



**ΠΑΝΕΠΙΣΤΗΜΙΟ
ΙΩΑΝΝΙΝΩΝ**



**ΤΜΗΜΑ
ΠΛΗΡΟΦΟΡΙΚΗΣ &
ΤΗΛΕΠΙΚΟΙΝΩΝΙΩΝ**



Πανεπιστήμιο Ιωαννίνων
ΣΧΟΛΗ ΠΛΗΡΟΦΟΡΙΚΗΣ ΚΑΙ ΤΗΛΕΠΙΚΟΙΝΩΝΙΩΝ
ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ ΚΑΙ
ΤΗΛΕΠΙΚΟΙΝΩΝΙΩΝ

2^ο Εξάμηνο 2019 – 2020



ΕΛΛΗΝΙΚΗ ΔΗΜΟΚΡΑΤΙΑ
ΠΑΝΕΠΙΣΤΗΜΙΟ ΙΩΑΝΝΙΝΩΝ
ΣΧΟΛΗ ΠΛΗΡΟΦΟΡΙΚΗΣ & ΤΗΛΕΠΙΚΟΙΝΩΝΙΩΝ
ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ & ΤΗΛΕΠΙΚΟΙΝΩΝΙΩΝ



Πρωτόκολλα & Αρχιτεκτονικές Δικτύων





Φόρμες

Οι φόρμες αποτελούν το μέσο συλλογής πληροφοριών από τον επισκέπτη μιας εφαρμογής. Ορισμός μιας τυπικής φόρμας περιλαμβάνει τη συμπλήρωση δυο χαρακτηριστικών: της μεθόδου (method) με την οποία θα σταλούν τα δεδομένα και των ενεργειών (action) που θα πραγματοποιηθούν σε αυτά.

```
<form  
action="do_something_script.php"  
method="get">  
.....</form>
```



POST ή GET. Βασικές διαφορές

Ημέθοδος GET.

- δεδομένα προσάρτηση στο URL

test/demo_form.asp?name1=value1&name2=value2

- δεδομένα υπόκεινται σε περιορισμό μεγέθους

get έως 2048 χαρακτήρες, περίπου

- φυλλομετρητής μπορεί να βάλει σελιδοδείκτη

Εντούτοις

- Χρήσιμη στην αποσφαλμάτωση



Παράδειγμα φόρμας

```
<form action="do_something_script.php"
method="get">
<input type="text" name="name" ><br>
<input type="text" name="email" ><br>
<input type="text" name="contact"><br>
<input type="submit" name="submit"
value="Submit">
</form>
```




Παράδειγμα φόρμας - Αποτέλεσμα

← → ↻ 🏠 ⓘ file:///H:/

 Google  Yahoo Mail 

john

john@gmail.com

9877989898

Submit



URL encoding

Πριν την αποστολή των πληροφοριών, αυτές κωδικοποιούνται σύμφωνα με κάποιους κανόνες που ονομάζονται σχήμα URL (URL encoding).

- μεταβλητές και οι τιμές τους συνδέονται με το σύμβολο της ισότητας (=) σχηματίζοντας ζεύγη,
- τα διαφορετικά ζεύγη διαχωρίζονται μεταξύ τους με το σύμβολο **&**.
- Τα διαστήματα αντικαθίστανται από το σύμβολο + και
- οποιοσδήποτε μη αλφαριθμητικός χαρακτήρας αντικαθίσταται από τη **δεκαεξαδική** του τιμή.
- Τα διαφορετικά είδη πληροφοριών διαχωρίζονται από το σύμβολο **?**.

http://www.example.com/do_something_script.php?name=john&email=john@gmail.com&contact=9877989898



Συλλογή στοιχείων

```
<!DOCTYPE html>
<html>
<head>
<title>Πρώτη ιστοσελίδα με φόρμα</title>
<meta charset="UTF-8">
</head>
<body>
<form action="address.php" method="post">
<label for="name">Το όνομά μου είναι:</label> <br>
<input type="text" id="name" name="YourName" > <br><br>
<label for="mail">Το email μου είναι:</label> <br>
<input type="text" id="mail" name="YourEmail"> <br><br>
<label for="addr">Η διεύθυνσή μου είναι:</label> <br>
<input type="text" id="addr" name="YourAddr"> <br><br>
<input type="submit" name="submit" value="Send">
</form>
</body>
</html>
```



