Министерство образования республики Беларусь

Учреждение образования

«БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ИНФОРМАТИКИ И РАДИОЭЛЕКТРОНИКИ»

**Институт информационных технологий**

Специальность ПОИТ

**КОНТРОЛЬНАЯ РАБОТА**

По курсу Веб технологии

Студент-заочник 3 курса

Группы № 581072

Богданова Кристина Евгеньевна

Тел. +375 (25) 929-99-11

Приняла

преподаватель

Коренская Ирина Николаевна

Минск, 2018

**Содержание**

[**Содержание** 2](#_Toc503478143)

[**Задание № 1** «Основы вёрстки» 3](#_Toc503478144)

[Решение 3](#_Toc503478145)

[Выводы 7](#_Toc503478146)

[**Задание № 2** «Основы PHP» 8](#_Toc503478147)

[Решение 8](#_Toc503478148)

[Выводы 13](#_Toc503478149)

[**Задание № 3** «Основы разработки сайтов» 14](#_Toc503478150)

[Решение 14](#_Toc503478151)

[Выводы 17](#_Toc503478152)

[**Задание № 4** «Основы взаимодействия с БД» 18](#_Toc503478153)

[Решение 18](#_Toc503478154)

[Выводы 22](#_Toc503478155)

[**Задание № 5** «Основы работы с регулярными выражениями» 24](#_Toc503478156)

[Решение 24](#_Toc503478157)

[Выводы 27](#_Toc503478158)

[**Задание № 6** «Основы работы с лентами новостей» 28](#_Toc503478159)

[Решение 29](#_Toc503478160)

[Выводы 33](#_Toc503478161)

# **Задание № 1** «Основы вёрстки»

Цель работы: изучить теоретический материал по теме «Основы вёрстки» и закрепить его путем выполнения индивидуального задания.

Задание: Разработать HTML-страницу, соответствующую приведённой ниже картинке. Использовать фреймы и каскадные таблицы стилей.

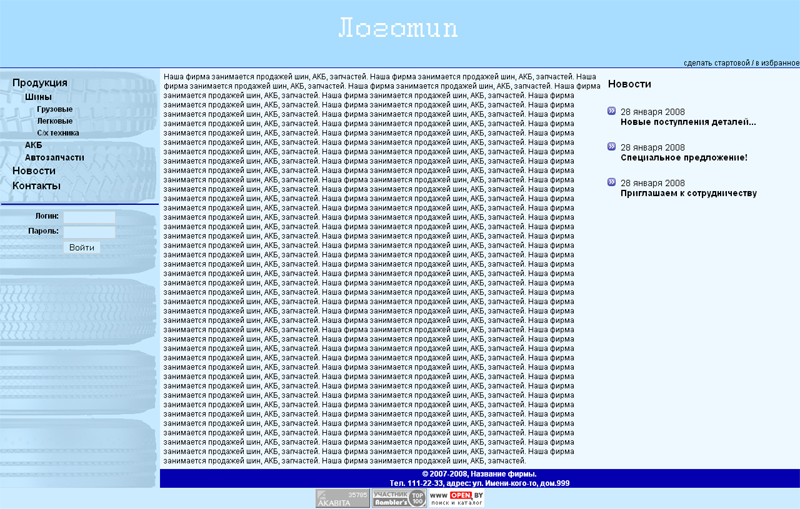


Рисунок 1.1 – Пример страницы

## Решение

Исходный код:

index.html

<!DOCTYPE html>

<html>

<head>

<meta charset="utf-8" />

<!--[if lt IE 9]><script src="https://cdnjs.cloudflare.com/ajax/libs/html5shiv/3.7.3/html5shiv.min.js"></script><![endif]-->

<title>1</title>

<meta name="keywords" content="" />

<meta name="description" content="" />

<link href="css/style.css" rel="stylesheet">

</head>

<body>

<div class="wrapper">

<header class="header">

<div><a href="#">сделать стартовой</a> / <a href="#">в избранное</a></div>

</header>

<!-- .header-->

<div class="container">

<aside class="left-sidebar">

<ul class="nav">

<li><a href="#">Продукция</a></li>

<li>

<ul>

<li><a href="#">Шины</a></li>

<li>

<ul>

<li><a href="#">Грузовые</a></li>

<li><a href="#">Легковые</a></li>

<li><a href="#">С/х техника</a></li>

</ul>

</li>

<li><a href="#">АКБ</a></li>

<li><a href="#">Автозапчасти</a></li>

</ul>

</li>

<li><a href="#">Новости</a></li>

<li><a href="#">Контакты</a></li>

</ul>

<div class="login">

<form class="" action="index.html" method="post">

<label for="login">Логин:</label>

<input id="login" type="text" />

<br />

<br />

<label for="password">Пароль:</label>

<input id="password" type="password" />

<br />

<br />

<button>Войти</button>

</form>

</div>

</aside>

<!-- .left-sidebar -->

<aside class="right-sidebar">

<h2>Новости</h2>

<div class="news">

<ul>

<li>

<p>28 января 2008</p>

<a href="#">Новые поступления деталей...</a>

</li>

<li>

<p>28 января 2008</p>

<a href="#">Специальное предложение!</a>

</li>

<li>

<p>28 января 2008</p>

<a href="#">Приглашаем к сотрудничеству</a>

</li>

</ul>

</div>

</aside>

<!-- .right-sidebar -->

<main class="content">

<frameset>

<iframe class="contentIframe" src="content.html " scrolling="no" frameborder="0" style="display: block; border: none; width: 100%; height: 640px;" name=CONTENT></iframe>

