PHP Functions

by George Girtsou

ΠΝΕΥΜΑΤΙΚΑ ΔΙΚΑΩΜΑΤΑ

Στην Ελλάδα θεμελιώδης είναι ο νόμος 2121/1993 (ΦΕΚ 25, 4/3/93) ο οποίος, εμπνεόμενος από τον Παγκόσμιο Οργανισμό Διανοητικής Ιδιοκτησίας, αναθεωρεί την προηγούμενη νομοθεσία του 1920 και συμμορφώνεται με τις οδηγίες της Ευρωπαϊκής Κοινότητας.

Ν. 3184/2003, Κύρωση της Συνθήκης του Παγκόσμιου Οργανισμού Διανοητικής Εργασίας για την Πνευματική Ιδιοκτησία.

ΕΙΣΑΓΩΓΗ

Η PHP υποστηρίζει ένα πλήθος προκαθορισμένων συναρτήσεων (functions) που μας βοηθάει να κάνουμε διάφορες εργασίες ευκολότερα χωρίς να χρειάζεται να γράφουμε πολύ κώδικα.

Σε αυτή την παρουσίαση θα προσπαθήσουμε να αναλύσουμε όσο το δυνατόν περισσότερο τα πιο σημαντικά functions και θα προσπαθήσουμε να απαντήσουμε σε ερωτήσεις σας.

ΣΧΕΤΙΚΑ ΜΕ ΤΟΝ ΣΥΓΓΡΑΦΕΑ

Ονομάζομαι Γεώργιος Γκίρτσου, είμαι προγραμματιστής και γεννήθηκα στην Ελλάδα. Γνωρίζω (X)HTML, CSS, Javascript, PHP, SQL, XML & AJAX. Μ' αρέσει ο προγραμματισμός και ασχολούμαι αρκετά χρόνια με την τεχνολογία των υπολογιστών, με τα δίκτυα και με την ασφάλεια ιστοσελίδων, server και γενικά με την ασφάλεια των Η/Υ. Χαίρομαι όταν οι άνθρωποι μου κάνουν διάφορες ερωτήσεις με σκοπό να πάρουν μια απάντηση που θα τους βοηθήσει να γίνουν καλύτεροι στον προγραμματισμό μιας και κάνουν τα πρώτα τους βήματα.

ΕΡΩΤΗΣΕΙΣ / ΣΧΟΛΙΑ

Θα μ' ενδιέφερε πάρα πολύ να ακούσω τα σχόλιά σας και να απαντήσω στις ερωτήσεις σας. Μπορείτε να επικοινωνήσετε μαζί μου στο george@4net.gr ή να επισκεφθείτε την ιστοσελίδα μου: http://pctech.4net.gr όπου μπορείτε να βρείτε πολύ χρήσιμο υλικό και πολλά tutorials για τον προγραμματισμό.

trim()

trim() – Αφαιρεί τα κενά (whitespaces) από την αρχή και το τέλος ενός string.

Υποστηρίζεται

Στην PHP4 & PHP5.

Παράδειγμα

```
<?php
$text = 'this  is      my text';
echo $text;
?>
```

Το παραπάνω θα εμφανίσει: "this is my text". Αν βάλουμε echo trim($text) τότε το αποτέλεσμα θα είναι: "this is my text"

Παράδειγμα με χρήση του trim()

```
<?php
$text = 'this  is      my text';
echo trim($text);
?>
```

addslashes()

addslashes() – Προσθέτει "ανάποδους" χαρακτήρες (\) σε ένα string για να αποφύγουμε π.χ. SQL Injections.

