Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное образовательное учреждение высшего образования
«ТОМСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ СИСТЕМ УПРАВЛЕНИЯ И РАДИОЭЛЕКТРОНИКИ» (ТУСУР)

Кафедра автоматизации обработки информации (АОИ)

РАЗРАБОТКА ИНТЕРНЕТ-ПРИЛОЖЕНИЙ

Методические указания по выполнению лабораторных работ и организации самостоятельной работы для студентов направления «Программная инженерия» (уровень бакалавриата)

2018

Ахтямов Эмиль Камильевич

Семенов Евгений Валериевич

Разработка интернет-приложений: Методические указания по выполнению лабораторных работ и организации самостоятельной работы для студентов направления «Программная инженерия» (уровень бакалавриата) / Э.К. Ахтямов, Е.В. Семенов. – Томск, 2018. – 37 с.

Оглавление

1 Введение

Цели изучения дисциплины «Разработка интернет-приложений» состоят в формировании знаний и практических навыков использования современных сетевых протоколов, проектирования, разработки и тестирования программных приложений, функционирующих в сети Интернет.

В качестве клиентского приложения используется браузер, установленный в операционной системе. Выполняя лабораторные работы, не используются какие-либо специфические элементы разметки или стилей, поэтому ограничений к браузеру не предъявляется.

В качестве текстового редактора используется программа «Notepad++». Скачать последнюю версию и прочитать её описание можно на сайте http://notepad-plus-plus.org/. Можно использовать аналогичную программу. Если в процессе выполнения лабораторных работ понадобятся другие программные продукты, то об этом будет сказано.

2 Содержание лабораторных работ

2.1. Лабораторная работа «Построение интернет-приложения. Разбор запроса пользователя при использовании методов POST и GET»

Цель работы

Создать гостевую книгу с сохранением каждой записи в отдельный файл, с именем и email пользователя, а также реализуем проверку на верность введенного email.

Порядок выполнения лабораторной работы

Создадим гостевую книгу с сохранением каждой записи в отдельный файл, с именем и email пользователя, реализуем проверку на верность введенного email (обязательное наличие знака @ и точки, а также минимум один символ до @, 2 между @ и точкой, и от 2-х до 6 символа после знака точки).

Создайте текстовый файл и переименуйте его в index.php.

Наберите php-код, как показано на рисунке 1.

```php
<?
$action = $_POST["action"];
if (!empty($action)) {
    $name = $_POST["name"];
    $msg = $_POST["msg"];
    $email = $_POST["email"];

    $name = substr($_POST["name"], 0, 50);
    $name = htmlspecialchars($name);
    $email = substr($_POST["email"], 0, 50);
    $email = htmlspecialchars($email);
    $msg = substr($msg, 0, 1024);

    $msg = htmlspecialchars($msg);
    $msg = nl2br($msg);
    $msg = str_replace("\n", " ", $msg);
    $msg = str_replace("\r", " ", $msg);

    $file = fopen("records/rec.".time().".txt", "w");
    fputs($file, $name."\n");
    fputs($file, $email."\n");
    fputs($file, $msg."\n");
    fclose($file);

    print "<HTML><HEAD>\n";
    print "<META HTTP-EQUIV='Refresh' CONTENT='0; URL=index.php'>\n";
    print "</HEAD></HTML>\n";
}

if (empty($action)) {
    ?>
```

Рисунок 1 – Отрывок php-кода

Здесь мы добавляем новую запись в файл (реагируем на нажатие кнопки «Добавить»).

Первая строка говорит, что далее следует php-код, затем мы присваиваем переменной action значение POST-запроса параметра action. Строки с 4-й по 28 выполняются только если выполняется условие не пустоты переменной action (то есть самого POST-запроса).

Строки 4, 5, 6 – получение из POST-запроса имени, почты и текста сообщения пользователя.

8, 10, 12 – обрезаем имя, почту и сообщение до 50, 50 и 1024 символов соответственно.

9, 11, 14 – преобразуем специальные символы в HTML-сущности (своего рода защита, чтобы ликвидировать возможность редактирования кода через сообщение), к примеру знак < заменит на <

nl2br (15-я строка) – перед каждым разрывом строки вставляет символ

16, 17 – удаляем разрывы строк (в результате сообщение будет идти одной строкой с тегами
 в местах разрыва строк).

19 – создаем файл в папке records с именем rec. [время в секундах, прошедшее с 01.01.1970 00:00].txt

20, 21, 22 – записываем в этот файл имя, почту и сообщение, на первую, вторую и третью строки соответственно, затем закрываем файл (23).

25, 26, 27 – обновляем страницу.

Далее если запрос action пуст (30-я строка) (т.е. мы просто зашли на страницу, без отправки POST-запроса), выполнится основной HTML код самой гостевой книги. Заканчиваем php-код (31-я строка), и пишем html

38-52 строки – скрипт для проверки правильности ввода email, вызывается при отпускании (событие onKeyUp) кнопки в поле ввода почты (59-я строка), в качестве параметра передается то, что уже введено в качестве email.

Строка 40 – присваиваем переменной элемент с ID 1 (для удобства обращения к нему, 1 – id кнопки «Добавить»)

Строка 41 – присваиваем переменной re регулярное выражение.

42 – если строка email (наш переданный параметр) соответствует регулярному выражению, делаем кнопку «Добавить» активной и элементу с ID=asd меняем свойство CSS рамки на стандартную, иначе (если не соответствует регулярному выражению) деактивируем кнопку «Добавить» и сделаем красную рамку вокруг поля ввода email.

Рисунок 2 – Продолжение отрывка php-кода

Для обязательного ввода имени и сообщения используется параметр required (строки 56, 58, 60).

Далее для вывода всех ранее добавленных сообщений необходимо организовать цикл (для повторения html-кода вывода одного сообщения).