</frameset>

</main>

<!-- .content -->

<div class="contacts">

<p>&copy; 2007-2008, Название фирмы.</p>

<p>Тел. 111-22-33, адрес: ул. Имени-кого-то, дом 999</p>

</div>

</div>

<!-- .container-->

<footer class="footer">

<a href="#"><img src="images/image1.bmp" /></a>

<a href="#"><img src="images/image1.gif" /></a>

<a href="#"><img src="images/image3.bmp" /></a>

</footer>

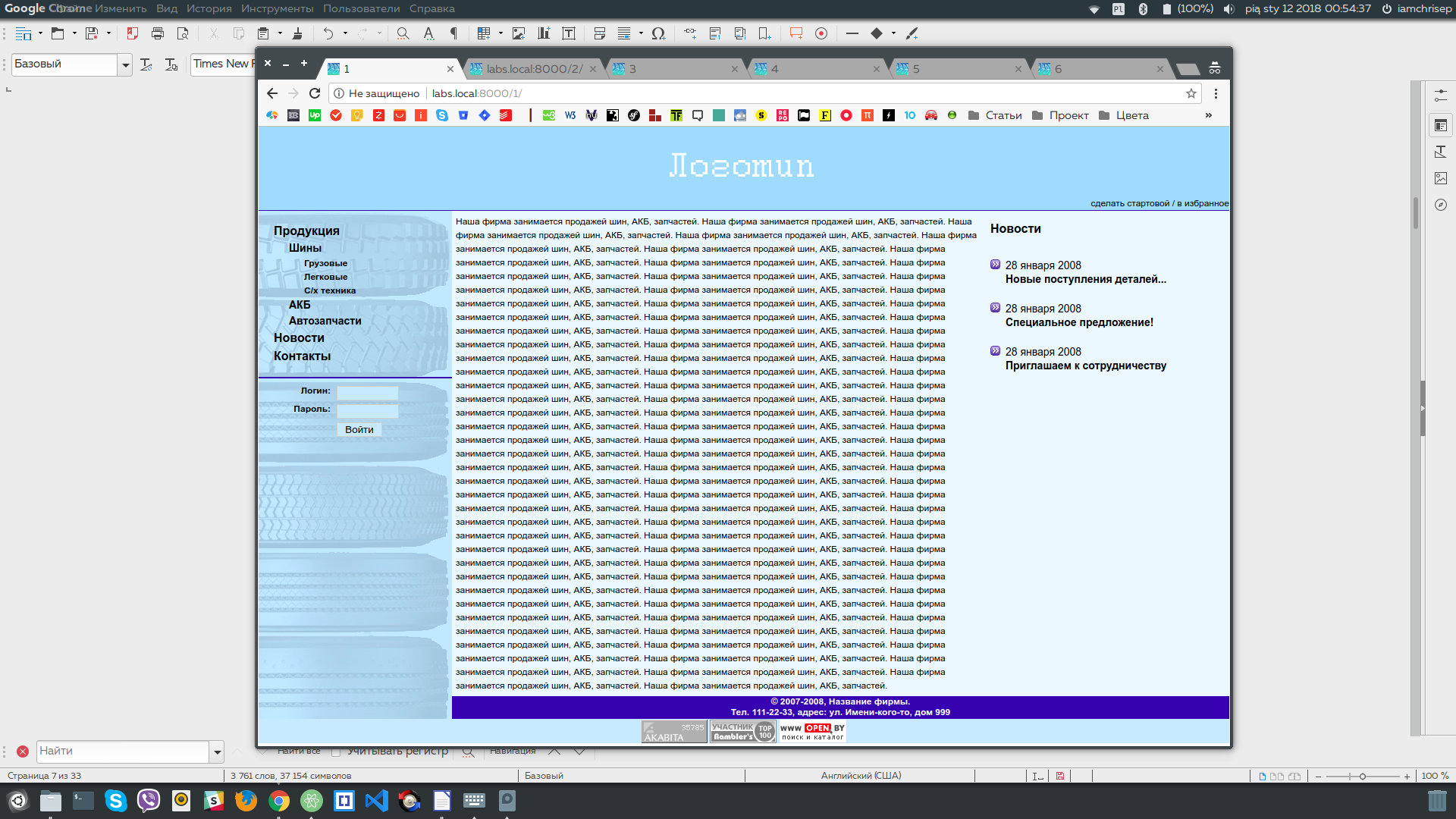
<!-- .footer -->

</div>

<!-- .wrapper-->

</body>

</html>

Рисунок 1.2 – HTML-страница в Ubuntu

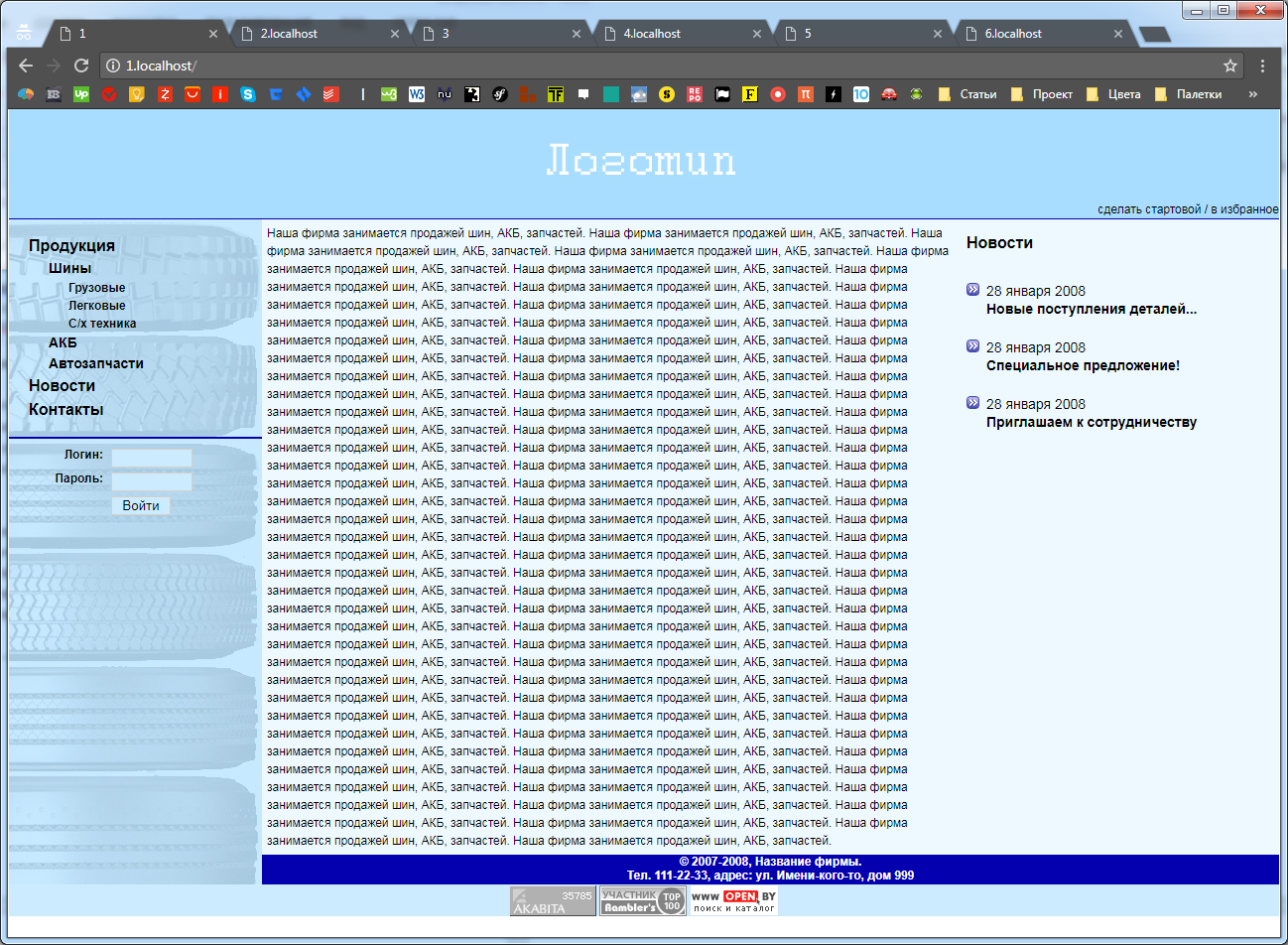


Рисунок 1.3 – HTML-страница в Windows

## Выводы

Верстка веб сайта осуществляется с применением двух основных способов: табличной и блочной div верстки. Блочная верстка более функциональна и предоставляет больше возможностей благодаря использованию CSS.

Особенности. При блочной вёрстке существенное значение уделяется универсальному тегу <div>, который выполняет множество функций. Фактически это основа, на которую «навешиваются» стили, от которых зависит оформление. Несмотря на использование разнообразных тегов в вёрстке с помощью слоёв тег <div> является кирпичиком вёрстки, её базовым фундаментом. В отличие от табличной, блочная верстка обладает рядом преимуществ:

* отделение стиля элементов от кода html;
* возможность наложения одного слоя на другой – такая возможность во многом облегчает позиционирование элементов.
* лучшая индексация поисковиками;
* высокая скорость загрузки страницы, состоящей от взаимно независимых элементов;
* легкость создания визуальных эффектов (выпадающих меню, списков, всплывающих подсказок).

Все стилевые решения вынесены за границы кода html в каскадные таблицы стилей. Доступ к ним осуществляется через идентификаторы или классы css.

Основным недостатком блочной верстки является различия понимания ее кода различными браузерами.

Применение. Создание структуры html-кода, размещающего элементы веб-страницы (изображения, текст и т.д.) в окне браузера, согласно разработанному макету, таким образом, чтобы элементы дизайна выглядели аналогично макету.

# **Задание № 2** «Основы PHP»

Цель работы: изучить теоретический материал по теме «Основы PHP» и закрепить его путем выполнения индивидуального задания.

Задание:

1. Объявить переменные следующих типов: целочисленную, строковую, дробную, логическую, массив.
2. Объявить переменные $a=555 и $b="ZZZ" и сложить их: а) как числа, б) как строки. Результат сложения не помещать в новую переменную, а сразу выводить на экран.
3. Есть три сотрудника:
   1. Иванов, тел. 111-22-33, e-mail: ivanov@domain.com
   2. Петров, тел. 112-24-36, e-mail: petrov@domain.com
   3. Сидоров, тел. 113-25-37, e-mail: sidorov@domain.com

Объявить двухмерный массив, первый уровень которого пронумерован, начиная с нуля, а второй уровень содержит элементы name, phone, email, в которых хранятся соответствующие данные вышеназванных сотрудников.

1. Дан массив, содержащий элементы: 1, 2, "A", 3.764, 34, "B", 12. Объявить этот массив, проанализировать его содержимое и удалить из него все элементы, не являющиеся целыми или дробными числами.
2. Сгенерировать HTML-таблицу, состоящую из трёх колонок и 1000 строк. В первой колонке разместить номера строк таблицы. Цвет каждой строки таблицы должен изменяться по алгоритму: R+1, G+1, B+1, начиная с 000000. Т.е.: первая строка: 000000, вторая – 010101, третья -020202 и т.д. Помните, что числа – шестнадцатеричные и максимальное значение цвета: FFFFFF.

## Решение

Исходный код:

index.php

<?php

echo php\_uname();

echo '<br />'.'<br />'.'<br />'.'<br />';

echo '<b>Задание 1</b>';

echo '<br />'.'<br />';

$int = 123;

echo $int.'<br />';

$str = 'Строка';

echo $str.'<br />';

$flo = 1.23;

echo $flo.'<br />';

$boo = true;

echo $boo.'<br />';

$arr = array ('Элемент 1', 'Элемент 2', 'Элемент 3');

foreach ($arr as $key => $value) {

echo '['.$key.'] = '.$value.'<br />';

}

unset($key);

unset($value);

echo '<br />'.'<br />';

echo '<b>Задание 2</b>';

echo '<br />'.'<br />';

$a=555;

$b='ZZZ';

echo $a.'<br />';

echo $b.'<br />';

echo $a + $b.'<br />';

echo $a . $b.'<br />';

echo '<br />'.'<br />';

echo '<b>Задание 3</b>';

echo '<br />'.'<br />';

$arrarr = array ( 0 => array ( 'name' => 'Иванов',

'phone' => '111-22-33',

'email' => 'ivanov@domain.com'

),

1 => array ( 'name' => 'Петров',

'phone' => '112-24-36',

'email' => 'petrov@domain.com'

),

2 => array ( 'name' => 'Сидоров',

'phone' => '113-25-37',

'email' => 'sidorov@domain.com'

)

);

foreach($arrarr as $k => $v) {

echo $k.':<br />';

foreach($v as $key => $value) {

echo '['.$key.'] = '.$value.'<br />';

}

}

unset($key);

unset($value);

unset($k);

unset($v);

$data = array(1, 2, "A", 3.764, 34, "B", 12);

foreach ($data as $key => $value) {

echo '['.$key.'] = '.$value.'<br />';

}

unset($key);

unset($value);

echo '<br />';

foreach ($data as $key => $value) {

if (!is\_int($value) and !is\_float($value)) {

unset ($data[$key]) ;

}

}

unset($key);

unset($value);

foreach ($data as $key => $value) {

echo '['.$key.'] = '.$value.'<br />';

}

unset($key);

unset($value);

echo '<table border="2" bordercolor="#000000" cellpadding="0" cellspacing="0">';

echo '<tr>

<th>№</th>

<th>Цвет</th>

<th>Значение</th>

</tr>';

$col = 0x00;

for ($i = 1; $i <= 1000; $i++) {

$cols = dechex($col);

if (strlen($cols) == 1) {

$cols = '0'.$cols;

}

$cols = strtoupper($cols.$cols.$cols);

echo '<tr>';

echo '<td>'.$i.'</td>';

echo '<td style="background: #'.$cols.'; width: 70px;"></td>';

echo '<td>'.$cols.'</td>';

echo '</tr>';

if ($col == 0xFF) {

$col = 0x00;

} else {

$col++;

}

}

echo '</table>';

echo '<br />'.'<br />';

echo '<b>Дополнительное задание</b>';

echo '<br />'.'<br />';

echo '

<form method="post" action="index.php">

<p>Name: <input type="text" name="name"></p>

<p>Email: <input type="email" name="email"></p>

<p>Password: <input type="password" name="password"></p>

<p>Phone: <input type="tel" name="phone"></p>

<p>

<input type="checkbox" name="food[]" value="Пирог"> Пирог

<input type="checkbox" name="food[]" value="Торт"> Торт

<input type="checkbox" name="food[]" value="Кекс"> Кекс

</p>

<p>

<input type="radio" name="drink" value="Сок"> Сок

<input type="radio" name="drink" value="Чай"> Чай

<input type="radio" name="drink" value="Кофе"> Кофе

</p>

<p>

Date: <input type="date" name="date">

</p>

<p>

Color: <input type="color" name="color">

</p>

<p>

Range: <input type="range" name="range">

</p>

<input type="submit">

</form>

';

if ($\_POST) {

$t='';

foreach($\_POST as $k => $v) {

if ($k == 'food') {

foreach($v as $key => $value) {

$t.=$k."[".$key."] = \"".$value."\" \n";

}

} else {

$t.=$k." = \"".$v."\" \n";