Υποστηρίζεται

Στην PHP4 & PHP5

Παράδειγμα

```
<?php
$str = "Σε λένε O'reilly?";

// Εμφανίζει: Σε λένε O\'reilly?
echo addslashes($str);
?>
```

Επεξήγηση

Μερικές φορές θέλουμε να φιλτράρουμε (filter) κάποια δεδομένα. Όταν πρόκειται να εισάγουμε μια καινούργια εγγραφή στη Βάση Δεδομένων και το query μας είναι αυτό:

```
SELECT fieldlist
  FROM table
 WHERE field = '$EMAIL';
```

Εδώ, το $EMAIL είναι η διεύθυνση που έχει υποβληθεί στην φόρμα από τον χρήστη και τα μονά εισαγωγικά ορίζουν το $EMAIL ότι είναι ένα string, με το οποίο θα κάνει την αναζήτηση στην Βάση Δεδομένων. Δεν γνωρίζουμε τα ονόματα τον πεδίων στο SQL query, αλλά γνωρίζουμε γιατί χρησιμοποιούνται.

Όταν γράφουμε steve@unixwiz.net' – προσέξτε το ' στο τέλος θα μας δώσει αυτό το query:

```
SELECT fieldlist
  FROM table
```

```
 WHERE field = 'steve@unixwiz.net'';
```

Όταν αυτό εκτελεστεί, και SQL Parser βρει εκείνο το extra μονό εισαγωγικό και θα επιστρέψει συντακτικό σφάλμα. Αυτό δεν μας δίνει κάποιο αποτέλεσμα αλλά δείχνει στο χρήστη ότι η εφαρμογή μας είναι ευάλωτη και έχει κενά ασφαλείας.

Από τη στιγμή που το $EMAIL πάει στο WHERE clause, ας αλλάξουμε λίγο την διεύθυνση email που θα δώσουμε στην φόρμα και ας δούμε τι θα συμβεί αν δώσουμε αυτό: *anything' OR 'x'='x*

Το query που θα έχουμε είναι:

```
SELECT fieldlist
  FROM table
 WHERE field = 'anything' OR 'x'='x';
```

Αφού η εφαρμογή δεν νοιάζεται και πολύ για το query, και πιστεύει ότι το $EMAIL είναι ένα απλό string, έχουμε κενό ασφαλείας. Το παραπάνω query θα κοιτάξει για 'anything' ή 'x' = 'x' που σημαίνει ότι μπορεί να ψάξει ολόκληρο τον πίνακα της βάσης δεδομένων με αυτό το Query.

Από εδώ καταλαβαίνουμε πόσο σημαντικό είναι να φιλτράρουμε τα δεδομένα μας, όταν αυτά έρχονται από URL, φόρμες, cookies, sessions ή από άλλες εξωτερικές πηγές που δεν εμπιστευόμαστε.

ΠΡΟΣΟΧΗ

Το addslashes() κάνει ακριβώς το ίδιο με το mysql_real_escape_string() function, μόνο που όταν είναι να φιλτράρουμε δεδομένα και πρόκειται για SQL query, είναι προτιμότερο να χρησιμοποιούμε το mysql_real_escape_string() παρά το addslashes() διότι το addslashes() δεν λειτουργεί πάντα. Το addslashes() μπορούμε να το χρησιμοποιούμε όπου θέλουμε στο site για data filtering (φιλτράρισμα δεδομένων) αλλά σας συνιστώ να χρησιμοποιήσετε το mysql_real_escape_string().

stripslashes()

stripslashes() – Αφαιρεί τους "ανάποδους" χαρακτήρες (\) από ένα string. Κάνει την αντίθετη δουλειά του addslashes().