2.2. Лабораторная работа «Использование гипертекстового препроцессора PHP с web-сервером»

Цель работы

Научиться на примере конкретной системы управления сайтом использовать гипертекстовый препроцессор PHP с web-сервером.

Порядок выполнения лабораторной работы

С точки зрения 1С-Битрикс шаблон – это внешний вид сайта, в котором определяется расположение различных элементов на сайте, художественный стиль и способ отображения страниц. Включается в себя программный html-код, графические элементы, таблицы стилей, дополнительные файлы для отображения контента. Может включать в себя шаблоны компонентов, шаблоны готовых страниц и сниппеты.

Для того, чтобы интегрировать шаблон в CMS, первым делом в папке /bitrix/templates/ раздел, название которого будет использоваться в качестве ID шаблона. Например, MyTemplate. Далее в качестве ID шаблона будет использоваться именно это название. Теперь в папку MyTemplate необходимо перенести общие ресурсы, такие как изображения, css файлы. js файлы, шрифты. Если в вашем шаблоне не используются специфические шрифты, js файлы, то их переносить не нужно.

После подготовительных операций можно перейти к созданию HTML каркаса шаблона. Шаблон состоит из нескольких частей, таких как header, #WORK_AREA# и footer.

По окончанию формирования структуры, у вас должно получиться в папке 3 файла. Header.php - в нем находится шапка шаблона, footer.php – подвал шаблона и Index.php – оставшийся код, т.е. контент шаблона.

Добавление элементов 1С-Битрикс

Для начала нужно добавить подключение пролога и языковых файлов. Это нужно для того, чтобы пользователь, не являющийся администратором сайта, не смог получить доступ к файлам. Это необходимо сделать как в header.php, так и в footer.php:

```
<?if(!defined("B_PROLOG_INCLUDED") ||
B_PROLOG_INCLUDED!==true)die();
IncludeTemplateLangFile(__FILE__);?>
```

Далее заменим текст внутри тега <title> и внутри тега <h1> на вызов функции.

```
<title><? $APPLICATION->ShowTitle(false); ?></title>
<h1><?$APPLICATION->ShowTitle()?></h1>
```

После необходимо задать относительный путь ко всем файлам шаблона (картинкам, css-файлам, js-файлам, шрифтам). Функция <?=SITE_TEMPLATE_PATH?> задает путь к шаблону относительно корня сайта.

Например, в результате можем получить следующий код:

```
<link href="<?=SITE_TEMPLATE_PATH?>/css/styles.css"
rel="stylesheet">
... <a href="/">
    <img src="<?=SITE_TEMPLATE_PATH?>/images/logo.png"
alt="Whitesquare logo">
  </a>
... <img src="<?=SITE_TEMPLATE_PATH?>/images/map.png"
class="img-responsive" alt="Our offices">
... <a href="/">
    <img src="<?=SITE_TEMPLATE_PATH?>/images/footer-
logo.png" alt="Whitesquare logo">
</a>
```

Добавим вызов функции $APPLICATION->ShowHead();, которая служит сразу нескольким целям (в частности динамическому подключению CSS и JS библиотек, объявленных в компонентах в теле страниц с помощью отложенных функций в заголовке, а так же объединяет и сжимает эти библиотеки, если используется соответствующая настройка 1С-Битрикс).

С помощью отложенных функций SetAdditionalCSS и AddHeadScript мы можем всегда подключить CSS и JS в <header>, а так же избежать повторного подключения одинаковых библиотек (это особенно актуально, например для JQuery, если вы подключаете её не централизованно в шаблоне сайта, в соответствующих компонентах и таких компонентов окажется на странице несколько).

Например:

```
<?  $APPLICATION->ShowHead();
    $APPLICATION-
>SetAdditionalCSS(SITE_TEMPLATE_PATH.'/css/styles.css');
    $APPLICATION->SetAdditionalCSS('http://fonts.googleapis.com');
?>
```

Так же необходимо добавить панель инструментов 1С-Битрикс сразу после открытия тега <body> - <?$APPLICATION->ShowPanel();?>

Создание меню с помощью компонента 1С-Битрикс

Для того, чтобы подключить компонент меню, необходимо в панели инструментов, установить режим правки. Для этого в верхнем правом углу переключить режим.

2.3. Лабораторная работа «Создание веб-приложений с помощью «1С-Битрикс: Управление сайтом»

Цель занятия

Получить практический навык создания веб-приложения.

Порядок выполнения лабораторной работы

Для начала, нужно создать инфоблоки, которые будут хранить новости и комментарии к ним.

Перейдите в административную часть сайта, нажав «Администрирование» на панели управления. Выберите пункты левого меню «Контент» – «Инфоблоки» – «Типы инфоблоков», затем нажмите. Нажмите на кнопку «Добавить новый тип».

В поле «Идентификатор (ID)» введите уникальный идентификатор типа инфоблока, например, «news». В поле «Название» в строке русского

языка укажите «Новости», в строке английского «News». Нажмите «Сохранить».

Создайте инфоблок «Новости» для хранения новостей. В дереве «Контент» выберите только что созданный тип инфоблоков и нажмите на кнопку «Добавить инфоблок».

В поле «Символьный код» введите код инфоблока «news», в поле «Название» введите «Новости». Выберите ваш сайт в списке «Сайты». Нажмите «Сохранить».

Аналогично создайте инфоблок «Комментарии» для хранения комментариев к новостям. При создании инфоблока «Комментарии» перейдите на вкладку «Свойства» и добавьте инфоблоку новое свойство «ID новости», тип «Число», код «PARENT_ID». В этом свойстве будет храниться идентификатор новости, к которой привязан комментарий.

Теперь вы можете создать новости и комментарии к ним перейдя к инфоблокам в дереве «Контент» и нажав «Добавить элемент».

У новости укажите название, текст анонса, детальное описание, можете выбрать картинку для анонса и детальную картинку. У комментария укажите только название, текст анонса и ID новости, к которой хотите его привязать.

Следующий шаг – это вывод новостей и комментариев.

