Formuláře

PVA4 Programování a vývoj aplikací

Adam Fišer

Obsah

HTML formulář

- HTML formulář je základním prvkem pro interakci s uživatelem.
- Umožňuje uživateli zadat data, která jsou odeslána na server.
- Formulář je tvořen HTML tagy `<form>`, `<input>`, `<textarea>`, `<select>`, `<button>`, `<label>`, `<fieldset>`, `<legend>`, `<optgroup>`, `<option>`, `<datalist>`, `<output>`.
- `action` - cílová URL, kam se mají data odeslat.
- `method` - metoda odeslání dat, GET nebo POST.

```
<form action="destUrl.php" method="post">
Name:   <input type="text" name="name"><br>
E-mail: <input type="text" name="email"><br>
<input type="submit" value="Odeslat">
</form>
```

Metoda GET

Metoda `GET`

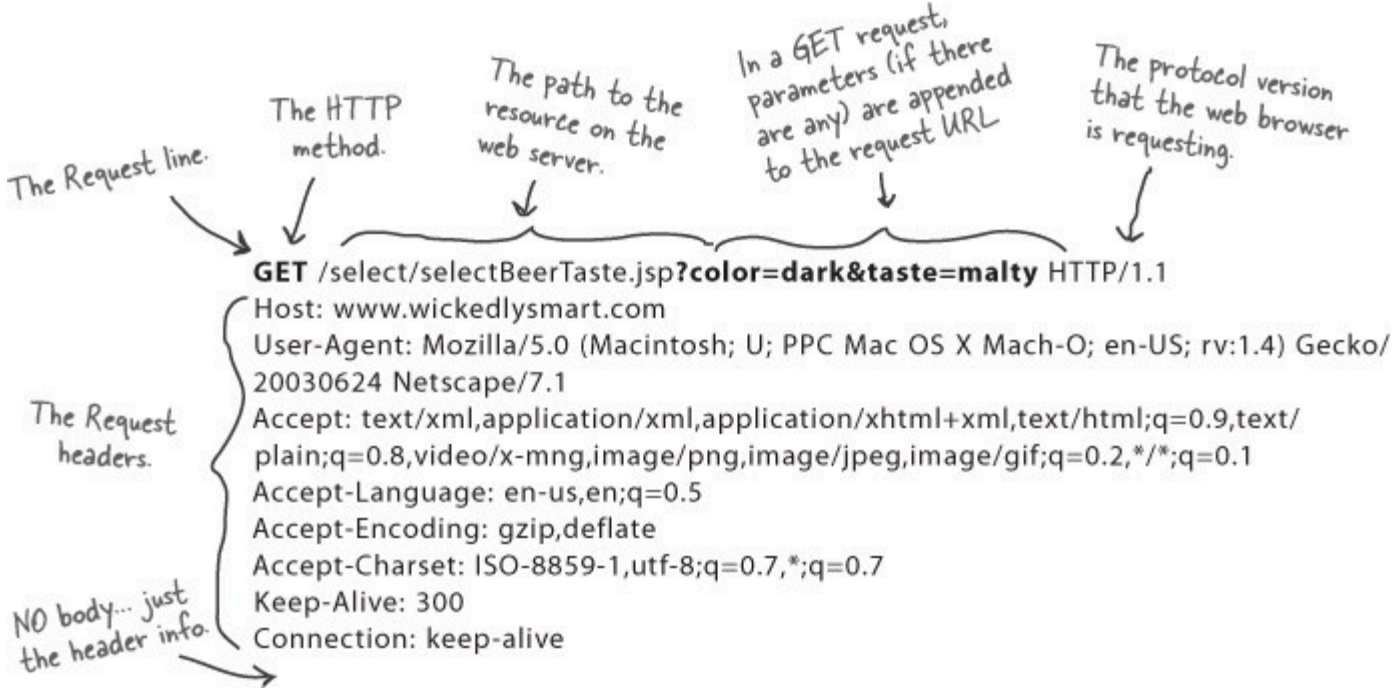
- Jedná se o vestavěnou superglobální proměnnou PHP pole, která se používá k získání hodnot odeslaných prostřednictvím metody HTTP GET.
- K proměnné pole lze přistupovat z libovolného skriptu v programu; má globální rozsah.
- Zobrazuje hodnoty formuláře v adrese URL.
- Je ideální pro formuláře vyhledávačů, protože umožňuje uživatelům uložit a zpracovávat výsledky.