Αποτέλεσμα

Πρώτη ιστοσελίδα με φόρμα - Mozilla Firefox

File Edit View History Bookmarks Tools Help

Πρώτη ιστοσελίδα με φόρμα x +

Το όνομά μου είναι:

Το email μου είναι:

Η διεύθυνσή μου είναι:

Send

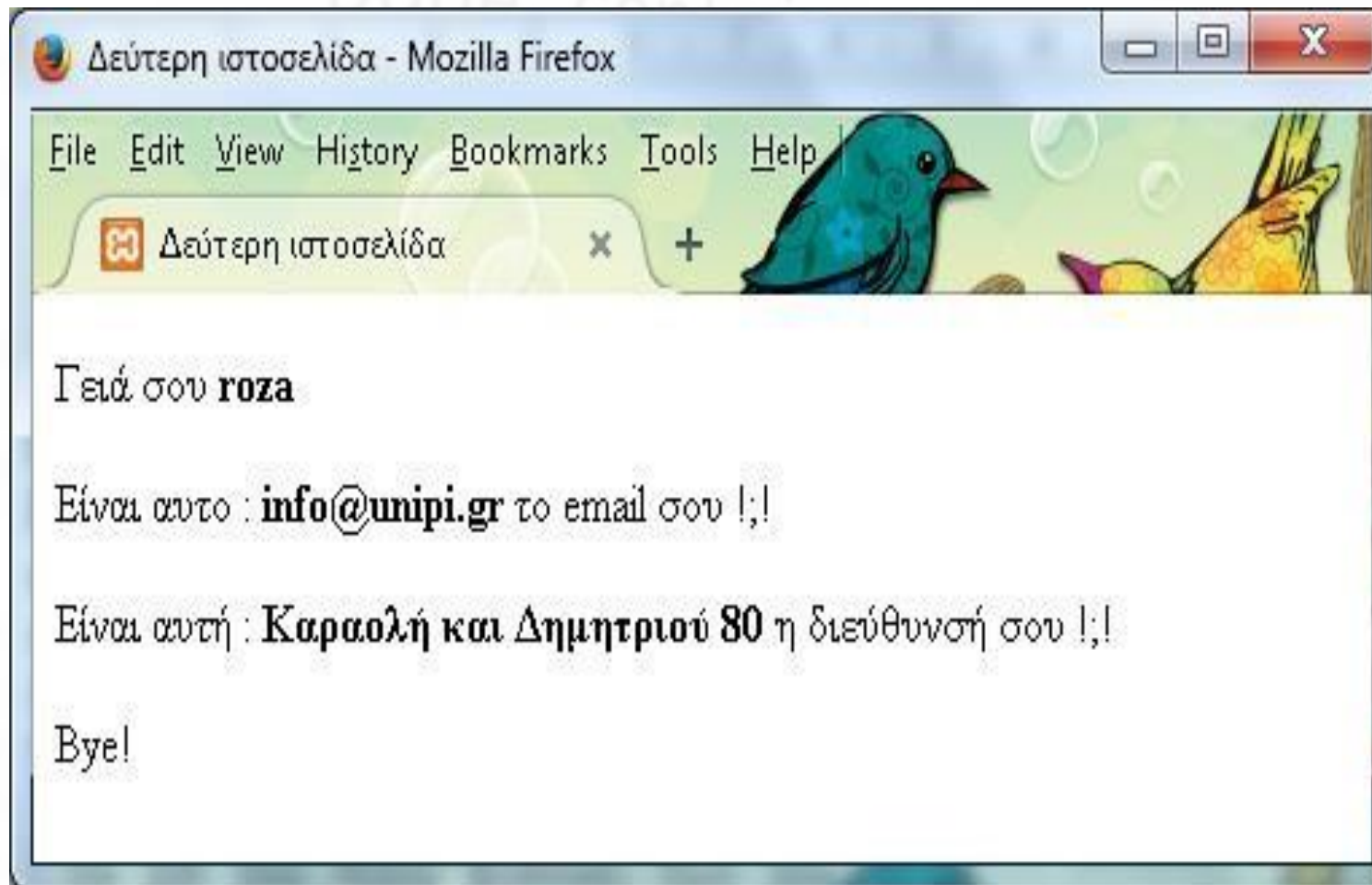


Παραλαβή στοιχείων

```
<!DOCTYPE html>
<html>
<head>
<title>Δεύτερη ιστοσελίδα</title>
<meta charset="UTF-8">
</head>
<body>
<p>Γειά σου <strong>
<?php echo $_POST['YourName'];?> </strong></p> <p>Είναι
αυτο : <strong>
<?php echo $_POST['YourEmail']; ?> </strong> το email σου
!;! </p>
<p>Είναι αυτή : <strong>
<?php echo $_POST['YourAddr']; ?>
</strong> η διεύθυνσή σου !;! </p> <p>Bye!</p>
</body> </html>
```



Παράδειγμα





Επικύρωση πεδίων

- πρέπει να παρέχονται αρκετές οδηγίες και υποδείξεις
- πρέπει να υπάρχει μια λογική σειρά στην πλοήγηση ανάμεσα στα πεδία της φόρμας
- οι χρήστες να μαθαίνουν τα λάθη τους κατά την πληκτρολόγηση
- οι φόρμες θα πρέπει να μπορούν να συμπληρωθούν και να αποσταλούν με τη χρήση μόνο του πληκτρολογίου, π.χ. στις συσκευές κινητής τηλεφωνίας.



Έλεγχος στην πλευρά του πελάτη. html

```
<form action="address.php" method="post"> <label  
for="name">Το όνομά μου είναι:</label> <br>  
<input type="text" id="name" name="YourName"  
required> <br><br>  
<label for="mail">Το email μου είναι:</label> <br>  
<input type="email" id="mail" name="YourEmail"  
placeholder="you@some.com"> <br><br>  
<label for="addr">Η διεύθυνσή μου είναι:</label>  
<br>  
<input type="text" id="addr" name="YourAddr"  
