

Systemutveckling PHP

Föreläsning 12 - PDO & lösenord

Dagens ämnen

- SQL injections
- Lösenord
- Livekoda inloggning

SQL injections

```
<form method="post">  
  <input type="text" name="username">  
  <input type="password" name="password">  
  <input type="submit" name="action" value="Logga in">  
</form>
```

```
$username = $_POST['username'];  
$password = $_POST['password'];
```

```
$query = "SELECT * FROM users WHERE user='$name' AND password =  
'$password'";
```

```
$username = "Bobby";DROP TABLE users; -- "
```

```
$query = "SELECT * FROM users WHERE user='$name' AND password =  
'Bobby';DROP TABLE users; -- '";
```

SQL injections

```
$stmt = $pdo->prepare('SELECT * FROM users WHERE email = ? AND status=?');  
$stmt->execute([$email, $status]);  
$user = $stmt->fetch();  
// or  
$stmt = $pdo->  
>prepare('SELECT * FROM users WHERE email = :email AND status=:status');  
$stmt->execute(['email' => $email, 'status' => $status]);  
$user = $stmt->fetch();
```

När vi binder med `execute` kommer alla värden att bindas som strängar.

SQL injections

```
/* Execute a prepared statement by binding PHP variables */
$calories = 150;
$colour = 'red';
$stmt = $dbh->prepare('SELECT name, colour, calories
    FROM fruit
    WHERE calories < :calories AND colour = :colour');
$stmt->bindValue(':calories', $calories, PDO::PARAM_INT);
$stmt->bindValue(':colour', $colour, PDO::PARAM_STR);
$stmt->execute();
```

Vi kan välja vilken datatyp vi vill binda mot genom att använda `bindValue()`.

Fetch

```
$row = $stmt->fetch(PDO::FETCH_ASSOC);
```

- `PDO::FETCH_NUM` returns enumerated array
- `PDO::FETCH_ASSOC` returns associative array
- `PDO::FETCH_BOTH` - both of the above
- `PDO::FETCH_OBJ` returns object
- `PDO::FETCH_LAZY` allows all three (numeric associative and object) methods without memory overhead.

IN-frågor

```
$arr = [1,2,3];  
$in  = str_repeat('?', count($arr) - 1) . '?';  
$sql = "SELECT * FROM table WHERE column IN ($in)";  
$stm = $db->prepare($sql);  
$stm->execute($arr);  
$data = $stm->fetchAll();
```

```
$arr = [1,2,3];  
$in  = str_repeat('?', count($arr) - 1) . '?';  
$sql = "SELECT * FROM table WHERE foo=? AND column IN ($in) AND bar=? AND baz=?";  
$stm = $db->prepare($sql);  
$params = array_merge([$foo], $arr, [$bar, $baz]);  
$stm->execute($params);  
$data = $stm->fetchAll();
```

Sortera

```
$orders = ["name","price","qty"]; //field names
$key    = array_search($_GET['sort'],$orders); // see if we have such a name
$orderby = $orders[$key]; //
if not, first one will be set automatically. smart enuf :)
$query  = "SELECT * FROM `table` ORDER BY $orderby"; //value is safe
```

array_search

(PHP 4 >= 4.0.5, PHP 5, PHP 7)

array_search — Searches the array for a given value and returns the first corresponding key if successful

Description

```
array_search ( mixed $needle , array $haystack [, bool $strict = FALSE ] ) : mixed
```

Searches **haystack** for **needle**.

Return Values

Returns the key for **needle** if it is found in the array, **FALSE** otherwise.

Kombinera

```
$sql = "SELECT productLine, productCode, productName FROM products ";
```

```
$orders = ["productName", "productCode", "qty"]; //field names  
$key = array_search($_GET['sort'] ?? null, $orders); // see if we have such a  
name  
$orderby = $orders[$key]; //if not, first one will be set automatically. smart  
enuf :)
```

```
if (isset($_GET['productLine'])) {  
    $sql .= " WHERE productLine = '" . filter_input(INPUT_GET, 'productLine',  
FILTER_SANITIZE_STRING) . "' ";  
}
```

```
$sql .= " ORDER BY $orderby ";
```

Stored Procedure

```
$stmt = $pdo->prepare("CALL foo()");  
$stmt->execute();  
do {  
    $data = $stmt->fetchAll();  
    var_dump($data);  
} while ($stmt->nextRowset() && $stmt->columnCount());
```

Password

- Sedan ett tag tillbaka har PHP ett par nya lösenordsfunktioner.
- `password_hash()` – hashar lösenordet
- `password_verify()` – verifierar ett lösenord mot dess hash
- `password_needs_rehash()` – används för omhashning
- `password_get_info()` - ger information om hashningen

password_hash(string \$password , int \$algo [, array \$options])

- PASSWORD_DEFAULT
- PASSWORD_BCRYPT

```
echo password_hash("rasmuslerdorf", PASSWORD_DEFAULT);
```

```
$2y$10$.vGA109wmRjrwAVXD98HNOgsNpDczlqm3Jq7KnEd1rVAGv3Fykk1a
```

password_verify (string \$password , string \$hash)

```
$hash = '$2y$07$BCryptRequires22Chrcte/VlQH0piJtjXl.0t1XkA8pw9dMXTpOq';  
  
if (password_verify('rasmuslerdorf', $hash)) {  
    echo 'Password is valid!';  
} else {  
    echo 'Invalid password.';  
}
```

Varför inte spara i klartext?

- Vad händer om någon kommer åt databasen?
 - Användaren kan läsa ditt lösenord.
 - Många användare använder dessutom samma lösenord på flera ställen.

Retro

- Gör nån slags avstämning i grupperna så ni vet hur ni ligger till, vad ni behöver hjälp med osv.
- Gör en retro

Sammanfattning

- SQL injections
- Lösenord
- Livekoda inloggning

Utvärdering

- Prata i grupper om 2-3 personer i två minuter.
- Vad har varit bra idag?
- Vad skulle kunna förbättras?

Tack för idag!

Mikael Olsson
mikael.olsson@emmio.se
076-174 90 43