Výhody GET

- Metodou GET lze získat informace identifikované pomocí požadavku-URl (Uniform Resource Identifier).
- Požadavky GET lze zobrazit v historii prohlížeče.
- Umožňuje uložit výsledky formuláře HTML.
- Metodu GET můžete snadno použít k vyžádání požadovaných údajů.

```
echo '
    <form action="destUrl.php" method="get">
        Name:   <input type="text" name="name"><br>
        E-mail:  <input type="text" name="email"><br>
        <input type="submit">
    </form>
';
```

Přístup na hodnoty



Přístup na hodnoty

```
echo '
    <form action="destUrl.php" method="get">
        Name:   <input type="text" name="name"><br>
        E-mail:  <input type="text" name="email"><br>
        <input type="submit" value="odeslat">
    </form>
';
```

Hodnoty jsou předávány přes URL

```
url: https://localhost/PVA4/destUrl.php?name=value&email=value
```

Přístup na hodnoty formuláře:

```
<?php

$_GET['name'];
$_GET['email'];

?>
```

Metoda POST

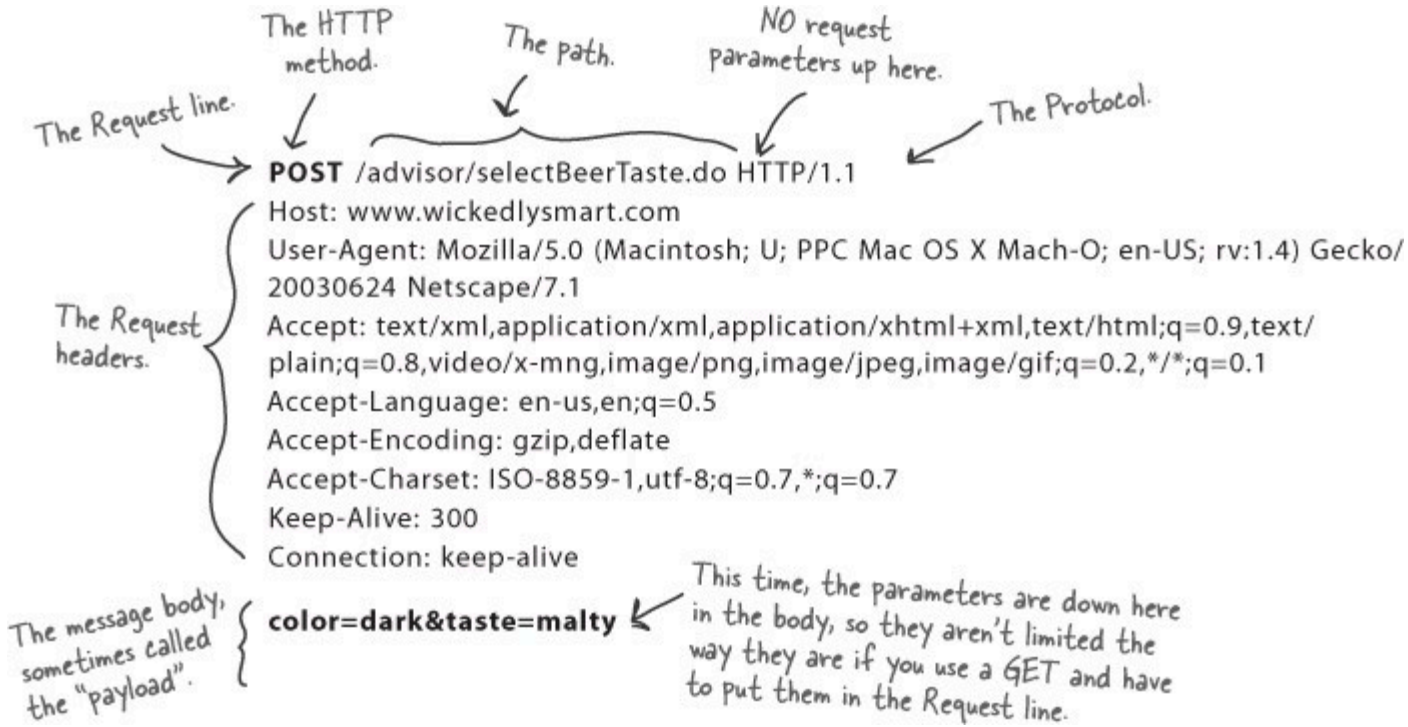
Metoda `POST`

- Metoda POST je způsob odesílání dat z formuláře na server.
- Data odeslaná pomocí metody POST nejsou viditelná v URL.
- Jedná se o vestavěnou superglobální proměnnou PHP typu pole, která se používá k získání hodnot odeslaných prostřednictvím metody HTTP POST.
- K proměnné lze přistupovat z libovolného skriptu v programu; má globální rozsah.
- Metoda je ideální v případě, že nechcete zobrazovat hodnoty odeslaného formuláře v adrese URL.
- Dobrým příkladem použití metody post je odesílání přihlašovacích údajů na server.

