

Drupal API

Témata

- Práce s databází
- Bezpečnost práce s Drupalem
- Forms API
- Jak udělat vlastní modul
- Hooks
- Lokalizace

Práce s databází

- Drupal poskytuje funkce pro práci s databází
- Výhoda:
 - Nezávislé na typu databáze
 - Při správném použití se stará o bezpečnost
- Tyto funkce zapouzdřují standardní funkce v PHP

Hlavní funkce

- Provede SQL dotaz a vrátí výsledek do proměnné \$result:

```
$result = db_query( 'dotaz' );
```

- Vrátí jednu řádku z výsledku jako objekt

```
$row = db_fetch_object($result);
```

- Vrátí jednu řádku z výsledku jako pole

```
$row = db_fetch_array($result)
```

Příklad

```
$result = db_query("SELECT n.title FROM {node} n");

while ($row = db_fetch_object($result)) {
    print 'Clanek: ' . check_plain($row->title);
}
```

Bezpečnost

- Bezpečnost webových stránek je důležitá
- Tisíce napadených webových stránek denně to dokazuje
- Nejčastější problémy:
 - SQL Injection
 - Cross Site Scripting
 - Cross Site Request Forgery

SQL Injection

- Vložení škodlivého kódu přímo do SQL dotazu
- Škodlivý kód se poté přímo provede

- Příklad:

```
$nid = 5;  
db_query("DELETE FROM {node} n WHERE n.nid = '$nid'")  
  
// vysledek  
DELETE FROM node WHERE node.nid = '5'
```

SQL Injection

- Vložení škodlivého kódu přímo do SQL dotazu
- Škodlivý kód se poté přímo provede

- Příklad:

```
$nid = "1' OR 1=1 --";  
db_query("DELETE FROM {node} n WHERE n.nid = '$nid'")
```

```
// vysledek
```

```
DELETE FROM node WHERE node.nid = '1' OR 1=1 --'
```


Při správném použití Drupal API SQL Injection **nehrozí**

Jak se bránit?

```
db_query("DELETE FROM node WHERE nid = '%d'", $nid);
```

- * Uvedený příklad neodpovídá Drupal standardům. Víte z jakého důvodu? (Pravý důvod je, že jsem potřeboval dotaz, co se vejde na řádku.)

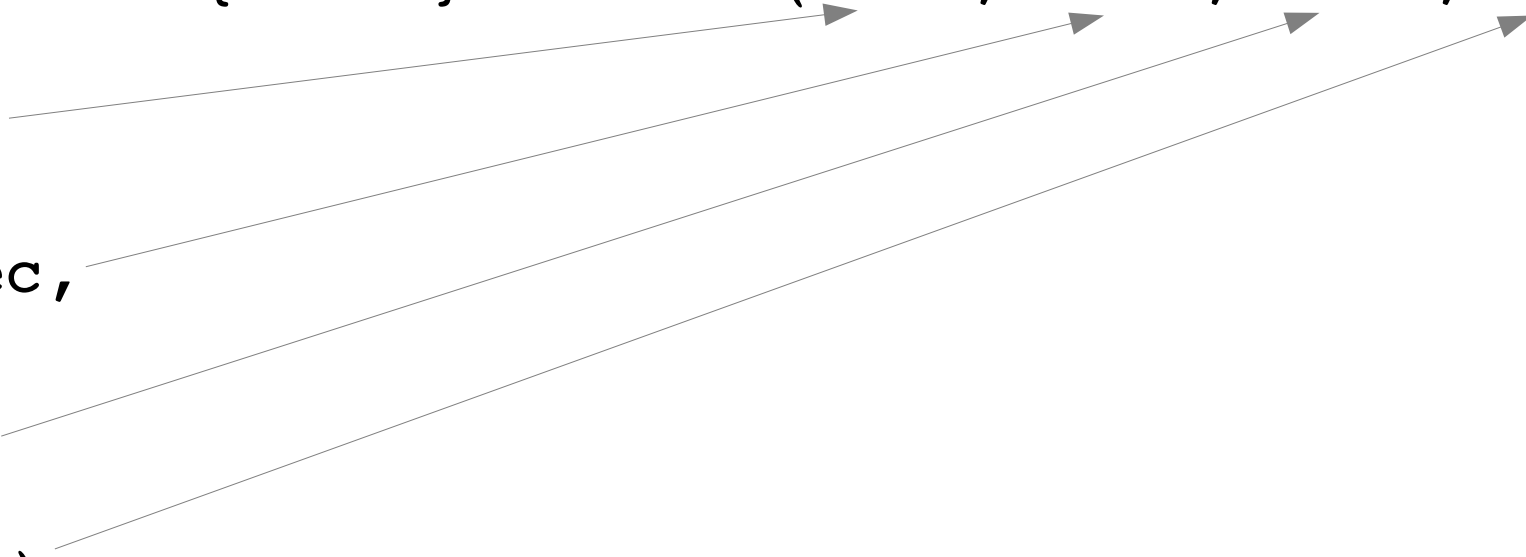
Jak se bránit?

```
db_query("
  INSERT INTO {table} VALUES ('%d', '%s', '%f', %b)",
  $cislo,
  $retezec,
  $float,
  $binary);
```