placeholder="οδός αριθμός"> <br><br>  
<input type="submit" name="submit" value="Send">  
</form>
```




Έλεγχος στην πλευρά του πελάτη. CSS

Παραδείγματα κανόνων CSS

```
input[required]{background: gray;}  
input[required]:valid{background:  
green;}  
input[type='email']:invalid{backgroun  
d: red;}
```



Έλεγχος στην πλευρά του πελάτη.

javascript

```
<script>
function CheckMe () {
  if (document.getElementById('name').value=="R") {
    alert("Γειά σου R"); }
  else {
    alert("Προπαθήστε ξανά!!")
  }
} document.getElementById('btn').onclick=
CheckMe;
</script>
```



input validation html+css

Input validation html+css



Έλεγχος στην πλευρά του εξυπηρετητή

Αρχικά, θα πρέπει να βεβαιωθεί ότι έχουν συμπληρωθεί όλα τα απαραίτητα πεδία της φόρμας. Για το σκοπό αυτό χρησιμοποιείται η συνάρτηση `empty()`. Η συνάρτηση αυτή ελέγχει εάν μια μεταβλητή είναι 'κενή', δηλαδή εάν δεν έχει οριστεί, εάν είναι `null` ή `false` ή 0 ή περιέχει κενό αλφαριθμητικό (empty string).

```
if (empty($_POST["surname"])) {  
    echo "Το Ονοματεπώνυμο δε μπορεί να  
    είναι κενό";  
}
```



Έλεγχος στην πλευρά του εξυπηρετητή

Στη συνέχεια, έστω ότι χρειάζεται να ελεγχθεί αν το πεδίο του email περιέχει μια διεύθυνση ηλεκτρονικού ταχυδρομείου με έγκυρη μορφή, δηλαδή περιέχει γράμματα, το σύμβολο @ καθώς και μια τελεία (.). Αυτό πραγματοποιείται με τη συνάρτηση `filter_var()` και εφαρμογή του κατάλληλου φίλτρου,

```
if (!filter_var($email, FILTER_VALIDATE_EMAIL)) {  
    echo "Μη αποδεκτή μορφή email";  
}
```