Υποστηρίζεται

Στην PHP4 & PHP5

Παράδειγμα

```
<?php
$str = "Σε λένε O\'reilly?";

// Εμφανίζει: Σε λένε O'reilly?
echo stripslashes($str);
?>
```

Επεξήγηση

Το stripslashes() είναι ένα πολύ χρήσιμο function, όταν π.χ. "διαβάζουμε" κάποια δεδομένα από ένα table της Βάσης Δεδομένων μας, και αυτά τα δεδομένα έχουν "ανάποδους" χαρακτήρες (\) για λόγους ασφαλείας (δηλ. έχουμε εφαρμόσει το mysql_real_escape_string() επάνω τους). Εμείς όμως δεν θέλουμε να εμφανίσουμε στο site μας ένα κείμενο με ανάποδους χαρακτήρες γιατί είναι αντιαισθητικό.

md5()

md5() – Κωδικοποιεί ένα string χρησιμοποιώντας τον αλγόριθμο κωδικοποίησης (encryption algorithm) md5.

Υποστηρίζεται

Στην PHP 4, PHP 5, PECL hash:1.1-1.3

Παράδειγμα

```
<?php
$str = 'apple';

// Εμφανίζει την κωδικοποιημένη μορφή του $str.
echo md5($str);
?>
```

Επεξήγηση

Όταν θέλουμε να αποθηκεύσουμε ευαίσθητα δεδομένα σε έναν πίνακα της Βάσης Δεδομένων μας, ψάχνουμε να βρούμε την καλύτερη λύση ώστε να είναι πιο ασφαλή. Ένας πολύ καλός τρόπος για να μείνουν ασφαλή αυτά τα δεδομένα, είναι να τα κωδικοποιήσουμε. Το md5 είναι πολύ ισχυρός αλγόριθμος και δεν είναι δυνατόν να τον "σπάσει" κάποιος εκτός αν ξέρει πολύ καλά μαθηματικά και μπορεί να αντέξει 2-3 χρόνια χωρίς ύπνο, φαϊ και ξεκούραση, γιατί αυτό απαιτεί το md5 για να "σπάσει".

Ένα παράδειγμα "ευαίσθητων δεδομένων" είναι οι κωδικοί πρόσβασης των χρηστών του site μας. Δεν θέλουμε κάποιος να κλέψει τους κωδικούς τους, ε; Ακόμη και αν το καταφέρει, δεν θα τον αφήσουμε να δει τους κωδικούς σε plain text (απλή μορφή κειμένου), έτσι;

Στην ουσία, όταν χρησιμοποιούμε το md5() function για να κωδικοποιήσουμε ένα string, παίρνουμε το hash. Το hash είναι ένας δεκαεξαδικός αριθμός 32 χαρακτήρων που δεν έχει καμία σχέση με τον πραγματικό κωδικό. Και να έχεις το hash, ποτέ δεν θα μπορέσεις να βρεις τον πραγματικό κωδικό. Αυτό είναι που κάνει το md5 μη-

ανατρέψιμο και τόσο δυνατό.

Έλεγχος ενός hash

Ας πούμε ότι κωδικοποιήσαμε μια λέξη 5 χαρακτήρων: admin και πήραμε το hash: 21232f297a57a5a743894a0e4a801fc3. Με το παρακάτω script θα κοιτάξουμε αν το string που θα δώσουμε δίνει ένα hash ίδιο με το παραπάνω.

```
<?php
if (md5('admin') == '21232f297a57a5a743894a0e4a801fc3') {
     echo 'Σωστός κωδικός!';
} else {
     echo 'Λάθος κωδικός!';
}
?>
```

Το παραπάνω θα επιστρέψει το μήνυμα "Σωστός κωδικός" γιατί βάλαμε md5('admin') ενώ αν βάζαμε md5('george') το hash θα έπρεπε να είναι διαφορετικό.

Αν θέλουμε να ελέγξουμε τον κωδικό που έδωσε ένας χρήστης με το hash που έχουμε αποθηκευμένο στο mySQL table μας, πρέπει να συνδεθούμε στον mySQL Server μας, μετά στην Βάση Δεδομένων και να επιλέξουμε το table και μετά να έχουμε ένα query που θα ζητάει από την mySQL να επιστρέψει το αποθηκευμένο hash (π.χ. που δημιουργήθηκε κατά την εγγραφή του χρήστη στο site).

empty()

empty() – Ελέγχει αν μια μεταβλητή είναι κενή.