Výhody POST

- Metoda POST je bezpečnější než metoda GET, protože data nejsou viditelná v URL.
- Požadavky nezůstávají v historii prohlížeče.
- Metoda POST umožňuje odesílat velké objemy dat.
- Na webový server můžete odeslat data vytvořená uživatelem. Velmi užitečné, pokud nemáte žádnou představu o prostředku, který máte v adrese URL uchovávat.
- POST použijte, pokud potřebujete server, který řídí generování adres URL vašich zdrojů.
- Pomocí POST můžete bez námahy přenášet velké množství dat. Data můžete udržovat v soukromí. Lze použít k odesílání binárních i ASCII dat.

Přístup na hodnoty



Přístup na hodnoty

```
echo '
    <form action="destUrl.php" method="post">
        Name:   <input type="text" name="name"><br>
        E-mail:  <input type="text" name="email"><br>
        <input type="submit" value="odeslat">
    </form>
';
```

Přístup na hodnoty formuláře:

```
<?php

$_POST['name'];
$_POST['email'];

?>
```

Srovnání POST a GET

POST	GET
Hodnoty nejsou viditelné v URL	Hodnoty jsou viditelné v URL
Není omezena délka hodnot odeslaných přes body HTTP	Omezená délka odeslaných dat běžně na 255, výjimečně až 2000 znaků. Omezení je dáno prohlížečem a maximální délkou zobrazenou a zpracovávanou v URL
Nižší výkon ve srovnání s PHP_GET kvůli času zapouzdření do těla HTTP	Ve srovnání s metodou GET má vyšší výkon ve zpracování pro svou jednoduchou povahu přidávání do URL

Srovnání POST a GET

POST	GET
Podporuje více datových typů např. string, číslo, binární data	Podporuje pouze základní datový typ string
Výsledky nelze uložit do záložky prohlížeče.	URL může být uložena v záložkách probhlížeče, neboť adresa obsahuje všechny data
Velmi těžko se cachuje	Je často cachovatelný
Parametry nejsou uloženy v historii prohlížení.	Parametry zůstávají v historii prohlížeče.

Zpracování dat formuláře

- Data odeslaná formulářem mohou být zpracována na serveru pomocí PHP.
- Data jsou uložena v superglobálních polích `$_GET` a `$_POST`.
- Data jsou uložena v asociativním poli, kde klíčem je název prvku formuláře a hodnotou je hodnota prvku formuláře.
- Každý prvek nutno ošetřit proti nežádoucím vstupům vlastní funkcí nebo min `htmlspecialchars`.
- htmlspecialchars() - funkce, která převede speciální znaky na HTML entity. Tím se zabrání XSS útokům.
- Nutno vždy zkontrolovat, zda byla data odeslána a zda obsahují očekávané hodnoty.

Po odeslání formuláře

POST

```php
// Podmínka kontrolující odeslání formuláře – metoda POST
// Pokud podmínka neexistuje, skript se spustí i při prvním načtení stránky
if ($_SERVER["REQUEST_METHOD"] == "POST") {

  // Tento kód je realizován, po odeslání formuláře
  $name = htmlspecialchars($_POST['name']);
  echo '<h2>Thank You '. $name .'</h2>';
}
```

GET

```php
if ($_SERVER["REQUEST_METHOD"] == "GET") {

  // Tento kód je realizován, po odeslání formuláře
  $name = htmlspecialchars($_GET['name']);
  echo '<h2>Thank You '. $name .'</h2>';
}
```

Bezpečnostní riziko

- Nutno vždy ošetřovat vstupy od uživatele.
- Ošetření vstupů Javascriptem je vhodné, ale nedostatečné.
- Vždy musí být ošetřeno min. na serverové úrovni aplikace.
- Přes GET nikdy neposílat hesla uživatelů.
- Přes POST nikdy neposílat hesla uživatelů v čitelné podobě.
- Vždy ošetřit vstupy proti SQL Injection, XSS, CSRF, atd.

Děkuji za pozornost

Otázky?

Repository / Prezentace