Άλλα φίλτρα

Φίλτρο	Περιγραφή
<code>FILTER_VALIDATE_BOOLEAN</code>	Ελέγχει εάν είναι boolean.
<code>FILTER_VALIDATE_FLOAT</code>	Ελέγχει εάν είναι float.
<code>FILTER_VALIDATE_INT</code>	Ελέγχει εάν είναι integer.
<code>FILTER_VALIDATE_IP</code>	Ελέγχει εάν είναι διεύθυνση IP.
<code>FILTER_VALIDATE_MAC</code>	Ελέγχει εάν είναι διεύθυνση MAC.
<code>FILTER_VALIDATE_REGEXP</code>	Ελέγχει εάν ικανοποιείται μια κανονικοποιημένη έκφραση, η οποία είναι συμβατή με τη γλώσσα Perl.
<code>FILTER_VALIDATE_URL</code>	Ελέγχει εάν είναι URL. Δεν ελέγχει εάν το URL περιλαμβάνει κάποιο έγκυρο πρωτόκολλο, πχ. http, https, ssh κλπ.



regular expression

Ένας διαφορετικός τρόπος για τον έλεγχο της τιμής URL γίνεται με την εφαρμογή της κατάλληλης κανονικοποιημένης έκφρασης (regular expression) στη συνάρτηση **preg_match**. Η συνάρτηση αυτή επιβεβαιώνει την ύπαρξη (ή όχι) ενός δοσμένου μοτίβου (pattern) μέσα σε ένα αλφαριθμητικό. Το μοτίβο παρουσιάζεται με όρους κανονικοποιημένων εκφράσεων. Επιστρέφει την τιμή 1 εφόσον βρεθεί ταύτιση, διαφορετικά 0 και 'false' εφόσον προκύψει κάποιο λάθος.

```
preg_match(κανονικοποιημένη έκφραση, $variable)
```



regular expression

`'rob@example.com'`

πραγματικά δεδομένα

`'/^ [a-z\d\._-]+@ ([a-z\d-]+\.)+
[a-z]{2,6}$ /i '`

Συνάρτηση PHP
πάνω στις ΚΕ

κανονικοποιημένη
έκφραση

`preg_match()`



Ορισμοί

- Περικλείονται σε '/'
π.χ. /rhr/ θα ταιριάζει με τα
'rhr', 'I love rhr' αλλά όχι και με τα
'RHR' ή 'rh p'
- Ολόκληρο το μοτίβο θα πρέπει να ταιριάζει, αλλά
όχι απαραίτητα με ολόκληρο το αλφαριθμητικό
- Πραγματοποιεί διάκριση ανάμεσα σε πεζά και
κεφαλαία



Το μόνο που μπορεί να χρησιμοποιηθεί μετά την τελική / είναι ο δείκτης i, ο οποίος πραγματοποιεί ή όχι διάκριση ανάμεσα σε πεζούς ή κεφαλαίους χαρακτήρες.

π.χ. το /rhp/i θα ταιριάζει με τα

'rhp', 'I love rhp', 'love rHp' και το 'RHP' αλλά όχι το 'rh p'



Το μοτίβο μιας κανονικοποιημένης έκφρασης ελέγχεται χαρακτήρα προς χαρακτήρα. Η χρήση των [] επιτρέπει την ομαδοποίηση χαρακτήρων.

π.χ. το /p[hu]r/ θα ταιριάζει με τα
'rhr', 'rur', αλλά όχι το 'rhur', 'ror',
'RHR'.



Εντός των αγκύλων [] μπορεί να οριστεί μια ακολουθία από χαρακτήρες ή αριθμούς π.χ. το /p[a-z0-5]p/ θα ταιριάζει με τα 'rhr', 'rur', 'rap', 'rzp', 'p0p', 'p3p', αλλά όχι τα 'rhur', 'p6p', 'RHP'.



Προκαθορισμένες τιμές

\d	Αποδίδει τα νούμερα από 0 έως 9
\s	Αποδίδει το χαρακτήρα του κενού, tab
\w	Αποδίδει όλους τους χαρακτήρες και τους αριθμούς και το _



- το `'/p\dr/'` θα ταιριάζει με
`'p3p'`, `'p7p'` αλλά όχι με τα `'p10p'`, `'P7p'`
- Το `'/p\wr/'` θα ταιριάζει με
`'p3p'`, `'pHp'`, `'por'` αλλά όχι με τα `'rhhp'`.