}

}

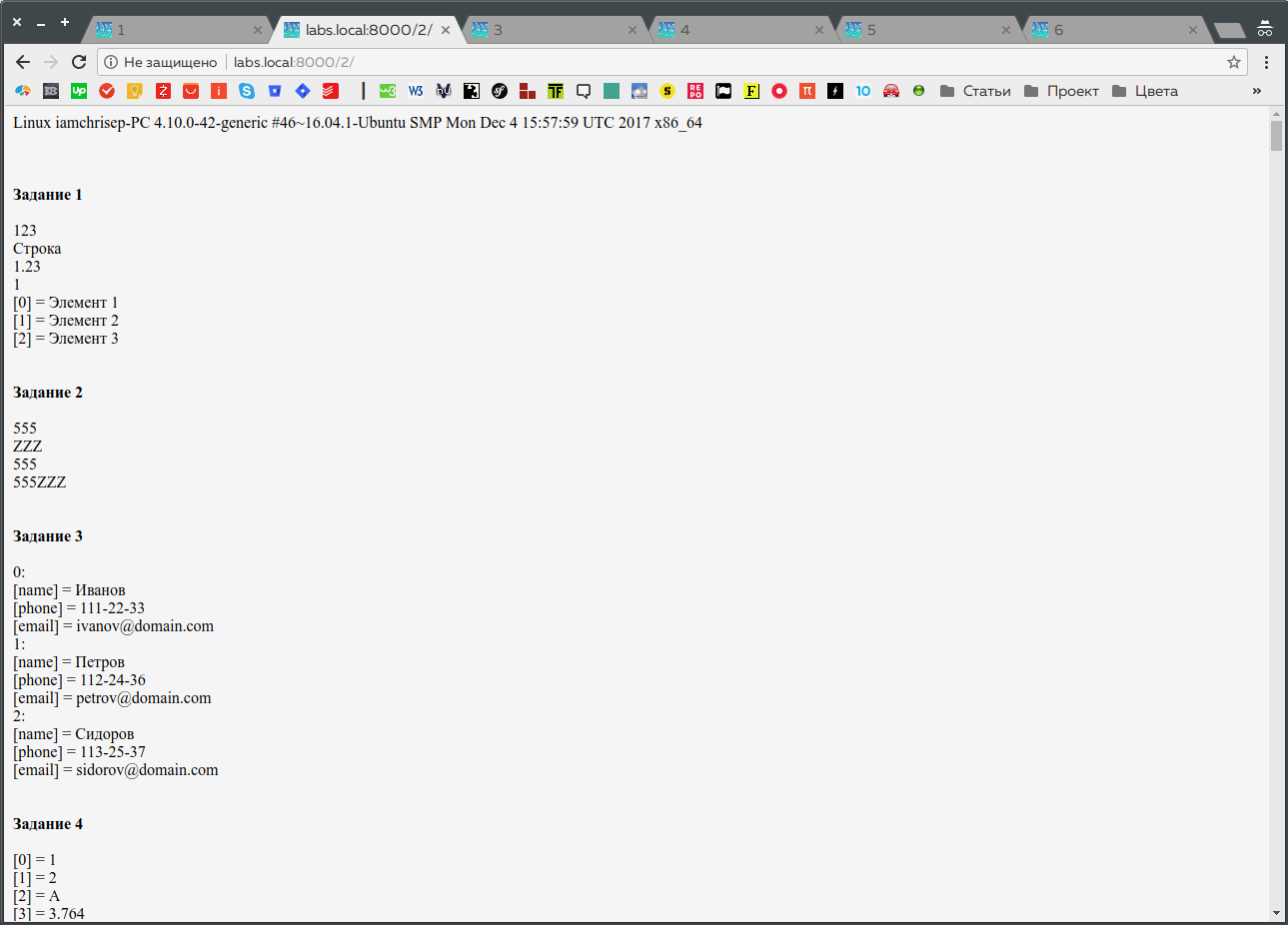
$f = fopen('form.txt', 'w');

fwrite($f, $t);

fclose($f);

}

?>

Рисунок 2.1 – Результат выполнения в Ubuntu

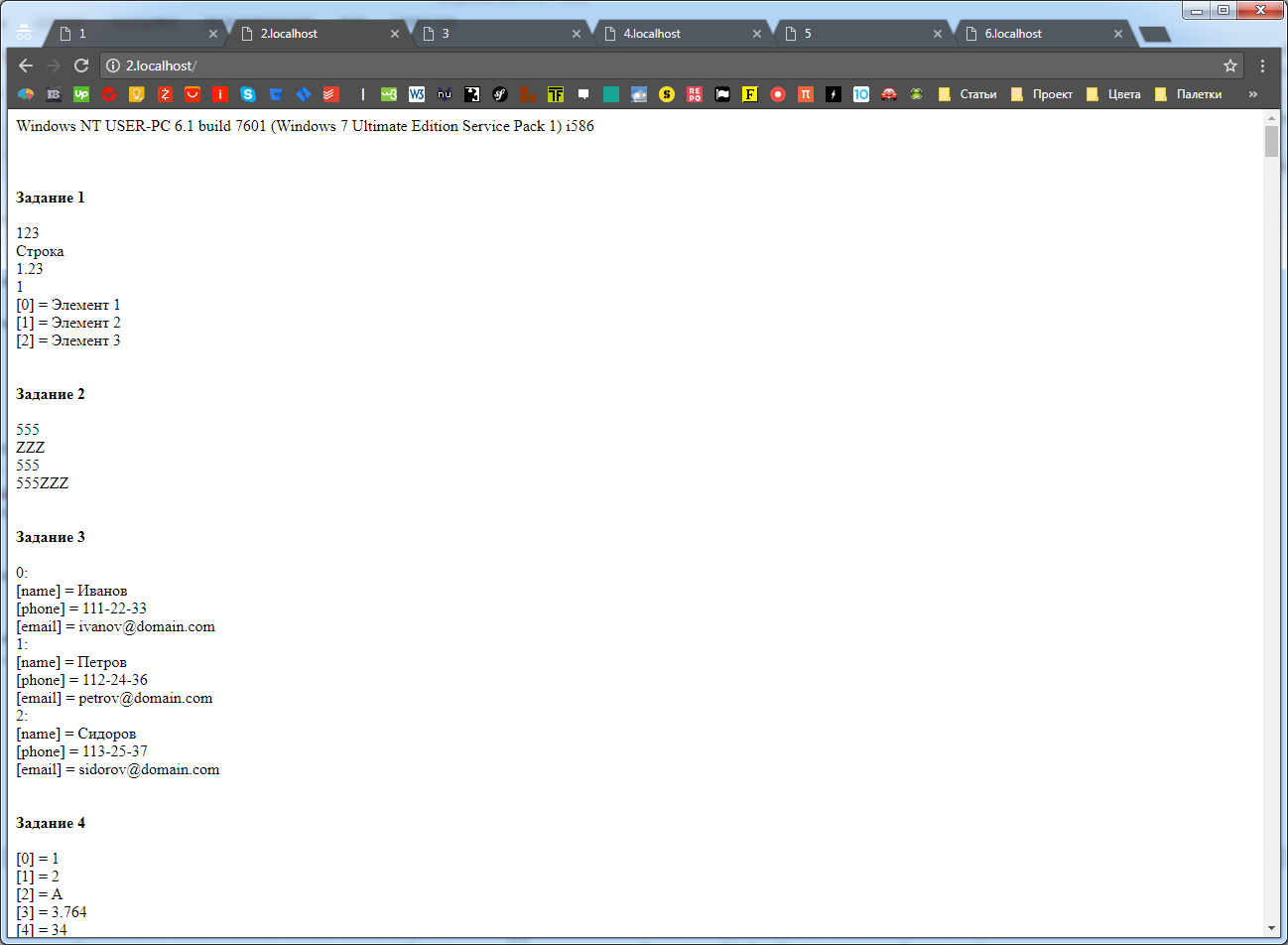


Рисунок 2.2 – Результат выполнения в Windows

## Выводы

В PHP поддерживается восемь простых типов данных (переменных): четыре скалярных типа: boolean (логические данные), integer (целые числа), float (число с плавающей точкой или 'double'), string (строки). Два смешанных типа: array (массивы), object (объекты). И два специальных типа: resource (ресурсы), NULL (пустой тип).

Виды операторов в PHP: Унарные (работают только с одним аргументом, например ! - оператор отрицания), бинарные, и тернарный оператор ? (используется для условного выбора между двумя операторами).

PHP не требует явного типа при определении переменной; тип переменной определяется по контексту, в котором она используется. То есть, при присвоении значения типа string переменной $var, она станет строкой. Если затем присвоить $var целочисленное значение, она станет целым числом. Приведение типов в PHP работает так же, как и в C: имя требуемого типа записывается в круглых скобках перед приводимой переменной. Вместо использования приведения переменной к string, можно также заключить ее в двойные кавычки.

Особенности. PHP доступен для большинства операционных систем, включая Linux, многие модификации Unix (такие как HP-UX, Solaris и OpenBSD), Microsoft Windows, Mac OS X, RISC OS, и многие другие. Также в PHP включена поддержка большинства современных веб-серверов, таких как Apache, IIS и многих других.

Применение. Главная область применения PHP - написание скриптов, работающих на стороне сервера; таким образом, PHP способен выполнять все то, что выполняет любая другая программа CGI, например, обрабатывать данные форм, генерировать динамические страницы или отсылать и принимать cookies.

PHP способен генерировать не только HTML. Доступно формирование изображений, файлов PDF и даже роликов Flash (с использованием libswf и Ming), создаваемых «на лету». PHP также способен генерировать любые текстовые данные, такие, как XHTML и другие XML-файлы. PHP может осуществлять автоматическую генерацию таких файлов и сохранять их в файловой системе вашего сервера вместо того, чтобы отдавать клиенту, организуя, таким образом, серверный кэш для вашего динамического контента.

# **Задание № 3** «Основы разработки сайтов»

Цель работы: изучить теоретический материал по теме «Основы разработки сайтов» и закрепить его путем выполнения индивидуального задания.

Задание:

1. Прочитать содержимое шаблона main.tpl в текстовую переменную.

2. Плейсхолдер {MAIN\_MENU} заменить на содержимое шаблона main\_menu.tpl

3. Блоки: а) с основным текстом страницы, б) новостями, в) областью с адресом и копирайтом удалить из шаблона main.tpl, пометив места их расположения плейсхолдерами. Содержимое удаляемых блоков разместить в отдельных шаблонах. Обеспечить замену плейсхолдеров содержимым соответствующих шаблонов (аналогично пункту 2).

4. Плейсхолдеры {TODAY\_D}, {TODAY\_M}, {TODAY\_Y}, {NOW\_H}, {NOW\_M}, {NOW\_S} заменить на фрагменты текущей даты и времени: день, месяц, год, час, минута, секунда соответственно.

5. Создать конфигурационный файл site.cfg (формат разработать самостоятельно; примечание: делать этот файл в формате php ЗАПРЕЩЕНО, т.е. это должен быть "простой текстовый файл"), в котором разместить две переменные: main\_color и copyright\_color, в которых хранить (и использовать для управления сайтом) цвета области с основным текстом страницы и области с адресом и копирайтом.

