er zijn andere methodes om (beperkt) gegevens vast te houden. Dit zijn sessie-varieblen en cookies.

Cookies worden op de client (browser) opgeslagen en sessie-variabelen worden op de server opgeslagen. Voor onze toepassing gaan we cookies gebruiken. We gebruiken de cookie als een soort mini-database-je.

We gaan terug naar onze class Cals en voegen aan de switch-case constructie een case toe. Noem die 'avg' (average/gemiddelde). Als deze functie wordt aangeroepen dan maken we een cookie waarin we de waarde van $number1 zetten.

```
switch($function) {
case('add'):
...
case('avg'):
    setcookie("myCalc", $this->number1, time()+60);
    break;
...
}
```

Hiermee maken we dus een cookie met de naam myCalc, geven het de waarde $this->number1 en we laten dit cookie verlopen na 1 minuut.

Voeg de optie avg (average) ook to aan het form.

```
<option value="avg">average</option>
```

Testen van cookie

Ga nu naar het form, vul een getal in en kies de optie avg. Post het form en als het goed is zou het cookie moeten worden gezet.

Zoek op internet op hoe je met joiw browser je cookies kunt zien. Bij Firefox moet je onder het developer menu kiezen voor 'Storage Inspector'. Je krijgt dan het volgende te zien:

Je ziet (in geel gearceerd) dat het cookie is aangemaakt en dat het de waarde 12 heeft.

We gaan nu niet alleen de waarde opslaan, maar ook de vorige waarde bewaren. Opgelet!

In de class definitie bij de case("avg") wordt nu het cookie gezet, maar voordat we dat doen gaan we eerst kijken of er al een cookie eerder is gezet met `$_COOKIE["myCalc"]` kunnen we de bestaande cookie opvragen, maar we weten niet zeker of die al is gezet. De allereerste keer is het cookie immers nog niet gezet. We moeten dus testen of de variabele `$_COOKIE["myCalc"]` bestaat. Dat doen we met de functie `isset()`.

Dus met isset testen we of het cookie al bestaat. Als dat zo is dan plakken we de nieuwe waarde aan de bestaande waarde.

```
$value=$_COOKIE["myCalc"].'-'. $this->number1;
```

We gebruiken '-' (min-teken) als koppel teken om de verschillende waarden te scheiden. Dit doen we omdat dit makkelijk leesbaar is. Let op dat de nieuwe cookie waarde nu in `$value` staat en nog niet in het cookie.

Als met de isset functie wordt bepaald dat er nog geen cookie is gezet dan stellen we deze in met:

```
$value= $this->number1;
```

Ook hier geldt dat de nieuwe cookie waarde nu in de variabele $value staat en nog niet in het cookie.

We moeten dus het cookie nog vullen met de waarde `$value`.

```
setcookie("myCalc", $value, time()+60);
```

Check of het geheel nu werkt.

In dit voorbeeld heeft de cookie de waarden 1,2 3 en 5.

We zijn er bijna!

Het enige dat we nog moeten doen is de waarden afdrukken en het gemiddelde berekenen.

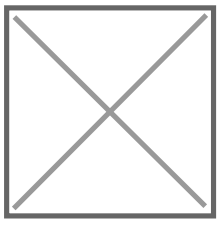
Laten we de `case("avg")` de waarde van de cookie als return waarde terug geven aan de `results.php`.

In de result pagina moeten we nu onderscheid maken tussen het geval als de avg functie gekozen is en als er een andere functie wordt gekozen.

```
<?php
...
if ( $_GET['function'] == 'avg' ){
    $result = $myCalcObject->executeCalculation($_GET['function']);
  // zet de variabele $result om in een $myArray, gebruik de explode functie
// maak een loop en druk alle elementen van het $myArray af.
    // druk het gemiddelde van het array af; gebruik array_sum en count
}else{
    echo "Result is: ".$myCalcObject->executeCalculation($_GET['function']);
}
...
?>
```

Maak de code af, lees het commentaar en vervang deze door code.

De output moet er ongeveer zo uit zien (nadat je 2 maal de functie avg hebt aangeroepen).



Je kunt nu de output nog een beetje mooier maken, gebruik een table en gebruik wat CSS.

De opdracht is klaar.

Revision #18
Created 2019-09-10 16:07:48 UTC by Admin
Updated 2020-05-08 08:48:45 UTC by Max