Η τελεία . αντικαθιστά τα πάντα εκτός την αλλαγή γραμμής.

`'/p.p/'`

ταιριάζει με `'rhp'`, `'r$p'`, `'r3p'`, `'r@p'`

αλλά όχι και με τα `'PHP'`, `'rhhp'`



- Η επανάληψη χαρακτήρων αναπαρίσταται με

?	Επαναλαμβάνει 0 ή 1 φορά
*	Επαναλαμβάνει 0 ή περισσότερες φορές
+	Επαναλαμβάνει 1 ή περισσότερες φορές
{a,b}	Επαναλαμβάνει a έως b φορές



- το $\text{'}/\text{rh?p}/$ θα ταιριάζει με
 'rp' , 'rhp' αλλά όχι με τα 'rhhp' , 'Rhp'
- Το $\text{'}/\text{rh}^*\text{p}/$ θα ταιριάζει με
 'rp' , 'rhp' , 'rhhhhhhhp' αλλά όχι με τα 'rhhohhp' .
- Το $\text{'}/\text{rh+p}/$ θα ταιριάζει με
 'rp' , 'rhp' , 'rhhhhhhhp' αλλά όχι με τα 'rp' , 'rhyhyp' .
- Το $\text{'}/\text{rh}\{1,3\}\text{p}/$ θα ταιριάζει με 'rhp' , 'rhhp' , 'rhhhp'
αλλά όχι με τα 'rp' , 'rhhhhhhhhhp' .



- Η επανάληψη μπορεί να εφαρμοστεί σε μια ομάδα χαρακτήρων.
- Το $'/(rhr)+/'$ θα ταιριάζει με $'rhr'$, $'rhrhrhrhrhr'$ αλλά όχι με τα $'rr'$, $'rhyhyp'$.



- /[^]rhp/ σημαίνει ταύτιση της ακολουθίας rhp, εφόσον αυτή βρίσκεται στην αρχή του αλφαριθμητικού
π.χ. Ενώ το /rhp/ θα ταιριάζει με τα
'rhp', 'I love rhp', 'rhp is great' αλλά όχι και με τα
'RHP' ή 'ph p'
Το /[^]rhp/ θα ταιριάζει με τα
'rhp', 'rhp is great' αλλά όχι και με τα
'RHP' ή 'I love rhp'



/rhp\$/ σημαίνει ταύτιση της ακολουθίας rhp, εφόσον αυτή βρίσκεται στο τέλος του αλφαριθμητικού

π.χ. το /rhp\$/ θα ταιριάζει με τα 'rhp', 'I love rhp'
αλλά όχι και με τα 'RHP' ή 'rhp is great'



- Το ταίριασμα χαρακτήρων με ειδική σημασία πραγματοποιείται με τη χρήση του \ (χαρακτήρας διαφυγής). Έτσι το '/p\..p/' θα ταιριάζει το 'p.p' αλλά όχι το 'php', 'p1p'