## Решение

Исходный код:

index.php

<?php

$fmain = file\_get\_contents('templates/main.tpl');

$fmainmenu = file\_get\_contents('templates/main\_menu.tpl');

$fmain = str\_replace('{MAIN\_MENU}',$fmainmenu,$fmain);

$text = file\_get\_contents('templates/text.tpl');

$fmain = str\_replace('{text}',$text,$fmain);

$fmain = str\_replace('{TODAY\_D}',date("d"),$fmain);

$fmain = str\_replace('{TODAY\_M}',date("m"),$fmain);

$fmain = str\_replace('{TODAY\_Y}',date("y"),$fmain);

$fmain = str\_replace('{NOW\_H}',date("H"),$fmain);

$fmain = str\_replace('{NOW\_M}',date("i"),$fmain);

$fmain = str\_replace('{NOW\_S}',date("s"),$fmain);

$fnews = file\_get\_contents('templates/news.tpl');

$flogo = file\_get\_contents('templates/logo.tpl');

$fmain = str\_replace('{LOGO}',$flogo,$fmain);

$fnews\_str = file\_get\_contents('templates/news\_str.tpl');

$array\_news = file('news.inf');

$str\_all = "";

for($i = 0; $i < count($array\_news); $i++)

{

if ($i%2 == 0)

{

$str1 = $fnews\_str;

$str1 = str\_replace('{news\_date}',$array\_news[$i],$str1);

}

else

{

$str1 = str\_replace('{news\_text}',$array\_news[$i],$str1);

$str\_all .=$str1;

}

}

$fnews = str\_replace('{news\_str}',$str\_all,$fnews);

$fmain = str\_replace('{news}',$fnews,$fmain);

$mcfg = file('site.cfg');

$cfg0 = str\_word\_count($mcfg[0], 1);

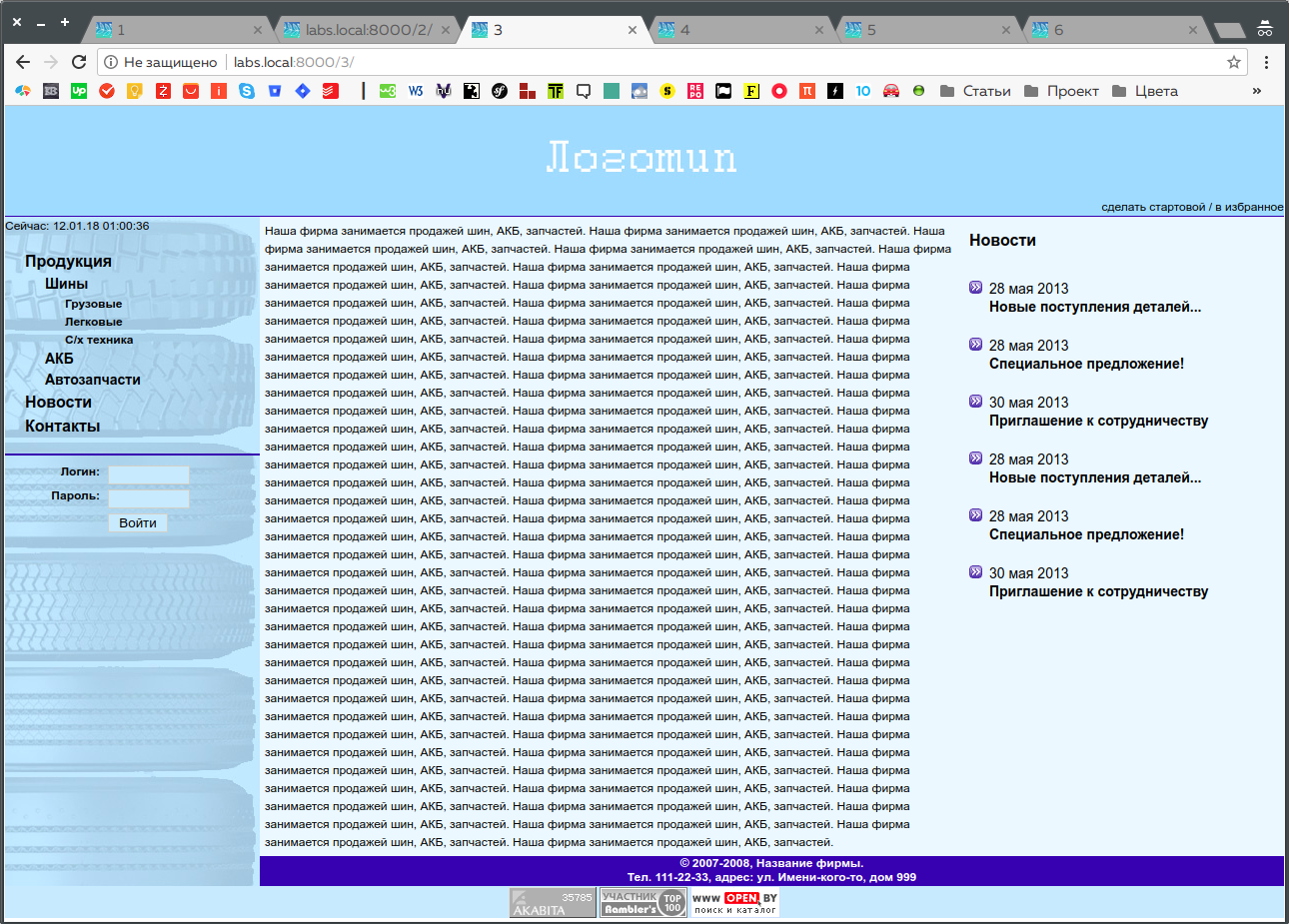
$cfg1 = str\_word\_count($mcfg[1], 1);

$fmain = str\_replace('{main\_color}', $cfg0[2], $fmain);

$fmain = str\_replace('{copyright\_color}', $cfg1[2], $fmain);

echo($fmain);

?>

Рисунок 3.1 – Результат выполнения в Ubuntu

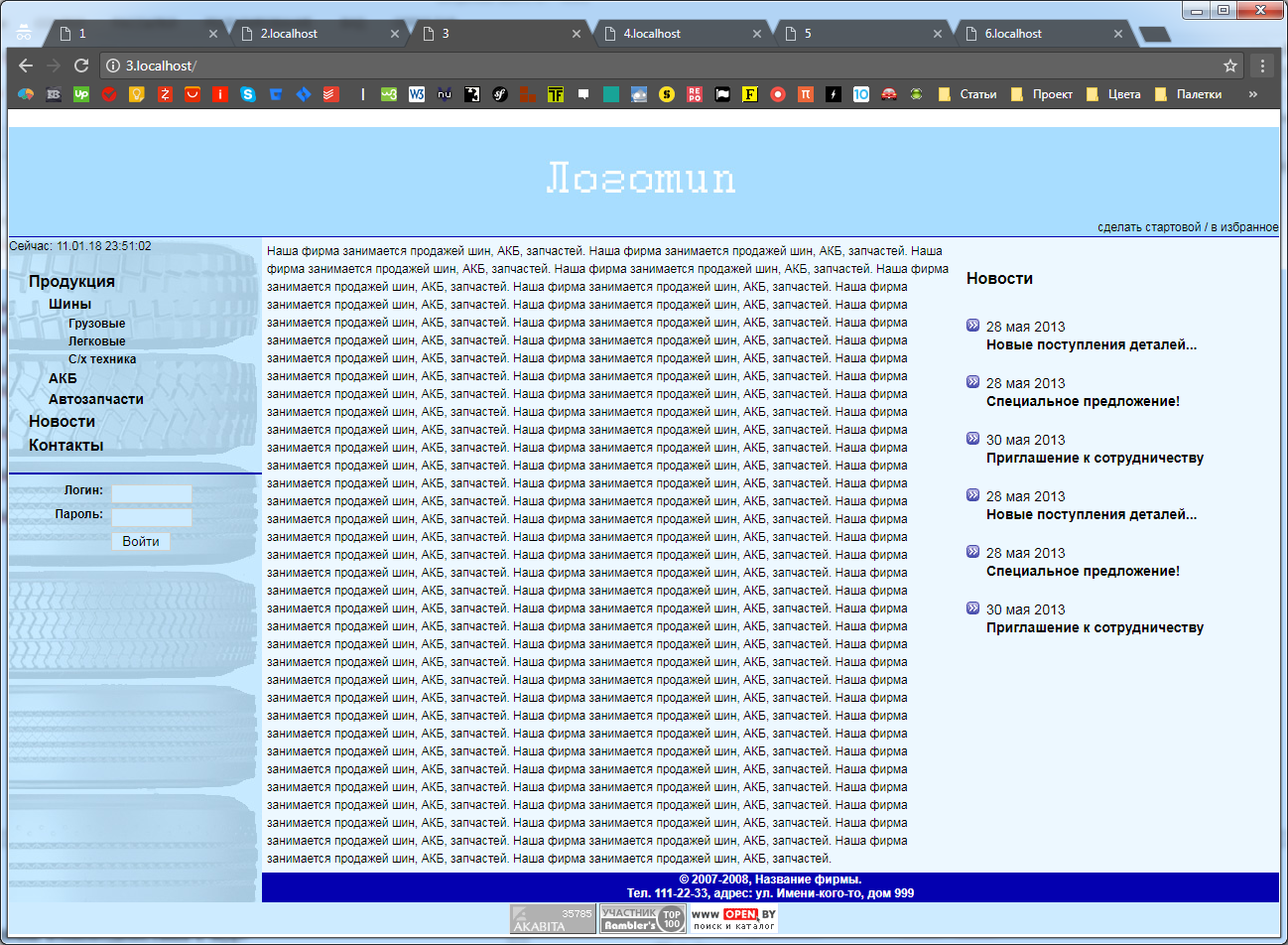


Рисунок 3.2 – Результат выполнения в Windows

## Выводы

В ходе выполнения работы были изучены и применены на практике методы работы с шаблонами в PHP.

Особенности. Использование шаблонов при создании сайта помогает сэкономить время разработчиков, т.к. обеспечивает разделение логики работы приложения от способа представления данных, т. е. дизайна, что делает код понятным и удобным для восприятия и редактирование при работе в команде.

Шаблоны также хороши тем, что в случае, когда часть информации полностью дублируется на нескольких страницах ее можно положить в один файл и использовать при генерации всех этих страниц.

Применение. Генерация страниц с одной и той же информацией, но различным оформлением – например обычная версия и версия для печати.

# **Задание № 4** «Основы взаимодействия с БД»

Цель работы: изучить теоретический материал по теме «Основы взаимодействия с БД» и закрепить его путем выполнения индивидуального задания.

Задание:

1. Создать БД с уникальным именем.