Создадим раздел для новостей. Для этого перейдите в публичную часть сайта, выбрав «Сайт» на панели управления. Перейдите на главную страницу сайта, затем нажмите на кнопку «Создать раздел» на панели управления. Укажите заголовок раздела «Новости», оставьте галочку у пункта «Добавить пункт меню». Перейдите в созданный раздел, выбрав пункт меню «Новости».

Выведем новости на главную страницу раздела «Новости». Для этого нажмите на кнопку «Изменить страницу» на панели управления, в появившемся окне, в списке компонентов, найдите компонент «Список

новостей» по адресу «Контент» – «Новости» – «Список новостей». Перенесите компонент в левую область окна. В появившемся окне настройки параметров компонента укажите тип информационного блока «[news] Новости» и код инфомационного блока «Новости». Найдите поле «URL страницы детального просмотра», в него введите «/novosti/detail.php?ID=#ID#», это укажет адрес будущей детальной страницы и то, что на неё требуется передать ID текущей новости. Сохраните изменения.

Страница должна перезагрузиться, теперь на ней виден список добавленных вами новостей.

Создайте страницу для детального просмотра новости. Для этого нажмите «Создать страницу» на панели управления. Укажите заголовок страницы «Детальная», имя файла «detail.php», уберите галочку у пункта «Добавить пункт меню».

Откройте окно изменения детальной страницы и разместите на ней компонент «Новость детально» по адресу «Контент» – «Новости» – «Новость детально». В окне настройки компонента укажите тип информационного блока «[news] Новости», код инфоблока «Новости» и ID новости «={$_REQUEST["ID"]}». Таким образом, компонент будет выводить информацию о той новости, ID которой был передан в запросе. Сохраните изменения.

Страница детального просмотра создана. Вы можете проверить её работу, перейдя в раздел «Новости» и выбрав любую из добавленных вами новостей.

2.4. Лабораторная работа «Настройки форм информационных блоков»

Цель занятия

Оптимизировать процесс создания большого количества элементов информационного блока.

Порядок выполнения лабораторной работы

Создание большого количества элементов информационного блока можно облегчить за счет предварительной настройки формы добавления элементов.

Настройка внешнего вида формы

Форма редактирования/добавления элемента инфоблока может быть изменена и настроена под потребности контент-менеджера сайта. Измененная форма отображается как при добавлении/редактировании элемента в административной части, так и при добавлении/редактировании элемента в публичной части.

При необходимости можно переместить любые поля с любой закладки формы на любые другие закладки, сформировав удобный вид формы для каждого конкретного инфоблока.

Например, можно заранее задать основные параметры так, чтобы добавление товара в каталог можно было произвести быстро и удобно.

Рассмотрим форму добавления элемента инфоблока. Ее внешний вид по умолчанию представлен на рисунке 3.

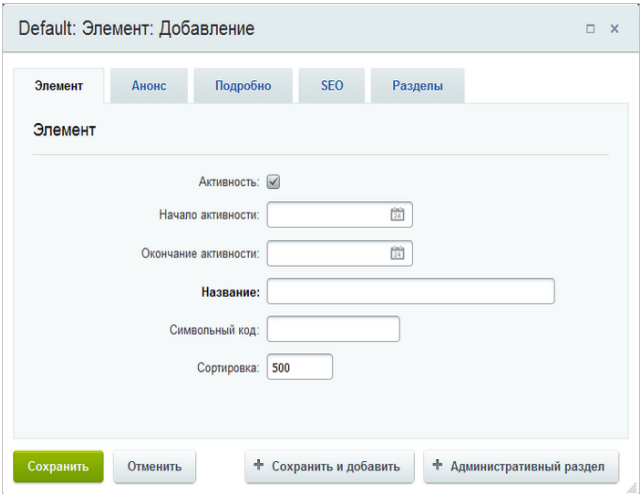

Рисунок 3 – Внешний вид формы добавления элемента инфоблока по умолчанию

Настроенная форма для добавления нового дивана в каталог может выглядеть как показано на рисунке 4.

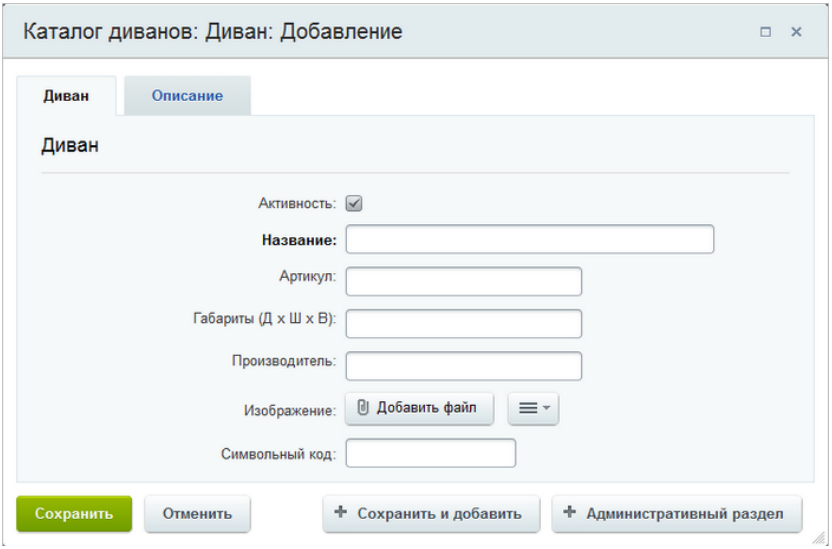

Рисунок 4 – Форма для добавления нового дивана

Как видите, сокращено количество закладок, удалены ненужные поля, перемещены в удобные места нужные поля формы. Работать стало проще и удобнее.

2.5. Лабораторная работа «Верстка под «1С-Битрикс: Управление сайтом» и создание шаблона – особенности, проблемы»

Цель занятия

Ознакомиться с основными особенностями разработки шаблонов на примере конкретной системы управления сайтом.