- Všimněte si pořadí tokenů, jejich uvozovek a tomu, čemu odpovídají

Jak se bránit?

```
db_query("
  INSERT INTO {table} VALUES ('%d', '%s', '%f', %b)",
  $cislo,
  $retezec,
  $float,
  $binary);
```



The diagram consists of four grey arrows pointing from the variable names in the code to their corresponding placeholders in the SQL query. The first arrow points from `$cislo,` to the `%d` placeholder. The second arrow points from `$retezec,` to the `%s` placeholder. The third arrow points from `$float,` to the `%f` placeholder. The fourth arrow points from `$binary);` to the `%b` placeholder. The placeholders `%d`, `%s`, `%f`, and `%b` are highlighted in red in the original image.

- Všimněte si pořadí tokenů, jejich uvozovek a tomu, čemu odpovídají

Cross Site Scripting

- Vložení škodlivého kódu do vstupu pro uživatele
- Také nazýváno XSS

- Příklad:

```
<input type="text" name="jmeno" />
<?php print "Vase jmeno je: ".$_GET['jmeno']; ?>
```

- Výsledek

```
$jmeno="Jenda";
Vase jmeno je: Jenda
```

```
$jmeno="<b>Jenda</b>";
Vase jmeno je: Jenda
```

bold!



Cross Site Scripting

- Co když do HTML kódu vložím např. javascript?
- Ten má přístup k mnoha informacím, např. cookies uživatele.
- Možné důsledky:
 - Ukradení přihlášení a tím celého webu
 - Přesměrování uživatele na jiný web (konkurence!)
 - Otravování ostatních uživatelů
 - A mnoho dalších

XSS je začátečnická chyba

Jak se bránit?

- Drupal API :-)
- `check_plain()` je Váš kamarád
- `check_markup()` je Váš druhý kamarád
- **Každý** výstup od uživatele **musí** jít přes jednu z těchto funkcí

```
<?php  
print "Vase jmeno je: ".check_plain($_GET['jmeno']);  
?>
```

```
$jmeno="<b>Jenda</b>";  
Vase jmeno je: <b>Jenda</b>
```


Cross Site Request Forgery

- Příklad:

Váš web používá GET formuláře nebo POST formuláře nezabezpečené proti CSRF např. pro smazání zápisu v databázi...

`http://www.example.com/smaz-zapis.php?id_zapisu=990`

- Podaří se mi někam vložit tento odkaz a donutím administrátora, který je zároveň přihlášen na svůj web na něj kliknout.

Co se stane?

Cross Site Request Forgery

- (Ano, záznam z databáze zmizí)
- Funguje jak pro GET, tak pro POST!

- Jak se bránit?