2. В созданной БД создать таблицу news, содержащую поля: уникальный идентификатор новости, дата публикации новости, заголовок новости, текст новости.

3. Написать функцию, отображающую на сайте N самых свежих новостей (N задаётся как параметр функции), новости сортируются по убыванию даты. Рекомендуется для выполнения этого задания использовать шаблоны.

4. В созданной БД создать таблицу pages, содержащую поля: уникальный идентификатор страницы, информация о родительской странице, текст для отображения в меню.

5. На основе таблицы, разработанной в пункте 4, сформировать меню сайта. Меню должно иметь не менее трёх уровней вложенности. (Необязательное требование: при клике по некоторому пункту меню должен открываться ("разворачиваться") его первый подуровень, остальные пункты меню при этом находятся в "свёрнутом" состоянии).

## Решение

Исходный код:

index.php

<?php

$fmain = file\_get\_contents('templates/main.tpl');

$fmainmenu = getContains();

$fmain = str\_replace('{MAIN\_MENU}',$fmainmenu,$fmain);

$text = file\_get\_contents('templates/text.tpl');

$fmain = str\_replace('{text}',$text,$fmain);

$fmain = str\_replace('{TODAY\_D}',date("d"),$fmain);

$fmain = str\_replace('{TODAY\_M}',date("m"),$fmain);

$fmain = str\_replace('{TODAY\_Y}',date("y"),$fmain);

$fmain = str\_replace('{NOW\_H}',date("H"),$fmain);

$fmain = str\_replace('{NOW\_M}',date("i"),$fmain);

$fmain = str\_replace('{NOW\_S}',date("s"),$fmain);

$fnews = file\_get\_contents('templates/news.tpl');

$flogo = file\_get\_contents('templates/logo.tpl');

$fmain = str\_replace('{LOGO}',$flogo,$fmain);

$N = 2;

$str\_all = getNews($N);

$fnews = str\_replace('{news\_str}', $str\_all, $fnews);

$fmain = str\_replace('{news}', $fnews, $fmain);

$mcfg = file('site.cfg');

$cfg0 = str\_word\_count($mcfg[0], 1);

$cfg1 = str\_word\_count($mcfg[1], 1);

$fmain = str\_replace('{main\_color}', $cfg0[2], $fmain);

$fmain = str\_replace('{copyright\_color}', $cfg1[2], $fmain);

echo($fmain);

function getNews($count)

{

$fnews\_str = file\_get\_contents('templates/news\_str.tpl');

$connection = new mysqli("localhost", "root", "12345", "news");

if ($connection->connect\_errno) {

printf("Не удалось подключиться: %s\n", $connection->connect\_error);

exit();

}

if (!$connection->set\_charset("utf8")) {

printf("Ошибка при загрузке набора символов utf8: %s\n", $connection->error);

exit();

}

$qry = $connection->query("SELECT \* FROM news ORDER BY datatime desc limit 0, $count ");

$news\_str = "";

$i = 0;

while($row = $qry->fetch\_array(MYSQLI\_BOTH))

{

$new\_str = $fnews\_str;

$new\_str = str\_replace('{news\_date}', $row[1], $new\_str);

$new\_str = str\_replace('{news\_text}', $row[2], $new\_str);

$news\_str .=$new\_str;

}

$qry->close();

$connection->close();

return $news\_str;

}

function getItemMenu($id)

{

$connection = new mysqli("localhost", "root", "12345", "news");

if ($connection->connect\_errno) {

printf("Не удалось подключиться: %s\n", $connection->connect\_error);

exit();

}

if (!$connection->set\_charset("utf8")) {

printf("Ошибка при загрузке набора символов utf8: %s\n", $connection->error);

exit();

}

$qry = $connection->query("SELECT \* FROM pages WHERE ID = $id");

while($row = $qry->fetch\_array(MYSQLI\_BOTH))

{

return $row[2];

}

$qry->close();

$connection->close();

}

function getContains()

{

return '

<ul class="nav">

<li><a href="#">'. getItemMenu(1).'</a></li>

<li><a href="#">'. getItemMenu(2).'</a>

<ul class="sub\_level">

<li><a href="#">'. getItemMenu(3).'</a></li>

<li><a href="#">'. getItemMenu(4).'</a>

<ul class="sub\_sub\_level">

<li><a href="#">'. getItemMenu(5).'</a></li>

<li><a href="#">'. getItemMenu(6).'</a></li>

<li><a href="#">'. getItemMenu(7).'</a></li>

</ul>

</li>

<li><a href="#">'. getItemMenu(8).'</a></li>

<li><a href="#">'. getItemMenu(9).'</a></li>

</ul>

</li>

<li><a href="#">'. getItemMenu(6).'</a></li>

<li><a href="#">'. getItemMenu(7).'</a></li>

</ul>

';

}

?>

script.js

$(".nav > li").hover(function() {

var a = $(this).find("ul.sub\_level");

a.toggle();

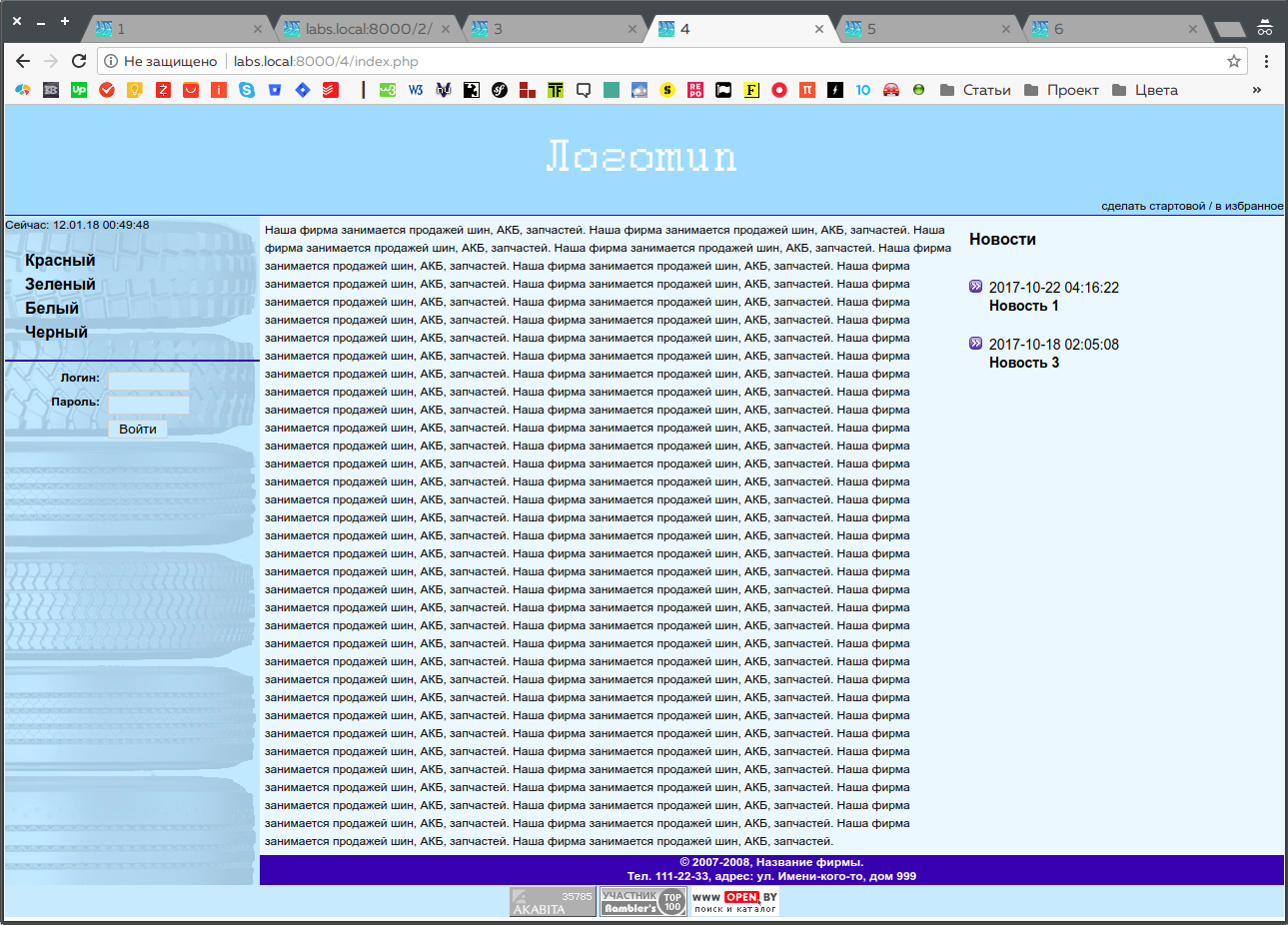
})

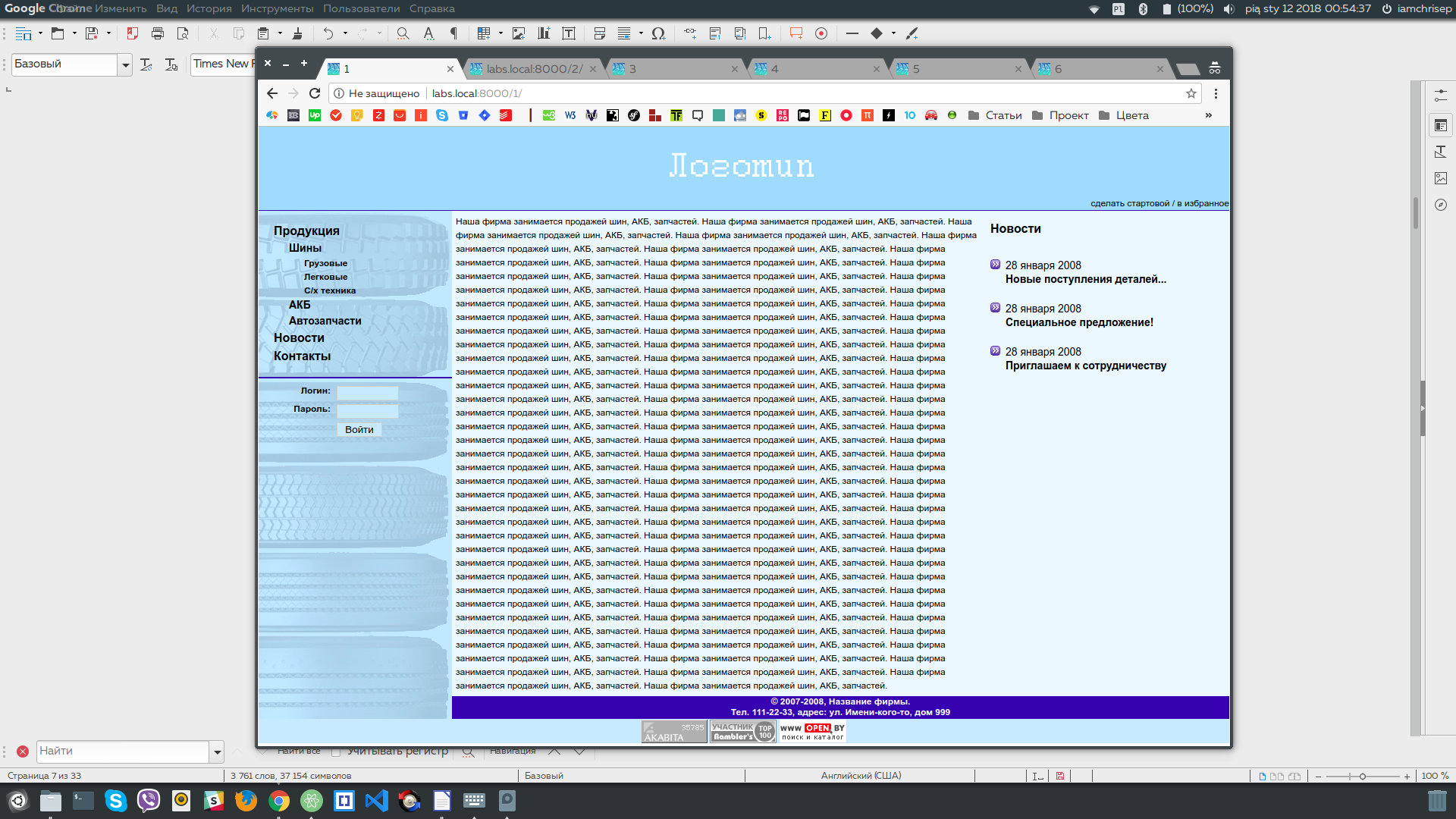
$(".sub\_level > li").hover(function() {