Порядок выполнения лабораторной работы

Шаблон дизайна - это внешний вид сайта, в котором определяется расположение различных элементов на сайте, художественный стиль и способ отображения страниц. Включает в себя программный html- и php-код, графические элементы, таблицы стилей, дополнительные файлы для отображения контента. Может так же включать в себя шаблоны компонентов, шаблоны готовых страниц и сниппеты.

Использование шаблонов позволяет проводить гибкую настройку дизайна сайтов, разделов и страниц сайта. Например, возможно использование специального праздничного дизайна в течение указанного периода времени, автоматическое управление внешним видом сайта в зависимости от группы посетителей и т.д.

У сайта может быть много шаблонов дизайна, и, наоборот, один шаблон может быть использован на нескольких сайтах.

Шаблон сайта определяет:

оформление сайта (дизайн, верстку страниц, набор основных каскадных стилей);

типы меню и их расположение;

наличие рекламных областей (областей для размещения баннеров);

наличие включаемых областей в шаблоне и страницах сайта;

наличие в дизайне сайта формы авторизации, оформления подписки и т.д.

Все используемые в системе шаблоны хранятся в отдельных папках каталога /bitrix/templates/ (например, /bitrix/templates/demo/ или /bitrix/templates/template1/), либо, начиная с версии 14.0.0, в /local/templates/. Также существует специальная папка .default, которая не является полноценным шаблоном сайта, а содержит шаблоны компонентов и файлы, общие для остальных шаблонов сайта.

Шаблон дизайна сайта обычно состоит из трех основных частей:

Верхней части дизайна (header);

Рабочей области страницы (Work area);

Нижней части дизайна (footer).

Процесс создания шаблона сайта включает два основных этапа:

1 создание прототипа шаблона дизайна;

2 создание полнофункционального шаблона.

Прототип представляет собой сверстанный в html шаблон дизайн сайта. При верстке в шаблоне выделяются функциональные области. Пример такого выделения представлен на рисунке 5.

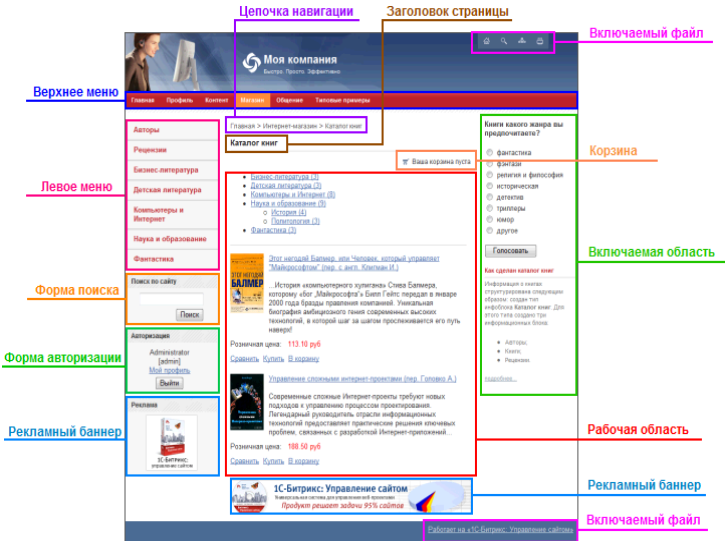

Рисунок 5 – Прототип дизайна сайта

- заголовок страницы;
- меню;
- цепочка навигации;
- форма авторизации;
- форма поиска;
- включаемые области и файлы;
- рекламные области;
- и т.д.

На этапе создания полнофункционального шаблона дизайна выполняется замена HTML элементов дизайна на соответствующие функциональные элементы: программный код и вызовы компонентов. В результате чего уже получается PHP-шаблон дизайна сайта.

Рисунок 6 – PHP-шаблон дизайна сайта

Основные моменты, которые нужно учитывать при создании шаблона:

При подготовке графического дизайна следует заранее разметить линию раздела дизайна на пролог (header.php) и эпилог (footer.php).

Следует выделить основные элементы дизайна, для последующей модификации таблицы стилей: шрифты, цвета заливки и т.п.

Разрабатывая дизайн меню различных уровней, желательно выделять повторяющиеся элементы - для упрощения создания шаблона меню и дальнейшего управления этими меню.

Для облегчения сопровождения различных языковых версий сайта по возможности следует использовать вместо графических элементов текстовые.

При нарезке графического дизайна и подготовке HTML шаблона, необходимо заранее предусмотреть место расположения основных компонентов системы управления сайтом. Выделить области меню, рекламные области, области размещения дополнительных форм.

Рекомендуется производить подготовку шаблона с учетом последующей табличной сборки. Одновременно допускается использование слоев.

При нарезке графического дизайна выделяются однотонные области. При сборке шаблона эти области могут быть представлены ячейками таблиц со сплошной заливкой цвета.

Размещать графические изображения, относящиеся к шаблону, следует в папке /bitrix/templates/<имя_шаблона>/images.

Редактирование шаблона дизайна сайта в визуальном режиме будет происходить корректно, если в атрибутах HTML-тегов не содержится php-код, а также, если, например строки и ячейки таблицы не прерываются php-кодом при формировании таблицы. Если в коде шаблона дизайна сайта есть такие особенности, то редактировать его следует только в режиме кода.

Каскадные стили, используемые в шаблоне, рекомендуется разделять на две таблицы стилей, хранящиеся в двух разных файлах. Оба файла находятся в директории /bitrix/templates/<идентификатор_шаблона>/. Файл styles.css содержит стили для представления информационного содержания страницы на сайте. Файл template_styles.css содержит стили для отображения текстов в самом шаблоне дизайна. При необходимости можно подключить в <head> любое количество стилевых файлов, дополнительно к styles.css и template_styles.css, подключаемым через showhead(). Делается это обычными линками перед закрытием тэга </head>, а дополнительные стилевые файлы положите в любую папку. Эффект будет тот же самый, как если бы вы собрали все ваши дополнительные стили и дописали их в два файла шаблона сайта со стандартными наименованиями.