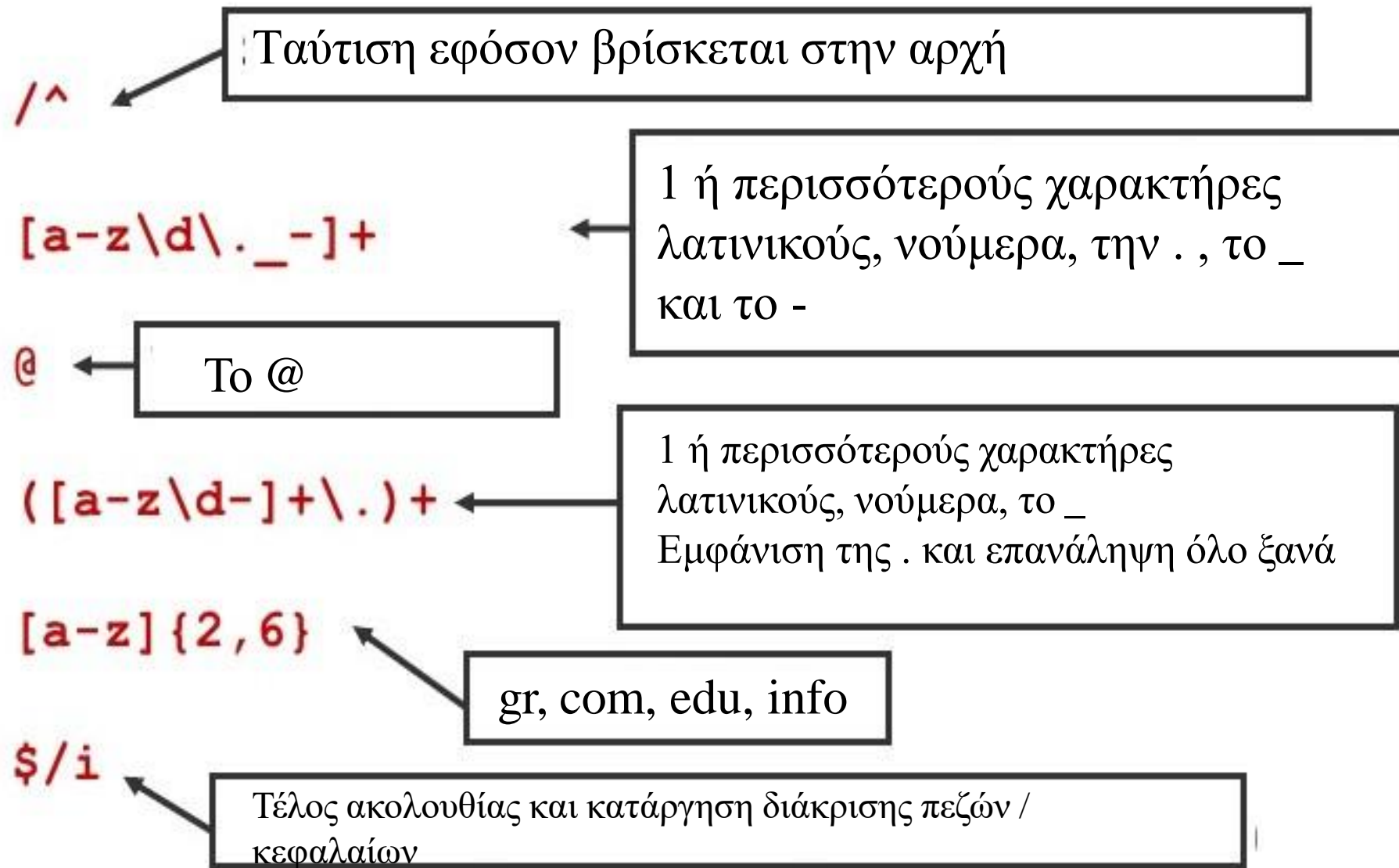


email regular expression

```
$emailRegex = '/^[a-z\d\._-]+@([a-z\d-]+\.)+[a-z]{2,6}$/i';
```

Matches: 'rob@example.com',
 'rob@subdomain.example.com'
 'a_n_o_t_h_e_r@example.co.uk'

Doesn't match: 'rob@exam@ple.com'
 'not.an.email.com'





```
$input = 'rob@example.com';  
if (preg_match($emailRegex,$input) {  
    echo 'Is a valid email';  
} else {  
    echo 'NOT a valid email';  
}
```