var a = $(this).find("ul.sub\_sub\_level");

a.toggle();

})

Рисунок 4.1 – Результат выполнения в Ubuntu

Рисунок 4.2 – Результат выполнения в Windows

## Выводы

Реляционная база данных – база данных, основанная на реляционной модели, для работы с ней применяют реляционные СУБД.

Для подключения к БД в PHP используется функция mysql\_connect ( [string server [, string username [, string password [, bool new\_link [, int client\_flags]]]]] ), функция возвращает указатель на соединение с MySQL или false при неудаче(параметр new\_link равный TRUE может заставить открыть еще одно соединение). Соединение с сервером завершается при завершении скрипта или с помощью функции mysql\_close(). Для постоянного подключения используют функцию mysql\_pconnect(). Для выбора базы данных с которой ведется работа используется функция mysql\_select\_db. Для отправки запроса к базе данных используется функция mysql\_query(). Mysql\_error() – возвращает строку, содержащую текст ошибки выполнения последней функции. Для извлечения данных используется mysql\_fetch\_array ( resource result [, int result\_type] ) – возвращает массив с одним рядом результата запроса, или FALSE, если рядов больше нет.

Особенности:

* Данные хранятся в таблицах, состоящих из столбцов и строк;
* На пересечении каждого столбца и строчки стоит в точности одно значение;
* У каждого столбца есть своё имя, которое служит его названием, и все значения в одном столбце имеют один тип. Например, в столбце id\_forum все значения имеют целочисленный тип, а в строке name - текстовый;
* Столбцы располагаются в определённом порядке, который определяется при создании таблицы, в отличие от строк, которые располагаются в произвольном порядке. В таблице может не быть не одной строчки, но обязательно должен быть хотя бы один столбец;

Запросы к базе данных возвращают результат в виде таблиц, которые тоже могут выступать как объект запросов.

Применение. Реляционные БД находят применение повсеместно:

* В организациях для учёта персонала, ведения бухгалтерии, учёта товаров на складе, поставщиков, партнёров, клиентов, ведения электронного документооборота.
* В адресных и телефонных книгах, словарях, справочниках.
* В биллинговых системах для учёта трафика у интернет-провайдеров, потреблённых услуг у телефонных операторов, в банковском деле.
* В интернет-технологиях для организации хранения учётных записей зарегистрированных пользователей, текстов сообщений на форумах, в гостевых книгах, социальных сетях, интернет-дневниках и в новостных лентах. Без всякого сомнения, базы данных используются поисковыми системами для хранения индексов отсканированных веб-страниц.

# **Задание № 5** «Основы работы с регулярными выражениями»

Цель работы: изучить теоретический материал по теме «Основы работы с регулярными выражениями» и закрепить его путем выполнения индивидуального задания.

Задание:

1. Написать регулярное выражение, проверяющее, является ли строка корректным e-mail адресом.

Примеры корректных адресов:

vasya-pupkin@mail.com

vasya\_pupkin@mail.com

vasya.pupkin@mail.com

v.v.pupkin@firma.mail.com

v.v.pupkin@firma-mail.com

v.v.pupkin12@firma\_mail.com

v.v.pupkin-director@firma.mail.com

Примеры некорректных адресов:

-vasya--pupkin@mail.com

vasya\_pupkin@mail..com

vasya.-pupkin@mail.com

v.v.pup kin@firma.mail.com

v.v.pup#kin@firma-mail.com\_

2. Написать регулярное выражение, удаляющее из текста HTML-комментарии, т.е. конструкции вида

<!-- комментарий -->

При этом учесть, что конструкции вида

<!--[if IE]><link href="/css/invstroyIEfix.css" rel="stylesheet" type="text/css" /><![endif]-->

удалять не надо, т.к. это, фактически, не комментарий.

3. Написать регулярное выражение, очищающее текст от HTML-тегов.

4. Написать регулярное выражение, выделяющее красным жирным шрифтом все слова с длиной в 5 и более символов, состоящие только из заглавных букв.

## Решение

Исходный код:

index.php

<html>

<head>

<meta http-equiv="content-type" content="text/html;charset=utf-8" />

<title>5</title>

</head>

<body>

<?php

echo "<b>Задание 1</b><br>";

function CheckEmail($str)

{

if ( preg\_match("|^[a-z]{1,}([\\_\-\.]?[a-z0-9])\*@[a-z]{1,}([\\_\-\.]?[a-z0-9])\*\.[a-z]{2,6}$|i", $str) ) {return "корректный email адрес";} else {return "некорректный email адрес";}

}

$cfg\_file = file('emails.txt');

for($i = 0; $i < count($cfg\_file); $i++)

{

$email = trim($cfg\_file[$i]);

echo "<br> $email - ".CheckEmail($email);

}

echo '<br><br>';

echo "<b>Задание 2</b><br>";

$str = 'текст, <!-- первый комментарий --> <b>текст в тегах</b> <!--[if IE]>

условный комментарий <!-- второй комментарий --> <i>второй текст в тегах</i> <![endif]-->';

echo "<br>Текст с HTML-комментариями:<br>".htmlspecialchars($str);

$str = preg\_replace("|\<!--(?!\[if .?)[\s\S]\*?--\>|im", "", $str);

echo "<br>Текст без HTML-комментариев:<br>".htmlspecialchars($str);

echo '<br><br>';

echo "<b>Задание 3</b><br>";

$html = '<html><head>

<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />

<title>Заголовок</title></head>

<body>

<p>Параграф текста <b>полужирный текст</b> <img src="#" /> </p>

</body>

</html>';

echo "<br>Текст с HTML-тегами:<br>".htmlspecialchars($html);

$html = preg\_replace("|<.\*?>|", "", $html);

echo "<br>Текст без HTML-тегов:<br>".htmlspecialchars($html);

echo '<br><br>';

echo "<b>Задание 4</b><br>";

$words = 'Текст ТЕКСТ текст TExtt bigtext BIGBIGTEXT myTEXXTmy';

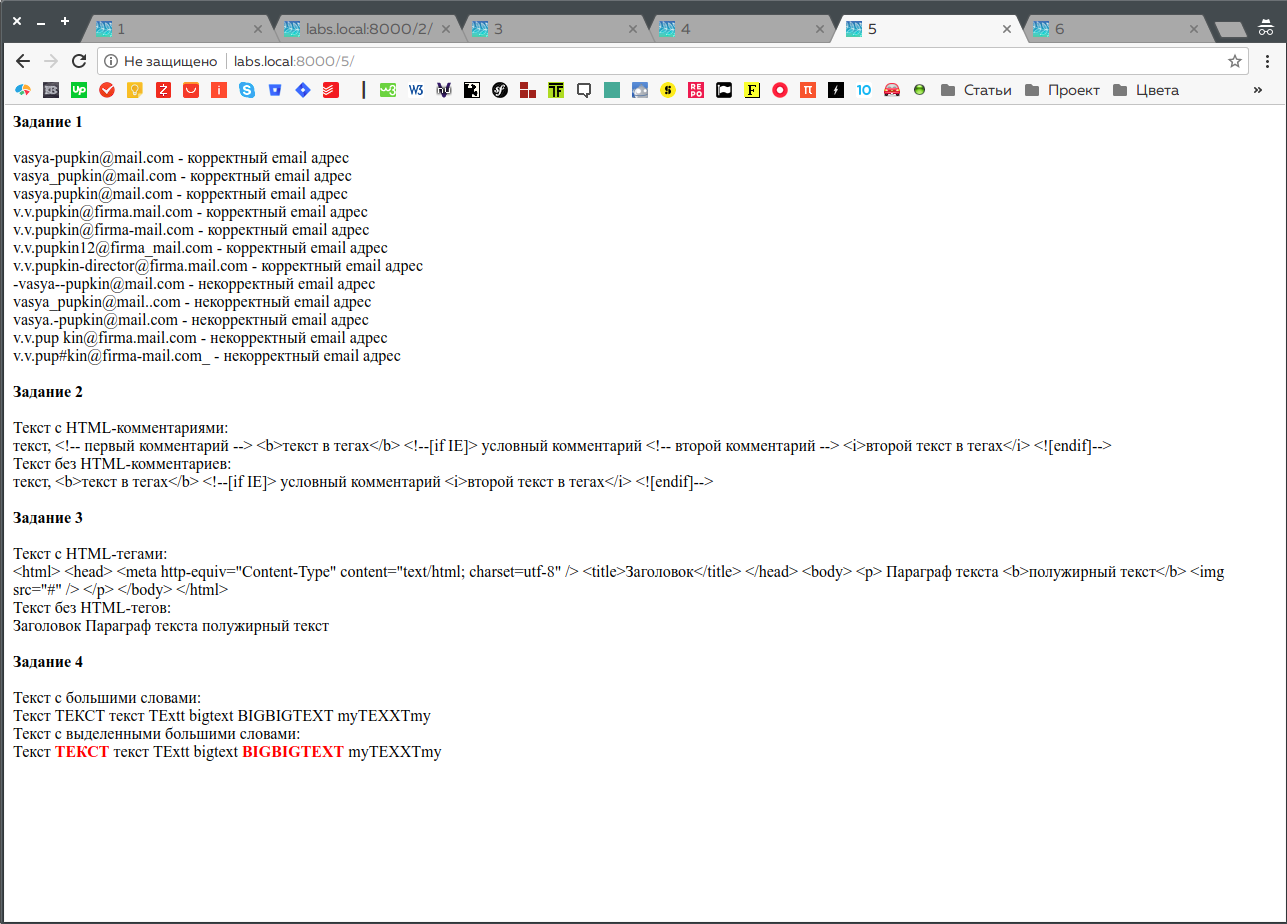
echo "<br>Текст с большими словами:<br>$words";