Υποστηρίζεται

Στην PHP4 & PHP5

Παράδειγμα

```
<?php
$my_var = 0;
if (empty($my_var)) {
     echo 'Είτε το $my_var είναι 0, ή δεν έχει οριστεί.';
}
else {
     echo 'Το $my_var δεν είναι κενό.';
}
?>
```

Το παραπάνω παράδειγμα θα μας ενημερώσει ότι το $my_var είναι κενό.

Επεξήγηση

Το empty() είναι πολύ χρήσιμο, ειδικά όταν θέλουμε να ελέγξουμε αν έχει υποβληθεί κάποιο πεδίο σε μια φόρμα. Δηλαδή, σε μια φόρμα εισόδου, πρέπει ο χρήστης να δώσει το username του και το password του.

Για να ελέγξουμε μια μεταβλητή $_POST, μπορούμε να κάνουμε αυτό:

```
<?php
if (empty($_POST['username'])) {
     die('Παρακαλώ δώστε το username σας.');
}
?>
```

Επίσης μπορούμε να το χρησιμοποιήσουμε και ως !empty (NOT empty, όχι κενό).

```
<?php
if (!empty($_POST['username'])) {
     echo 'Username δώθηκε.';
} else {
     die('Username δεν δώθηκε.');
}
?>
```

Στο παραπάνω παράδειγμα βλέπουμε την χρήση του empty χρησιμοποιώντας το NOT (!).

echo()

echo() – Εμφανίζει ένα ή περισσότερα strings

Υποστηρίζεται

Στην PHP4 & PHP5

Παράδειγμα

```
<?php
echo 'Γεια σας!';
?>
```

Σύνταξη

Υπάρχουν 2 τρόποι για να εμφανίσουμε ένα κείμενο χρησιμοποιώντας την echo(). Ο ένας τρόπος είναι ο παραπάνω και ο άλλος ο τρόπος είναι να βάλουμε παρενθέσεις:

```
echo('Γεια σας!');
```

Η μόνη διαφορά στον τρόπο σύνταξης είναι στις παρενθέσεις.

ΠΡΟΣΟΧΗ

Πρέπει να είστε προσεκτικοί όταν θέλετε να εμφανίσετε την τιμή μιας μεταβλητής μέσα σε ένα echo(). Φυσικά, μπορείτε να το κάνετε έτσι:

```
echo $var;
```

Αλλά πώς θα εμφανίσετε μια μεταβλητή στη μέση ενός string; Ο σωστός τρόπος είναι:

```
<?php
$my_var = 'σήμερα';
echo 'Που θα πάμε ' . $my_var . ';';
?>
```

Το παραπάνω θα εμφανίσει "Που θα πάμε σήμερα;" Όπως βλέπετε έβαλα μονό εισαγωγικό μετά το "πάμε" για να τερματίσω το string και να εκτελεστεί κάποια άλλη ενέργεια εκεί. Μετά έβαλα τελεία (.) και έγραψα το όνομα της μεταβλητής που θέλω να εμφανίσω. Μετά ξαναβάζω τελεία και βάζω πάλι μονο εισαγωγικό για να συνεχίσω να γράφω το string μου. Ύστερα το κλείνω και βάζω το ελληνικό ερωτηματικό για να τελειώσει η εντολή echo εκεί.

Όταν ξεκινάτε το echo σας με μονό εισαγωγικό πρέπει να το τελειώσετε με μονό εισαγωγικό, όχι με διπλό εισαγωγικό γιατί ο parser θα σας εμφανίσει σφάλμα σύνταξης.