Παράδειγμα σε φόρμα

```
// αρχικοποίηση μεταβλητών με μηδενικές τιμές
$nameErr = $emailErr = $genderErr = $websiteErr =
""; $name = $email = $gender = $comment = $website =
"";
if ($_SERVER["REQUEST_METHOD"] == "POST") {      if
(empty($_POST["YourName"])) {
    $nameErr = "Το Ονοματεπώνυμο δε μπορεί να
είναι κενό."; }
    else {
        $name = sanit_input($_POST["YourName"]);
/*Έλεγχος ότι περιέχει μόνο γράμματα και κενά
διαστήματα */
        if (!preg_match("/^[a-zA-Z ]*$/", $name)) {
            $nameErr = "Στο Ονοματεπώνυμο
επιτρέπονται μόνο γράμματα και κενά διαστήματα."; }
        }
```




```
if (empty($_POST["YourEmail"])) {  
    $emailErr = "Το Email δε μπορεί  
να είναι κενό."; }  
else {  
    $email =  
    sanit_input($_POST["YourEmail"]);  
    // Έλεγχος έγκυρης μορφής email  
    if (!filter_var($email,  
    FILTER_VALIDATE_EMAIL)) { $emailErr =  
    "Μη αποδεκτή μορφή email."; } }
```



```
if (empty($_POST["website"])) {  
    $website = "";  
} else {  
    $website = sanit_input($_POST["website"]);  
    // Έλεγχος έγκυρης μορφής URL)  
    if  
    (!preg_match("/\b(?:?:https?|ftp):\/\/\/  
|www\.)[-a-z0-9+&@#\/%?~_!|:,.;]*[-a-  
z0-9+&@#\/%?~_!|]/i", $website)) {  
        $websiteErr = "Μη αποδεκτή μορφή URL";  
    }  
}
```



```
if (empty($_POST["YourAddr"])) {  
    $comment = "";  
} else {  
    $comment = sanit_input($_POST["YourAddr"]);  
}  
}
```



```
function sanit_input($data) { /*Πριν  
τη χρήση των δεδομένων επιβεβαίωσε  
ότι δεν περιέχουν κακόβουλο  
λογισμικό, τμήμα κώδικα ή λάθος  
χαρακτήρα ειδικού σκοπού π.χ. ; */  
return $data; }  
  
if (!empty($nameErr)) echo $nameErr;  
if (!empty($emailErr)) echo $emailErr;  
if (!empty($websiteErr)) echo $websiteErr;  
?>
```



input validation html+css

Input validation php



Διαχείριση αρχείων

Τα αρχεία είναι σημαντικά για την αποθήκευση πληροφοριών, καθώς:

- Οι πληροφορίες είναι χωρίς κάποια δομή και το μέγεθος τους είναι εύκολα διαχειρίσιμο από το σύστημα αρχείων του εξυπηρετητή.
- Οι πληροφορίες δεν μπορεί να συσχετιστούν μεταξύ τους, ώστε ο συνδυασμός τους να διεκπεραιώνει κάποιο ερώτημα.
- Το μέγεθος των αποθηκευμένων δεδομένων δεν αναμένεται να ξεπερνά τις δυνατότητες του υπολογιστικού συστήματος.
- Δεν είναι εφικτή η χρήση κάποιας βάσης δεδομένων.
- Οι πληροφορίες έχουν κάποια ειδική μορφή, όπως να είναι εικόνες, αρχεία pdf, κ.α.



Πριν την οποιαδήποτε ενέργεια σε κάποιο αρχείο θα πρέπει να ερευνηθεί εάν αυτό υπάρχει ήδη. Ο παρακάτω κώδικας χρησιμοποιεί τη συνάρτηση `file_exists` για τον έλεγχο ύπαρξης ενός αρχείου.

```
if (file_exists("testfile.txt")) echo  
"Το αρχείο υπάρχει ήδη";
```

Εφόσον το αρχείο δεν υπάρχει θα πρέπει, ίσως, να δημιουργηθεί.



Αρχεία και php

```
<?php
$file_name='testfile.txt';
$fh = fopen($file_name, 'w') or
die("Αποτυχία δημιουργίας αρχείου");
$text = <<<_END
Line 1
Line 2 Line 3
_END;
fwrite($fh, $text) or die("Δεν ήταν δυνατή
η εγγραφή στο αρχείο $file_name ");
fclose($fh); echo "Το αρχείο με όνομα
$file_name εγγράφηκε επιτυχώς";
?>
```