$words = preg\_replace("|(\b[A-ZА-Я]{5,}\b)|u", "<b><font color=red>$1</font></b>", $words);

echo "<br>Текст с выделенными большими словами:<br>$words";

?>

</body>

Рисунок 5.1 – Результат выполнения в Ubuntu

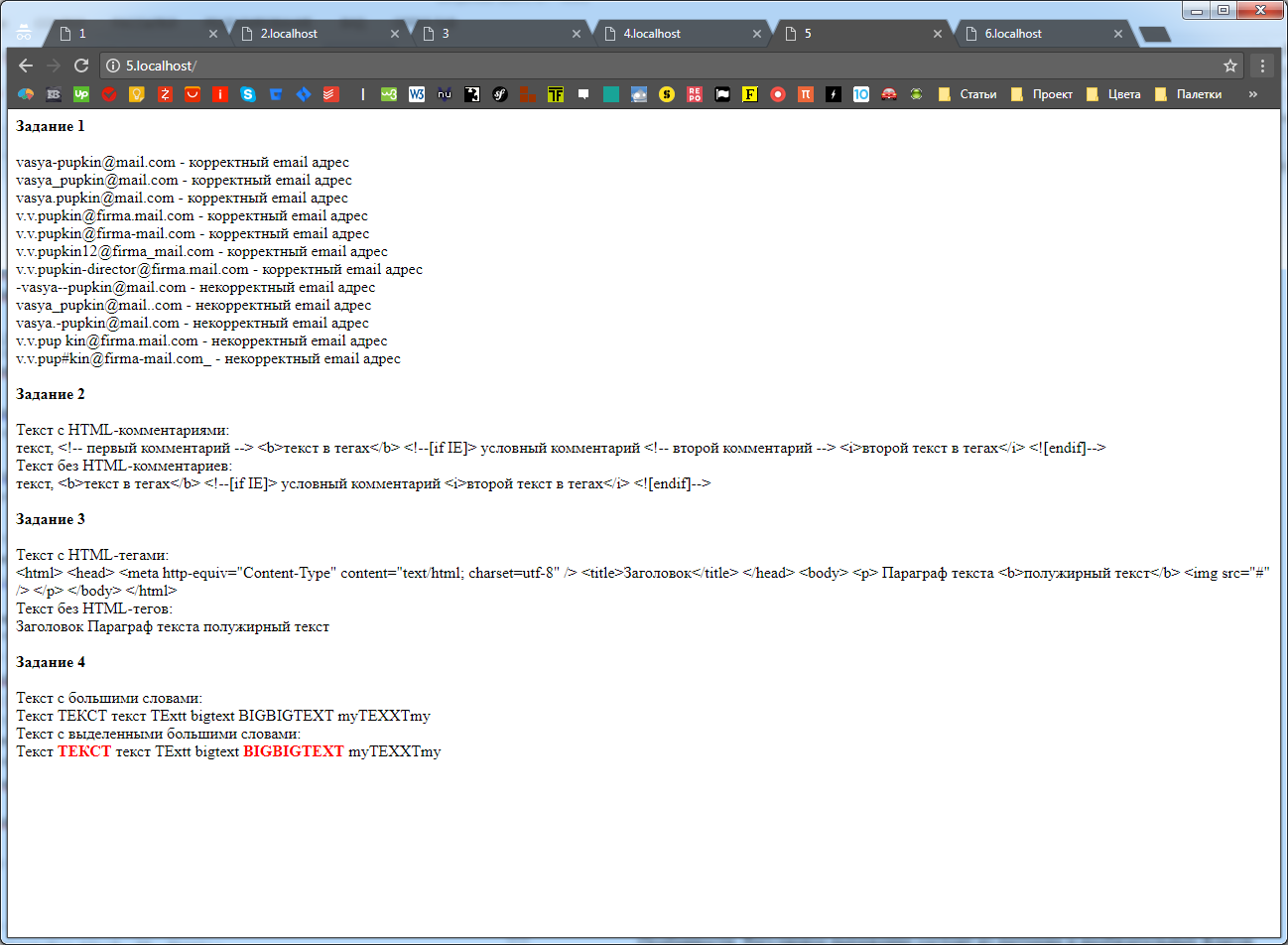


Рисунок 5.2 – Результат выполнения в Windows

## Выводы

Регулярные выражения – система обработки текста, основанная на специальной системе записи образцов для поиска. Образец (англ. pattern), задающий правило поиска, по-русски также иногда называют «шаблоном», «маской».

Особенности. Регулярное выражение состоит из паттерна и необязательных флагов. Основа регулярного выражения – паттерн. Это строка, которую можно расширить специальными символами, которые делают поиск намного мощнее. В простейшем случае, если флагов и специальных символов нет, поиск по паттерну – то же самое, что и обычный поиск подстроки. Регулярные выражения могут иметь флаги, влияющие на поиск. В JavaScript их всего три: i – если этот флаг есть, то выражение ищет независимо от регистра, g – если этот флаг есть, то выражение ищет все совпадения, иначе – только первое, m – многострочный режим.

Применение. Они используются многими текстовыми редакторами и утилитами для поиска и изменения текста на основе выбранных правил. Многие языки программирования уже поддерживают регулярные выражения для работы со строками. Например, Perl и Tcl имеют встроенный в их синтаксис механизм обработки регулярных выражений. Набор утилит, поставляемых в дистрибутивах Unix, одним из первых популяризировал понятия регулярных выражений.

# **Задание № 6** «Основы работы с лентами новостей»

Цель работы: изучить теоретический материал по теме «Основы работы с лентами новостей» и закрепить его путем выполнения индивидуального задания.

Задание:

Представить новости в формате RSS.

Пример представления:

<?xml version="1.0" encoding="windows-1251" ?>

<rss version="2.0">

<channel>

<title>Название ленты новостей</title>

<link>http://www.somesite.com/rss/</link>

<description>Описание ленты новостей</description>

<lastBuildDate>Sun, 18 Jan 2009 16:49:01 +0200</lastBuildDate>

<item>

<title>Заглавие новости (дата публикации новости)</title>

<link>http://www.somesite.com/news/newsid4792/</link>

<description>Краткое содержание новости</description>

<comments>http://www.somesite.com/news/newsid4792/</comments>

<pubDate>Fri, 16 Jan 2009 21:09:01 +0200</pubDate>

<guid>http://www.somesite.com/news/newsid4792/</guid>

</item>

……………………………………….

</channel>

</rss>

Здесь:

Название ленты новостей – произвольное название (напр. "Наши новости").

http://www.somesite.com/rss/ – URL ленты новостей. Можете взять значение из примера.

Описание ленты новостей – произвольное описание (напр. "Самые свежие в мире новости у нас!")

Sun, 18 Jan 2009 16:49:01 +0200 – дата обновления ленты новостей. Генерируйте на основе текущей даты (и времени) в таком же формате.

Заглавие новости (дата публикации новости) – заглавие и дата публикации новости в произвольном формате (напр. "Скоро лето! (01.01.2000)" )

http://www.somesite.com/news/newsid4792/ -- ссылка на полный текст новости. Вам нужно менять только идентификатор (помечено красным). Остальное значение можно просто переписать из примера.

Fri, 16 Jan 2009 21:09:01 +0200 – дата публикации новости в ленте. Генерируйте на основе даты-времени внесения новости в БД в таком же формате.

## Решение

Исходный код:

index.php

<?php

$fmain = file\_get\_contents('templates/main.tpl');

$fmainmenu = getContains();

$fmain = str\_replace('{MAIN\_MENU}',$fmainmenu,$fmain);

$text = file\_get\_contents('templates/text.tpl');

$fmain = str\_replace('{text}',$text,$fmain);

$fmain = str\_replace('{TODAY\_D}',date("d"),$fmain);

$fmain = str\_replace('{TODAY\_M}',date("m"),$fmain);

$fmain = str\_replace('{TODAY\_Y}',date("y"),$fmain);

$fmain = str\_replace('{NOW\_H}',date("H"),$fmain);

$fmain = str\_replace('{NOW\_M}',date("i"),$fmain);

$fmain = str\_replace('{NOW\_S}',date("s"),$fmain);

$fnews = file\_get\_contents('templates/news.tpl');

$fnews\_str = file\_get\_contents('templates/news\_str.tpl');

$flogo = file\_get\_contents('templates/logo.tpl');

$fmain = str\_replace('{LOGO}',$flogo,$fmain);

$news\_all = new SimpleXMLElement(getNews());

$str\_all = "";

for($i = 0; $i < count($news\_all->channel[0]->item); $i++)

{

$str1 = $fnews\_str;

$str1 = str\_replace('{news\_date}',$news\_all->channel[0]->item[$i]->pubDate,$str1);

$str1 = str\_replace('{news\_text}',$news\_all->channel[0]->item[$i]->title,$str1);

$str\_all .=$str1;

}

$fnews = str\_replace('{news\_str}', $str\_all, $fnews);

$fmain = str\_replace('{news}', $fnews, $fmain);

$mcfg = file('site.cfg');

$cfg0 = str\_word\_count($mcfg[0], 1);

$cfg1 = str\_word\_count($mcfg[1], 1);

$fmain = str\_replace('{main\_color}', $cfg0[2], $fmain);

$fmain = str\_replace('{copyright\_color}', $cfg1[2], $fmain);

echo($fmain);

function getNews()

{

$html = '<?xml version="1.0" encoding="utf-8" ?>

<rss version="2.0">

<channel>

<title>Название ленты новостей</title>

<link>http://www.somesite.com/rss/</link>

<description>Описание ленты новостей</description>

<lastBuildDate>'.date(DATE\_RSS).'</lastBuildDate>

';

$connection = new mysqli("localhost", "root", "12345", "news");

if ($connection->connect\_errno) {

printf("Не удалось подключиться: %s\n", $connection->connect\_error);

exit();