ΛΑΘΟΣ:

```
echo 'hello";
echo "hello';
```

ΣΩΣΤΟ:

```
echo 'hello';
echo "hello";
```

Αν θέλετε να εμφανίσετε μια μεταβλητή μέσα σε ένα echo string (χωρίς να κλείνετε το echo βάζοντας τα κατάλληλα εισαγωγικά και βάζοντας τελεία), πρέπει να συντάξετε την echo εντολή σας με διπλά εισαγωγικά:

```
$my_var = 'hi';
echo "Η τιμή της my_var είναι $my_var";
```

Αν βάζαμε μονά εισαγωγικά τότε δεν θα μας εμφάνιζε την τιμή της $my_var, αλλά θα εμφάνιζε απλά "$my_var". Ίσως είναι λίγο περίπλοκο, αλλά αν κάνετε διάφορα πειράματα με την echo και με τις μεταβλητές θα καταλάβετε πως ακριβώς δουλεύει. Απλά προσέξτε να μην κάνετε λάθος στα εισαγωγικά (να μην ξεκινήσετε με μονά εισαγωγικά και να κλείσετε με διπλά εισαγωγικά ή και το αντίθετο).

Επίσης η echo εμφανίζει όλες τις παραμέτρους.

```
<?php
echo('hi '),(' how'),(' are'),(' you?');
?>
```

Η echo επίσης έχει μια short σύνταξη που μπορεί να εμφανίσει PHP κώδικα μέσα στον HTML πολύ απλά:

```
<html>
<head>
     <title><?=$page_title;?></title>
</head>
<body>
...
</body>
</html>
```

Αν υποθέσουμε ότι η μεταβλητή $page_title έχει οριστεί κάπου παραπάνω, μπορούμε να την εμφανίσουμε μέσα στον κώδικά μας με τον πολύ σύντομο τρόπο: <?=$page_title;?>

print()

print() – Εμφανίζει ένα string.

Υποστηρίζεται

Στην PHP4 & PHP5

Σύνταξη

Η σύνταξη είναι ακριβώς ίδια με την echo: υπάρχουν και εδώ δύο τρόποι σύνταξης.

Παράδειγμα

```
print(“test”);
print “test”;
print ('test');
print 'test';
```

Όλοι οι παραπάνω τρόποι σύνταξης είναι σωστοί. Η μόνη διαφορά είναι ότι η print() επιστρέφει πάντα 1.

strlen()

strlen – εμφανίζει το μήκος (πόσους χαρακτήρες) ενός string

Υποστηρίζεται

Στην PHP4 & PHP5

Παράδειγμα

```
<?php
$my_var = 'test';
echo strlen($my_var); // 4
?>
```

str_replace()

str_replace($search, $replace, $string) – αντικαθιστά το κείμενο που δώθηκε στο $search με αυτό στο $replace μέσα σε στο $string.

Υποστηρίζεται

Στην PHP4 & PHP5

Παράδειγμα

```
<?php
$keimeno = 'το κείμενό μου';
echo str_replace('μου', 'σου', $keimeno); // το κείμενό σου
?>
```

sha1()

sha1() - κωδικοποιεί ένα string χρησιμοποιώντας τον US Secure Hash Algorithm 1 αλγόριθμο.

Υποστηρίζεται

Στην PHP 4 >= 4.3.0, PHP 5, PECL hash:1.1-1.3

Παράδειγμα

```
<?php
$str = 'password';
// Εμφανίζει την κωδικοποιημένη μορφή του $str
echo sha1($str);
?>
```

Επεξήγηση

Το sha1() function είναι αρκετά ισχυρό και είναι και αυτό μη-ανατρέψιμο όπως και το md5(). Θα μπορούσατε να αποθηκεύσετε κωδικούς πρόσβασης σε ένα table της Βάσης Δεδομένων σας με ασφάλεια και δεν θα χρειαζόταν να ανησυχείτε.

Καλύτερο encryption θα έδινε ο συνδυασμός του md5 με το sha1.

Παράδειγμα *sha1 & md5*

```
<?php
$password = 'secret password';
echo sha1(md5($password));
?>
```