2.6. Лабораторная работа «Работа с включаемыми и рекламными областями. Применение шаблона дизайна»

Цель работа

Научиться работать с вариантами размещения справочной информации на веб-странице.

Порядок выполнения лабораторной работы

Включаемая область – это специально выделенная область на странице сайта, которую можно редактировать отдельно от основного содержания страницы.

Включаемые области служат для размещения справочной информации, различных форм (подписки, голосования, опросов), новостей и любой другой статической и динамической информации. Также в виде включаемой области могут быть выполнены области с указанием авторских прав, графические ссылки, контактная информация, логотип компании и т.п.

Содержимое включаемых областей хранится в отдельных PHP или HTML файлах. Области для страниц или разделов сохраняются с некоторым суффиксом. Например, в поставляемых файлах продукта в качестве обозначения включаемой области для страницы используется суффикс _inc (например, index_inc.php), а включаемая область для раздела сайта сохраняется в файле с именем sect и добавлением к нему суффикса (например, sect_inc.php).

Файл с включаемой областью должен быть сохранен в той же директории, что и страница, для которой он был создан. Включаемая область для раздела - в папке этого раздела.

Подключение областей в шаблоне дизайна сайта выполняется с помощью компонента Вставка включаемой области либо с помощью функции IncludeFile().

Суффикс, используемый для обозначения включаемых областей, определяется одноименной опцией в настройках компонента Вставка включаемой области. Компонент можно размещать не только в шаблоне

дизайна, но и страницах сайта с условием, что суффикс файла должен быть задан отличным от того, который используется в шаблоне.

Для размещения включаемой области выполните следующее:

Откройте для редактирования шаблон сайта или страницу в визуальном редакторе.

Добавьте компонент Вставка включаемой области (bitrix:main.include) в шаблон сайта (или в тело страницы) и настройте его параметры.

Создание включаемых областей может быть выполнено:

из административного раздела в Менеджере файлов (Контент > Структура сайта > Файлы и папки), создав файл с соответствующим именем;

из публичного раздела сайта в режиме правки. В тех местах, где предполагается вывод включаемых областей, будут показаны иконки для быстрого перехода к созданию этих областей.

После выбора команды Добавить область будет запущен визуальный редактор для создания содержимого включаемой области. При выборе команды Добавить область как PHP станет возможным добавление области в режиме PHP кода

Аналогично перейти к редактированию включаемых областей можно:

- непосредственно из публичного раздела сайта в режиме правки;
- либо из административного раздела, открыв для редактирования соответствующий файл в Менеджере файлов.

Включаемые области создаются на основе шаблонов, хранящихся в папках с именем /page_templates/:

/bitrix/templates/.default/page_templates/ - если данный шаблон включаемой области используется для всех шаблонов дизайна сайта;

/bitrix/templates/<идентификатор шаблона>/page_templates/ - если для шаблона сайта используются отдельные шаблоны включаемых областей.

Чтобы в визуальном редакторе можно было выбирать шаблон, на основе которого создается редактируемая область, список шаблонов для редактируемых областей должен быть добавлен в файл .content.php.

Файл .content.php хранится в папке /page_templates/ в каталоге соответствующего шаблона сайта.

2.7. Лабораторная работа «Создание компонента. Настройка модуля универсального списка»

Цель работы

Создать компонент и настроить модуль универсального списка.

Порядок выполнения лабораторной работы

Компонент - это логически завершённый код, предназначенный для извлечения информации из инфоблоков и других источников и преобразования её в HTML-код для отображения в виде фрагментов web-страниц. Состоит из собственно компонента (контроллер) и шаблона (представление). Компонент, с помощью API одного или нескольких модулей, манипулирует данными. Шаблон компонента выводит данные на страницу.

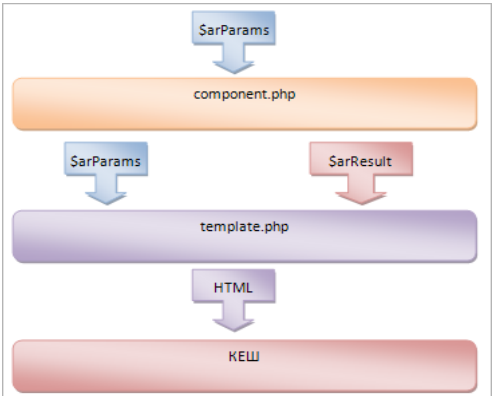

Рисунок 7 – Классическая схема работы компонента

Порядок создания собственно компонента

Выделить необходимый php-код в отдельный файл для того, чтобы использовать его потом в виде вызываемого файла не сложно. Но компонент еще нужно подключить в систему с помощью файла описания, который опознается ядром Bitrix Framework, в результате чего пользователь видит в визуальном редакторе иконку с названием компонента и может настраивать его свойства.

Напомним, что компонент – это выделенный в отдельный файл php-код с законченной функциональностью, файл регистрации компонента в системе и описания его параметров, а также файлы локализации.

Регистрация компонента

Выделение необходимого php-кода в отдельный файл.

Создание файла описания .description.php

Размещение файлов в папке в собственном пространстве имен.

Задание параметров в коде компонента

Локализация

Подготовка файлов с текстовыми константами для компонента и файла регистрации: /lang/ru/<имя_компонента>/component.php и /lang/ru/<имя_компонента>/.description.php

Внесение изменения в код обоих файлов компонента для использования этих констант (подключение файла локализации делается при помощи функции IncludeTemplateLangFile).

Рассмотрим пример создания компонента для сообщений администратору об ошибке.

С помощью этого компонента можно реализовать функционал, который бы позволял пользователям сообщать ответственным за контент о найденной на сайте ошибке. Ошибка будет высылаться почтовым уведомлением. Алгоритм работы пользователя с компонентом очень простой: если пользователь находит на портале ошибку, то он выделяет текст, нажимает Ctrl+Enter и получает форму (Рисунок 8).