Forms API

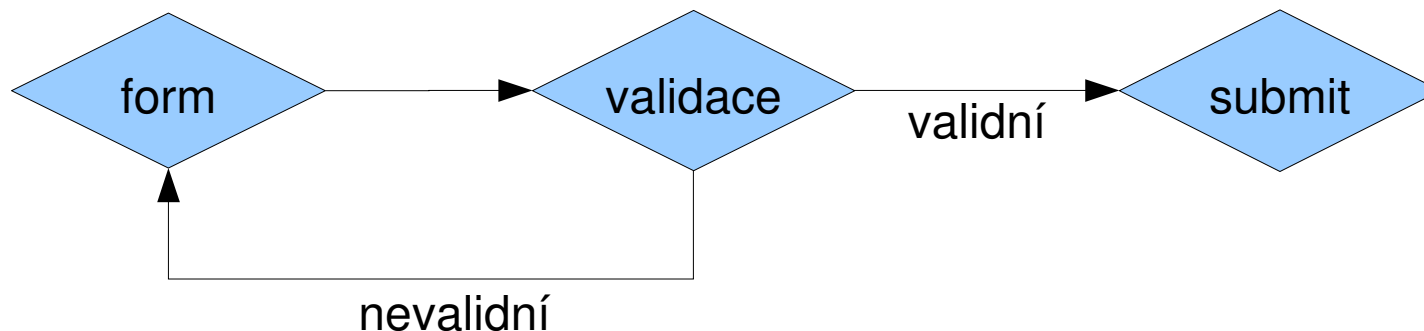
- Forms API je systém pro tvorbu webových formulářů
- Dostupné od Drupalu 4.7
- Cíl:
 - Poskytnout standardizovaný způsob tvorby formulářů
 - Ošetřit nebezpečí spojená s formuláři

Forms API z hlediska programátora

- Vytvoří asociativní pole, které definuje formulář
- Zapouzdří ho do funkce `jmenoformulare_form()`
- Vytvoří funkce `jmenoformulare_form_validate()`
a `jmenoformulare_form_submit()`
- Zavolá `drupal_get_form('jmenoformulare_form')`

Forms API z hlediska Drupalu

- Zařídí vygenerování HTML kódu formuláře
- `jmenoformulare_form_validate()`
po stisknutí Submit tlačítka
- `jmenoformulare_form_submit()`
po stisknutí Submit tlačítka a úspěšné validaci



Příklad

```
function demo_form(&$form_state) {  
  $form['nid'] = array(  
    '#title' => 'Titulek',  
    '#type' => 'textfield',  
    '#required' => TRUE,  
  );  
  $form['submit'] = array(  
    '#type' => 'submit',  
    '#value' => 'Odeslat',  
  );  
  return $form;  
}
```

Příklad

```
function demo_form_validate($form, &$form_state) {  
  if (strlen($form_state['values']['nid']) < 3) {  
    form_set_error('nid', t('Enter 3 chars!'));  
  }  
}
```

Anglicky!

```
function demo_form_submit($form, &$form_state) {  
  // Delej NECO  
}
```

Možnosti Forms API

- Generuje formuláře
- Typy políček: textfield, textarea, checkbox, radio, submit, file password, atd.
- Automaticky je zabezpečuje proti CSRF pomocí tokenu
- Automaticky zabezpečuje možnosti checkbox a radio
- Umožňuje autodoplňování
- Umožňuje plné šablonování formuláře
- Ovládání povinných a nepovinných položek
- A mnoho dalšího

Jak udělat vlastní modul

- Moduly jsou prostředky, jak do Drupalu doplňovat funkce bez nutnosti úpravy kódu samotného jádra
- To, co dělá Drupal silnou zbraní
- 90 % všech požadavků lze dnes zajistit pomocí již existujících modulů
- Zvláště pak CCK + Views
- Co když to nejde jinak?
- Buďte si jisti! Radši se zeptejte, než začnete vynalézat kolo

Základy

- Vytvořte adresář v sites/all/modules/jmenomodulu (pro nás demo)

- V něm vytvořte soubor demo.info

```
name = "Jmeno modulu"
```

```
description = "Popis modulu delsi text."
```

```
core = 6.x
```

Kompatibilita jádra



- Vytvořte soubor demo.module
- demo.module je Váš hlavní soubor

Základy

- Do demo.module patří všechny Vaše funkce
- Drupal se stará o volání těch potřebných
- Best practices: oddělovat funkční celky do více souborů

Základní stránka a její zobrazení

- Funkce `demo_menu`, vytvoří stránku s URL `demo/stranka`
- Při návštěvě Drupal zavolá funkci `demo_display`

```
function demo_menu() {  
  $items = array();  
  $items['demo/stranka'] = array(  
    'title' => 'Demo stranka',  
    'description' => 'Pouze ukazka menu',  
    'page callback' => 'demo_display',  
    'access arguments' => array('access site'),  
  );  
  return $items;  
}
```