}

if (!$connection->set\_charset("utf8")) {

printf("Ошибка при загрузке набора символов utf8: %s\n", $connection->error);

exit();

}

$qry = $connection->query("SELECT \* FROM rss");

while($row = $qry->fetch\_array(MYSQLI\_BOTH)){

$html .= '

<item>

<title>'.$row[0].'</title>

<link>'.$row[1].'</link>

<description>'.$row[2].'</description>

<comments>http:'.$row[3].'</comments>

<pubDate>'.$row[4].'</pubDate>

<guid>'.$row[5].'</guid>

</item>';

}

$html .= '</channel>

</rss>

';

$qry->close();

$connection->close();

return $html;

}

function getItemMenu($id)

{

$connection = new mysqli("localhost", "root", "12345", "news");

if ($connection->connect\_errno) {

printf("Не удалось подключиться: %s\n", $connection->connect\_error);

exit();

}

if (!$connection->set\_charset("utf8")) {

printf("Ошибка при загрузке набора символов utf8: %s\n", $connection->error);

exit();

}

$qry = $connection->query("SELECT \* FROM pages WHERE ID = $id");

while($row = $qry->fetch\_array(MYSQLI\_BOTH))

{

return $row[2];

}

$qry->close();

$connection->close();

}

function getContains()

{

return '

<ul class="nav">

<li><a href="#">'. getItemMenu(1).'</a></li>

<li><a href="#">'. getItemMenu(2).'</a>

<ul class="sub\_level">

<li><a href="#">'. getItemMenu(3).'</a></li>

<li><a href="#">'. getItemMenu(4).'</a>

<ul class="sub\_sub\_level">

<li><a href="#">'. getItemMenu(5).'</a></li>

<li><a href="#">'. getItemMenu(6).'</a></li>

<li><a href="#">'. getItemMenu(7).'</a></li>

</ul>

</li>

<li><a href="#">'. getItemMenu(8).'</a></li>

<li><a href="#">'. getItemMenu(9).'</a></li>

</ul>

</li>

<li><a href="#">'. getItemMenu(6).'</a></li>

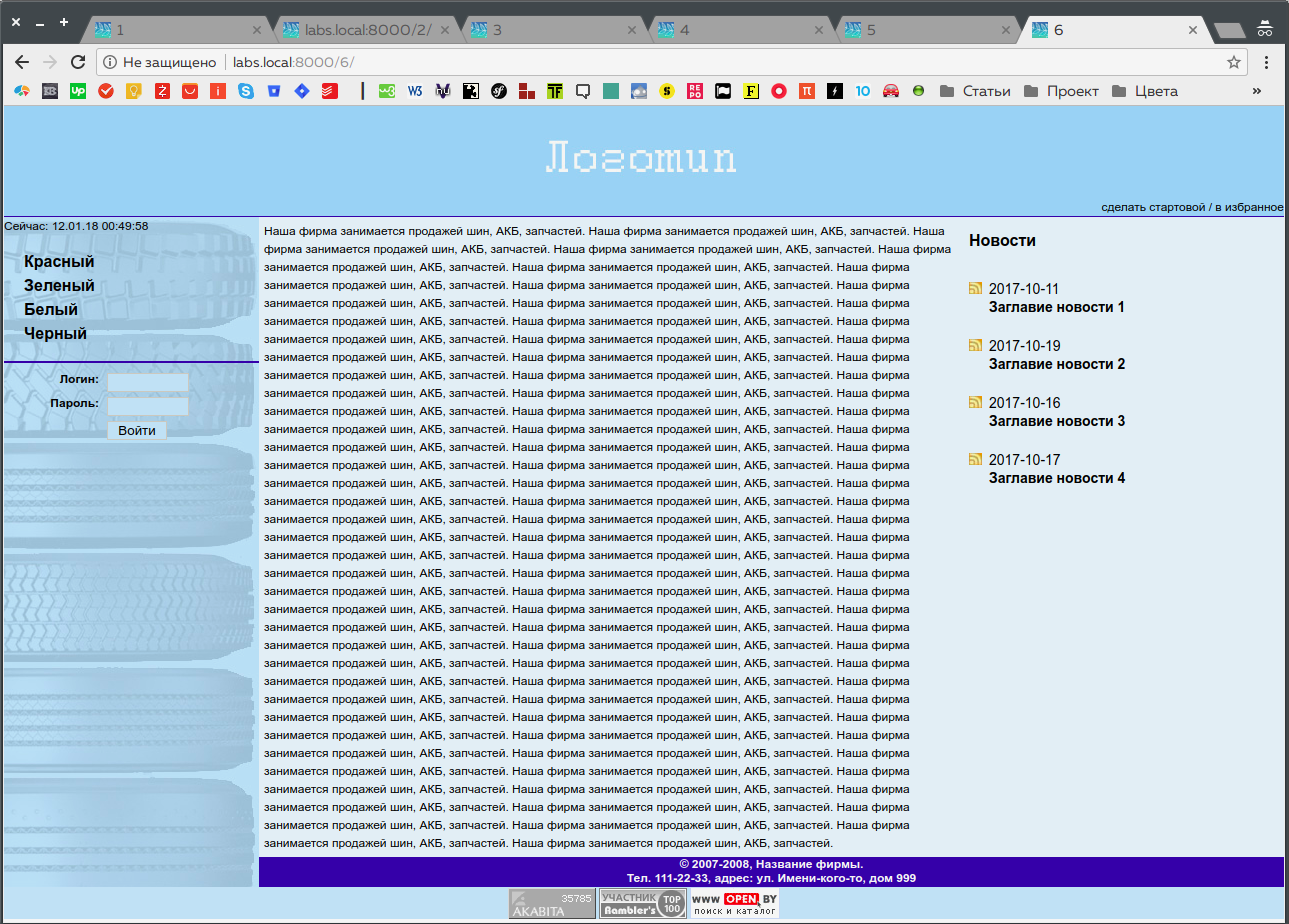
<li><a href="#">'. getItemMenu(7).'</a></li>

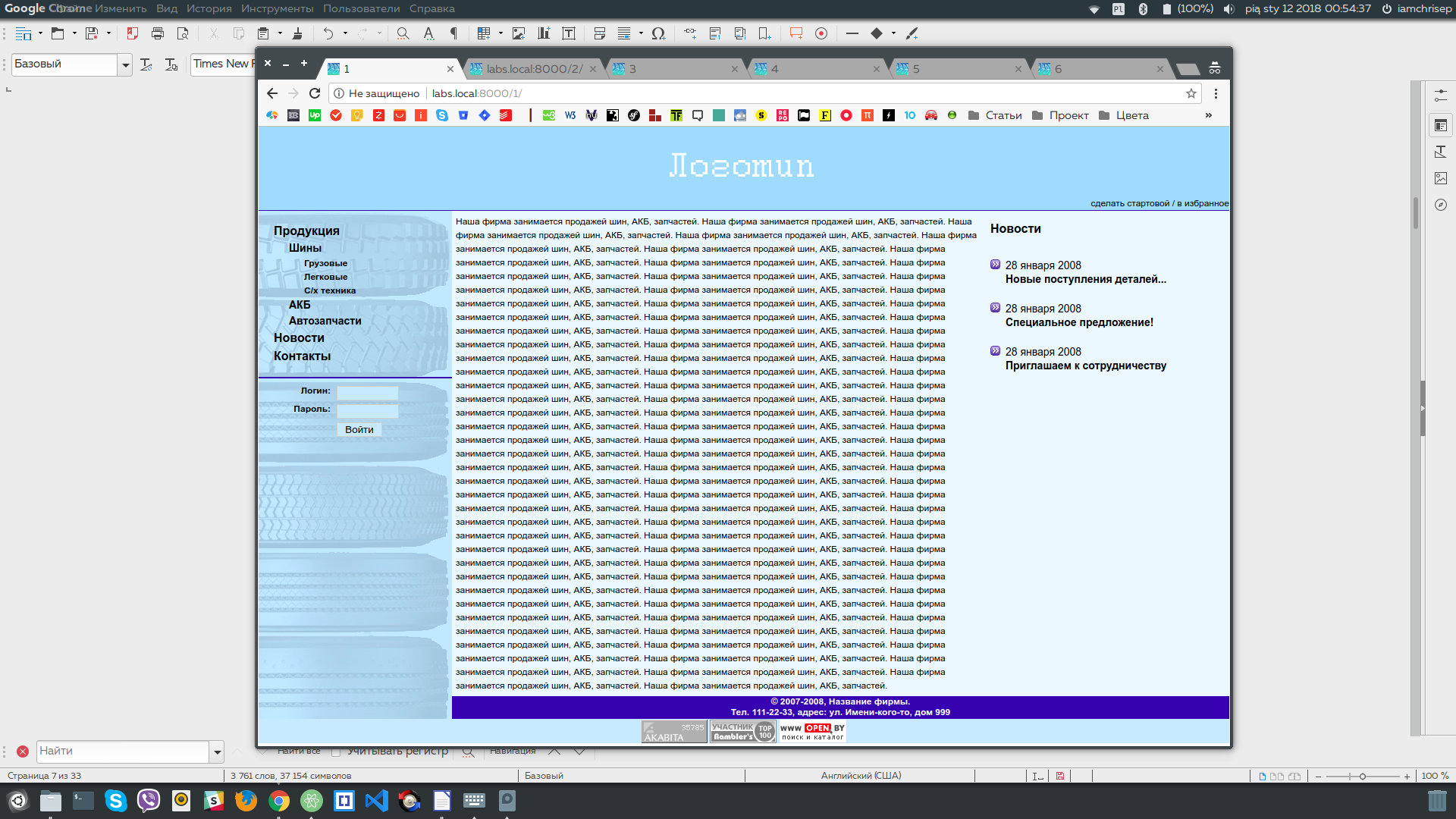
</ul>

';

}

?>

Рисунок 6.1 – Результат выполнения в Ubuntu

Рисунок 6.2 – Результат выполнения в Windows

## Выводы

RSS (Really Simple Syndication) — это распространённый стандарт для экспорта новостных лент.

Особенности. Транслируемые данные хранятся в специальном файле, который расположен на сайте-владельце. Программы-клиенты загружают эти данные в автоматическом режиме, экономя трафик и время: как правило, для загрузки доступны заголовки и краткие аннотации новостей.

Применение. Информация из различных источников, представленная в формате RSS, может быть собрана, обработана и представлена пользователю в удобном для него виде специальными программами-агрегаторами или онлайн-сервисами. Особенно удобно пользоваться RSS, когда речь идет сборе информации из нескольких источников одновременно: достаточно следить за появлением новостей на всех интересующих сайтах, не посещая их. Таким образом, можно существенно сэкономить время, поскольку сбором и доставкой нужных сведений будет заниматься специальная программа.