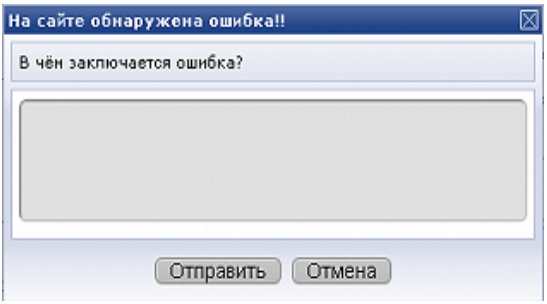

Рисунок 8 – Форма обратной связи в случае обнаружения ошибки

Так как сообщение об ошибке будет отправлено на почту, то потребуется создать новый почтовый тип.

- Перейдите на страницу Настройки > Настройки продукта > Почтовые события > Типы почтовых событий.
- Заполните поля формы:
- Перейдите на страницу Настройки > Настройки продукта > Почтовые события > Почтовые шаблоны.
- Нажмите в контекстной панели на Добавить шаблон, откроется форма создания шаблона.
- Задайте шаблон для созданного типа почтового события.

Создайте в собственном пространстве имен папку feedback.error со следующей структурой:

папка /images
файл feedback.gif
папка /templates
папка /.default
файл script.js
файл template.php
файл .description.php
файл .parameters.php
файл component.php

Файл feedback.gif - иконка, которая будет отображаться в визуальном редакторе.

код файла script.js:

```
BX.bind(document, "keypress", SendError);

function SendError(event, formElem)
{
	event = event || window.event;

	if((event.ctrlKey) && ((event.keyCode == 0xA)||(event.keyCode == 0xD)))
	{
	   var Dialog = new BX.CDialog({
				title: "На сайте обнаружена ошибка!!",
				head: "В чём заключается ошибка?",
				content:    '<form method="POST" id="help_form">\
					<textarea name="error_desc" style="height: 78px; width: 374px;"></textarea>\
					<input type="hidden" name="error_message"value="'+getSelectedText()+'">\
					<input type="hidden" name="error_url" value="'+window.location+'">\
					<input type="hidden" name="error_referer" value="'+document.referrer+'">\
					<input type="hidden" name="error_useragent" value="'+navigator.userAgent+'">\
					<input type="hidden" name="sessid" value="'+BX.bitrix_sessid()+'"></form>',
				resizable: false,
				height: '198',
				width: '400'});

	   Dialog.SetButtons([
	   {
```

```
            'title': 'Отправить',
            'id': 'action_send',
            'name': 'action_send',
            'action': function(){
               BX.ajax.submit(BX("help_form"));
               this.parentWindow.Close();
            }
         },
         {
            'title': 'Отмена',
            'id': 'cancel',
            'name': 'cancel',
            'action': function(){
               this.parentWindow.Close();
            }
         }
         ]);
         Dialog.Show();
      }
}
function getSelectedText(){
  if (window.getSelection){
    txt = window.getSelection();
  }
  else if (document.getSelection) {
    txt = document.getSelection();
  }
  else if (document.selection){
    txt = document.selection.createRange().text;
  }
  else return;
  return txt;
}
```

код файла template.php:

```
<?
CUtil::InitJSCore(array('window', 'ajax'));
?>
```

код файла .description.php:

```
<?
if (!defined("B_PROLOG_INCLUDED") ||
B_PROLOG_INCLUDED!==true) die();

$arComponentDescription = array(
   "NAME" => "Send Error",
   "DESCRIPTION" => "Send Error",
   "ICON" => "/images/feedback.gif",
   "PATH" => array(
      "ID" => "utility",
   ),
);
?>
```

код файла .parameters.php:

```
<?
if (!defined("B_PROLOG_INCLUDED") ||
B_PROLOG_INCLUDED!==true)die();

$arComponentParameters = array();
?>
```

код файла component.php:

```
<?
if (check_bitrix_sessid() && $_SERVER['REQUEST_METHOD'] ==
"POST" && !empty($_REQUEST["error_message"]) &&
!empty($_REQUEST["error_url"]))
{
   $arMailFields = Array();
```

```
        $arMailFields["ERROR_MESSAGE"] = trim
($_REQUEST["error_message"]);
        $arMailFields["ERROR_DESCRIPTION"] = trim
($_REQUEST["error_desc"]);
        $arMailFields["ERROR_URL"] = $_REQUEST["error_url"];
        $arMailFields["ERROR_REFERER"] =
$_REQUEST["error_referer"];
        $arMailFields["ERROR_USERAGENT"] =
$_REQUEST["error_useragent"];


        CEvent::Send("BX", SITE_ID, $arMailFields);
    }
    $this->IncludeComponentTemplate();
    ?>
```

Функция getSelectedText() получает выделенный мышью текст. И далее идет отправка письма в тексте файла component.php

2.8. Лабораторная работа «Перевод сайта на «1С-Битрикс» на технологию композитного сайта»

Цель работы

Создать программный код, который позволит ускорить выдачу веб-страницы пользователю.

Порядок выполнения лабораторной работы

При создании композитного сайта, а также для отладки его работы, необходимо определить в файле dbconn.php константу define("BX_COMPOSITE_DEBUG", true);. В этом случае в лог будут писаться все голосования "против", а также создаваться история изменений страниц в кеше (/bitrix/html_pages/<domain>/) с расширением .delete.<microtime>.

На работающем сайте использование этой константы приведет к увеличению использования дискового пространства. Поэтому её лучше отключить после настройки Композитного сайта.

Основной инструмент работы – это лог, который генерирует функция AddMessage2Log. Место расположения лога определяется настройками dbconn.php:

define("LOG_FILENAME", $_SERVER[«DOCUMENT_ROOT"]."/../log.txt");

В логе можно увидеть список шаблонов, которые голосовали «против» (Рисунок 9).