Základní stránka a její zobrazení

```
function demo_display() {  
  $obsah = 'Ahoj svete!';  
  return $obsah;  
}
```



Základní stránka a její zobrazení

- Vytvoří stránku demo/testformulare
- Při návštěvě Drupal zavolá funkci

```
drupal_get_form('demo_neco_form')
```

Forms API

```
function demo_menu() {  
  $items = array();  
  $items['demo/testformulare'] = array(  
    'title' => 'Poptavka naseho reseni',  
    'page callback' => 'drupal_get_form',  
    'page arguments' => array('demo_neco_form'),  
  );  
  return $items;  
}
```

Základní stránka a její zobrazení

- Vytvoří stránku demo/testformulare
- Při návštěvě Drupal zavolá funkci

```
drupal_get_form('demo_neco_form')
```

Forms API

```
function demo_menu() {  
  $items = array();  
  $items['demo/testformulare'] = array(  
    'title' => 'Poptavka naseho reseni',  
    'page callback' => 'drupal_get_form',  
    'page arguments' => array('demo_neco_form'),  
  );  
  return $items;  
}
```

Hooks

- Hooks (háky) jsou prostředkem, jak zasahovat do dění v Drupal jádru bez nutnosti měnit stávající kód
- Hook se věší na určitou událost v Drupalu
- Když se událost stane, příslušný modul zavolá všechny funkce, které implementují daný hook
- Hook se vždy jmenuje `hook_jmenohooku()` kde hook se zamění za jméno modulu

Příklad

- Pamatujete na `demo_menu()` ?
- `demo_menu()` je implementací `hook_menu()`
- Když Drupal zjišťuje, jaké URL (stránky) v instalaci existují, zavolá všechny funkce, které implementují `hook_menu()`
- Hook spouští Drupal, ne programátor

Příklad

- Drupal ukládá uzel a zavolá všechny funkce, které se jmenují `jmenomodulu_nodeapi()`

Nejznámější hooky

- `hook_block` – Vytváří, zobrazuje bloky
- `hook_comment` – Spouští se při vytvoření, úpravě, smazání, zobrazení, apod. komentáře
- `hook_form_alter` – Velmi užitečný, mění formulář vygenerovaný pomocí Forms API
- `hook_menu` – Pro definici stránek
- `hook_taxonomy` – Spustí se při vytvoření, úpravě, ... slovníku či termínu taxonomy
- `hook_user` – Při vytvoření, úpravě, ... uživatele
- `hook_perm` – Definuje práva přístupu

Další hooky

<http://api.drupal.org/api/group/hooks>

Lokalizace

- Drupal může mít lokalizované rozhraní
- Jak se k tomu postavit jako vývojář modulu?

- Funkce `t()` slouží jako zapouzdření všech vypisovaných textů
- Napíšu: `print t('Node');`
- Výsledek: Slovo přeložené do aktuálního jazyka

- Drupal překládá Anglicky -> Jazyk
- Proto je nutné psát texty v angličtině

Schema API

- Drupal 6 představil Schema API
- Obecné rozhraní pro deklaraci databázových schémat
- Asociativní pole
- Zlepšuje kompatibilitu mezi databázemi

- <http://api.drupal.org/?q=api/group/schemaapi/HEAD>

Batch API

- Drupal 6 představil Batch API
 - API pro spouštění časově náročných úloh
 - Typický příklad: import českého překladu
 - Programátor rozdělí úlohu na libovolný počet iterací
 - Batch API zajistí spuštění všech iterací po sobě
 - Řeší problémy s nedostatkem paměti či času v PHP
-
- Pamatujete na bílé obrazovky při importu lokalizace v Drupalu 5 na svých low-cost webhosting účtech?

XML-RPC API

- Vzdálené spouštění definovaných Drupal funkcí
- XML-RPC je obecné API, Drupal ho pouze podporuje
- Příklad: Grafický klient pro blogování do Drupalu

Další API

- Cache API
- AJAX API
- AHAH API

Jak pokračovat

- <http://api.drupal.org>
- <http://drupal.org/node/326>

KONEC

Máte dotazy?

Jakub Suchý

e-mail: info@jsuchy.cz

<http://www.drupal.cz>