Παρά μετρος	Περιγραφή
r	Άνοιγμα για ανάγνωση, ξεκινώντας από την αρχή του αρχείου (πρώτος χαρακτήρας μέσα στο αρχείο). Σε περίπτωση σφάλματος, π.χ. εάν το αρχείο δεν υπάρχει, θα επιστρέψει FALSE.
r+	Άνοιγμα για ανάγνωση και εγγραφή, ξεκινώντας από την αρχή του αρχείου (πρώτος χαρακτήρας μέσα στο αρχείο). Σε περίπτωση σφάλματος, π.χ. εάν το αρχείο δεν υπάρχει, θα επιστρέψει FALSE.
w	Άνοιγμα για εγγραφή μόνο. Ο κέρσοντας θα τοποθετηθεί στην αρχή του αρχείου. Τα περιεχόμενα του αρχείου θα διαγραφούν. Εφόσον το αρχείο δεν υπάρχει θα δημιουργηθεί.
w+	Άνοιγμα για εγγραφή και ανάγνωση. Ο κέρσοντας θα τοποθετηθεί στην αρχή του αρχείου. Τα περιεχόμενα του αρχείου θα διαγραφούν. Εφόσον το αρχείο δεν υπάρχει θα δημιουργηθεί.
a	Άνοιγμα για εγγραφή μόνο. Ο κέρσοντας θα τοποθετηθεί στο τέλος του αρχείου. Εφόσον το αρχείο δεν υπάρχει θα δημιουργηθεί.
a+	Άνοιγμα για εγγραφή και ανάγνωση. Ο κέρσοντας θα τοποθετηθεί στο τέλος του αρχείου. Εφόσον το αρχείο δεν υπάρχει θα δημιουργηθεί.



Το παρακάτω παράδειγμα παρουσιάζει δύο διαφορετικούς τρόπους ανάγνωσης του περιεχομένου ενός αρχείου. Ο ένας τρόπος περιλαμβάνει τη συνάρτηση **fgets**, η οποία 'διαβάζει' τα περιεχόμενα ανά γραμμή, δηλαδή σειρά σειρά. Ο δεύτερος τρόπος περιλαμβάνει τη συνάρτηση **fread**, η οποία 'διαβάζει' τα περιεχόμενα ανά χαρακτήρα.



```
<?php
$fh = fopen("testfile.txt", 'r') or die("To
αρχείο ή δεν υπάρχει ή δεν έχετε τα
κατάλληλα δικαιώματα"); $line = fgets($fh);
//ανάγνωση γραμμής
$text = fread($fh, 3);
// ανάγνωση 3 χαρακτήρων
fclose($fh);
echo $line; //θα εκτυπώσει το: Line 1
echo $text; /*θα εκτυπώσει το: Lin , δηλαδή
τους 3 πρώτους χαρακτήρες */
?>
```



Μια συνάρτηση με την οποία πραγματοποιείται η ανάγνωση ολόκληρου του περιεχομένου ενός αρχείου είναι η **file_get_contents** η οποία δέχεται ως παράμετρο το όνομα του αρχείου ή το URL του, εάν βρίσκεται σε απομακρυσμένο εξυπηρετητή. Ο παρακάτω κώδικας αποτελεί ένα απλό παράδειγμα χρήσης της.



```
<?php
echo "<pre>";
/* ώστε να εμφανιστούν τακτοποιημένα τα περιεχόμενα
στην HTML*/
echo file_get_contents("testfile.txt"); echo
"</pre>";
/*Εμφάνιση από απομακρυσμένο
εξυπηρετητή*/
echo
file_get_contents("http://oreilly.com");
?>
```



Το παρακάτω παράδειγμα χρησιμοποιείται για την αντιγραφή, μετακίνηση και διαγραφή ενός αρχείου. Πραγματοποιεί έλεγχο της επιτυχίας της διαδικασίας (αντιγραφής ή μετακίνησης ή διαγραφής) και σε περίπτωση σφάλματος εκτυπώνει μήνυμα λάθους.



```
<?php //αντιγραφή
if (!copy('testfile.txt', 'testfile2.txt'))
echo "Δεν ήταν δυνατή η αντιγραφή"; else
echo "Επιτυχία αντιγραφής";
//μετακίνηση
if (!rename('testfile2.txt', 'testfile2.new'))
echo "Δεν ήταν δυνατή η μετακίνηση";
else echo "Επιτυχία μετακίνησης ως
μετονομασία";
//Διαγραφή
if (!unlink('testfile2.new')) echo "Δεν ήταν
δυνατή η διαγραφή";
else echo "Επιτυχία διαγραφής";
?>
```