Рисунок 9 – Фрагмента лога

В примере лога видно, что компонент меню «возражает» против использования его в технологии Композитный сайт.

Открыв шаблон, можно увидеть, что в рамках одной страницы меню не имеет изменяемых частей. Поэтому мы добавляем в шаблон разрешение на использование его в технологии.

Теперь результат работы компонента попадает в статическую страницу, в html файл.

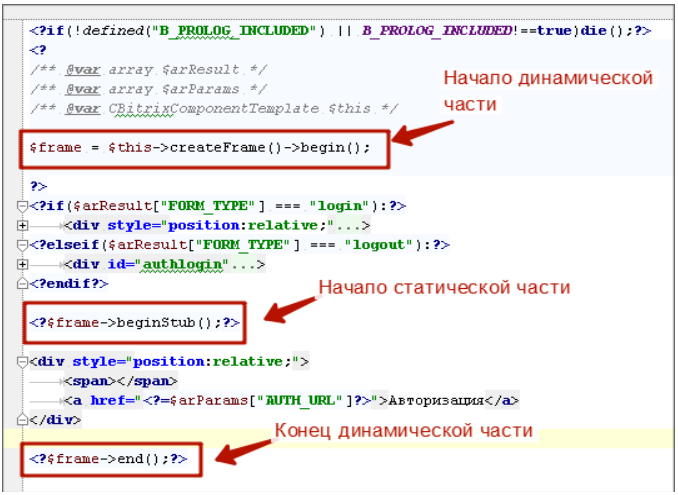

Рисунок 10 – Шаблон компонента Авторизация

Рассмотрим на примере шаблона компонента Авторизация (Рисунок 10).

Анализируя код, отмечаем часть, где меняется содержимое выводимой информации. Мы видим, что если пользователь авторизован, то у него выводится меню - работает одна часть кода, а если не авторизован, то работает другая часть кода. Значит это и есть динамическая часть.

Получается, что всё, что между createFrame()->begin и end() - это динамичная зона. При этом всё что до beginStub() - не пишется в кеш, а досылается после второго запроса, а то что после него - пишется в кеш, выдаётся пользователю и заменяется после второго запроса.

При создании кеша на диске нужно обратить внимание на ограничения файловой системы. (30 тыс файлов в одной папке на UNIX.)Этой проблемы можно избежать, если правильно настроить ЧПУ. Например, делать пути до товаров не /catalog/ID/, а /catalog/SECTION/ID/, тогда система сама разложит все файлы по папкам.

3 Методические указания для организации самостоятельной работы

3.1. Общие положения

Целями самостоятельной работы являются систематизация, расширение и закрепление теоретических знаний в области формирования и производства гипертекстового представления интернет-контента.

Самостоятельная работа студента по дисциплине «Разработка интернет-приложений» включает следующие виды деятельности:

1) проработка лекционного материала;
2) подготовка к лабораторным работам;
3) оформление отчетов по лабораторным работам;
4) подготовка к экзамену.

В ходе самостоятельной работы студент, ориентируясь на изложенные рекомендации, планирует свое время и перечень необходимых работ в зависимости от индивидуальных психофизических особенностей. Формат самостоятельной работы студентов может отличаться в зависимости от формы обучения и объема аудиторной работы.

3.2. Проработка лекционного материала

Для качественного усвоения учебного материала целесообразно осуществлять проработку лекционного материала, которая направлена как на систематизацию имеющегося материала, так и на подготовку к освоению практических аспектов, связанных с содержанием дисциплины.

Проработка лекционного материала включает деятельность, связанную с изучением рекомендуемых преподавателем источников, в которых отражены основные моменты, затрагиваемые в ходе лекций. Кроме того, важное место отведено работе с собственноручно составленным конспектом лекций. При конспектировании во время лекции помните, что не следует записывать все, что говорит и/или демонстрирует лектор: старайтесь выявить главное и записать только это.

Цель конспекта – формирование целостного логически выстроенного взгляда на круг вопросов, затрагиваемых в ходе изучения соответствующей темы, а не механическая фиксация текстовой и графической информации.

Во внеаудиторное время проработка лекционного материала может быть выстроена в двух основных форматах:

а) отработка прослушанной лекции (прочтение конспекта и рекомендованных преподавателем источников с сопоставлением записей) и восполнение пробелов, если они имелись (например, если студент не понял чего-то, не успел записать);

б) прочтение перед каждой последующей лекцией предыдущей, дабы не тратилось много времени на восстановление контекста изучения дисциплины при продолжающейся или связанной теме.

В ходе проработки лекционного материала обращайте внимание на контрольные вопросы, которые, как правило, имеются в конце каждой темы учебника (учебного пособия). Отвечая на них, можно сделать вывод о степени понимания материала. Если ответы на какие-то вопросы вызвали затруднения, то следует предпринять еще одну попытку изучения отдельных вопросов.

3.3. Подготовка к лабораторным работам и оформление отчетов к лабораторным работам

При подготовке к лабораторным занятиям необходимо заранее изучить методические рекомендации по его проведению, обратить внимание на цель, формат и содержание занятия. Если какие-то моменты вызвали дополнительные вопросы, целесообразно обратиться к содержанию лекционного материала, рекомендациям преподавателя по изучению теоретической части курса (рекомендуемым источникам) или за личной консультацией. В ходе подготовки к лабораторным работам может потребоваться обращение к различным источникам. Проявляйте инициативу и самостоятельность в данном вопросе. При этом следует пользоваться только авторитетными изданиями, как печатными, так и электронными.

Темы самостоятельных работ совпадают с темами лабораторных работ. Содержание работы по каждой теме включает три задания (первые два задания выполняются до лабораторной работы, третье – после):

1. Изучение теории, необходимой для выполнения лабораторной работы.

2. Сбор информации по конкретной теме, необходимой для выполнения лабораторной работы. Осуществляется на основе изучения литературных источников (книг, статей в журналах, в сборниках), публикаций в Интернете, законодательных, нормативно-правовых актов, знаний и опыта коллег и знакомых. Источники информации выбираются студентом самостоятельно.

3. Оформление результатов лабораторной работы. Составляется отчет в печатном виде.

Самостоятельная работа «Построение интернет-приложения. Разбор запроса пользователя при использовании методов POST и GET»

1. Изучение методов и принципов построения интернет-приложений.

2. Сбор информации:

- о синтаксисе PHP-кода;

- о синтаксисе запросов в HTTP и HTTPS протоколах.

3. Описание выполненной созданного интернет-приложения, оформление отчета в соответствии с порядком выполнения лабораторной работы «Построение интернет-приложения. Разбор запроса пользователя при использовании методов POST и GET».

Самостоятельная работа «Использование гипертекстового препроцессора PHP с web-сервером»

1. Изучение возможностей взаимодействия гипертекстового препроцессора PHP с web-серверами.

2. Сбор информации о способах взаимодействия PHP и web-серверов.

3. Описание разработанных алгоритмов и функций, оформление отчета в соответствии с порядком выполнения лабораторной работы «Использование гипертекстового препроцессора PHP с web-сервером».

Самостоятельная работа «Создание веб-приложений с помощью «1С-Битрикс: Управление сайтом»

1. Изучение принципов создания веб-приложений в «1С-Битрикс: Управление сайтом».

2. Сбор информации о применении инфоблоков, принципов работы с инфоблоками, порядке создания разделов в «1С-Битрикс: Управление сайтом».

3. Описание созданного интернет-приложения в ходе лабораторной работы, оформление отчета в соответствии с порядком выполнения лабораторной работы «Создание веб-приложений с помощью «1С-Битрикс: Управление сайтом».

Самостоятельная работа «Настройки форм информационных блоков»

1. Изучение возможностей процесса создания большого количества элементов информационных блоков в «1С-Битрикс: Управление сайтом».

2. Сбор информации о возможностях настройки визуального представления форм информационных блоков «1С-Битрикс: Управление сайтом».

3. Описание порядка настройки форм информационных блоков, изученного в ходе лабораторной работы, оформление отчета в соответствии с порядком выполнения лабораторной работы «Настройки форм информационных блоков».

Самостоятельная работа «Верстка под «1С-Битрикс: Управление сайтом» и создание шаблона – особенности, проблемы»

1. Изучение особенностей верстки под «1С-Битрикс: Управление сайтом» и часто возникающих при этом проблем.

2. Сбор информации о шаблонах дизайна, системе и порядке хранения шаблонов в «1С-Битрикс: Управление сайтом».

3. Описание хода лабораторной работы, оформление отчета в соответствии с порядком выполнения лабораторной работы «Верстка под «1С-Битрикс: Управление сайтом» и создание шаблона – особенности, проблемы».

Самостоятельная работа «Работа с включаемыми и рекламными областями. Применение шаблона дизайна»

1. Изучение возможностей включаемых областей, порядке работы с ними.

2. Сбор информации о возможностях применения включаемых областей для решения задач построения интернет-приложений.

3. Описание решения с использованием включаемых областей, разработанного в ходе лабораторной работы, оформление отчета в соответствии с порядком выполнения лабораторной работы «Работа с включаемыми и рекламными областями. Применение шаблона дизайна».

Самостоятельная работа «Создание компонента. Настройка модуля универсального списка»

1. Изучение процесса создания компонента с помощью модуля универсального списка.

2. Сбор информации о порядке создания компонент с использованием сторонних модулей.

3. Описание решения с помощью модуля универсального списка, разработанного на лабораторной работе, оформление отчета в соответствии с порядком выполнения лабораторной работы «Создание компонента. Настройка модуля универсального списка».

Самостоятельная работа «Перевод сайта на «1С-Битрикс» на технологию композитного сайта»

1. Изучение технологии композитного сайта.

2. Сбор информации о технологии композитного сайта.

3. Описание процесса внедрения технологии композитного сайта в интернет-приложение, оформление отчета в соответствии с порядком выполнения лабораторной работы «Перевод сайта на «1С-Битрикс» на технологию композитного сайта».

3.4. Подготовка к экзамену

Подготовка к экзамену осуществляется во время сессии и включает в себя изучение теоретического материала и выполнение практических заданий. Экзаменационный билет содержит теоретические вопросы и практическую задачу, направленную на определение умений применить знания на конкретном примере.

4 Рекомендуемая литература

Рекомендуемые для изучения источники:

Ехлаков, Ю. П. Основы гипертекстового представления интернет-контента: учебное пособие [Электронный ресурс] / Ю. П. Ехлаков, Э. К. Ахтямов. — Томск: ТУСУР, 2017. — 181 с. — Режим доступа: https://edu.tusur.ru/publications/7086

Дубаков, А.А. Сетевое программирование [Электронный ресурс] : учебное пособие / А.А. Дубаков. — Электрон. дан. — Санкт-Петербург : НИУ ИТМО, 2013. — 248 с.: В другом месте, https://e.lanbook.com/book/43580

Александров, Д.В. Инструментальные средства информационного менеджмента. CASE-технологии и распределенные информационные системы [Электронный ресурс] : учебное пособие / Д.В. Александров. — Электрон. дан. — Москва : Финансы и статистика, 2011. — 224 с.: В другом месте, https://e.lanbook.com/book/5306

Юдахин Р. В. Основы программирования на JAVA : Учебное пособие. - Томск : ТУСУР , 2004. - 195 с. (82 экз.)

Одиночкина, С.В. Основы технологий XML [Электронный ресурс] : учебное пособие / С.В. Одиночкина. — Электрон. дан. — Санкт-Петербург : НИУ ИТМО, 2013. — 56 с.: В другом месте, https://e.lanbook.com/book/43573