



1С-Битрикс: Управление сайтом

Быстрый старт разработчика

Как создать простой сайт





Содержание

Введение	3
Глава 1. Как создать простой сайт.....	4
ВЕРСТКА БАЗОВОГО ШАБЛОНА	4
Шаблон сайта и визуальное редактирование	4
Определение количества необходимых шаблонов	5
Редактирование шаблона	5
ИНТЕГРАЦИЯ КОМПОНЕНТОВ	7
Примеры размещения компонентов	8
КАСТОМИЗАЦИЯ ШАБЛОНОВ КОМПОНЕНТОВ	10
Пример редактирования шаблона на основе компонента меню	10
СОЗДАНИЕ СТРУКТУРЫ	14
УПРАВЛЕНИЕ СЛУЖЕБНЫМИ ДАННЫМИ	15
Кодировка страниц и формат отображения дат	15
Управление метаданными	16
Управление заголовком документа	18
ДОБАВЛЕНИЕ ПРОИЗВОЛЬНОГО PHP КОДА.....	19



Введение

Учебный курс для начинающего разработчика. Вы первый раз обращаетесь к популярной платформе Bitrix Framework и не знаете откуда начать? Начните с этого курса.

⚠ Внимание! Курс *Быстрый старт разработчика* - "выжимка" из огромного объема справочной информации по *Bitrix Framework*. Он создан с целью облегчить начинающему разработчику изучение системы и не является ни в коей мере заменой документации и базовых учебных курсов. В нем даются только основы с указанием более детальных источников информации.

При условии качественного изучения материалов курса, по его окончании специалист получит базовые понятия о платформе и ссылки на все, что может вам понадобиться при углубленном изучении *Bitrix Framework*.

Минимальный уровень знаний, необходимый при изучении:

- PHP
- HTML, CSS

⚠ Примечание. Для успешной разработки структуры информационных блоков сложных по структуре проектов, полезно будет получить хотя бы базовые знания по [реляционным базам данных](#).

Список ссылок по теме:

- [PHP](#)
- [HTML, CSS](#)



Глава 1. Как создать простой сайт

Примерный алгоритм действий по созданию простого сайта следующий:

- Получение верстки и ознакомление с техническим заданием на сайт;
- Определение необходимого числа шаблонов и их структуры;
- Установка «1С-Битрикс»;
- Создание шаблонов и применение их к сайту;
- Создание и настройка необходимых элементов шаблона для SEO-продвижения сайта;
- Создание структуры сайта;
- Создание и настройка инфоблоков;
- Создание шаблонов визуальных компонентов;
- Тестирование;

Под простыми сайтами понимаются проекты, функционал которых можно реализовать штатными средствами *Bitrix Framework* с кастомизацией только шаблонов компонентов. Кастомизация шаблона компонента - самый простой и несложный для освоения способ изменения вывода данных компонентов, не требующий серьезного программирования.

В этой главе будут рассмотрены основные шаги по созданию простого сайта.

Верстка базового шаблона

Теоретическое описание шаблона уже давалось [выше](#). В этом уроке описано создание базового шаблона (без размещения компонентов).

Когда дизайн готов, обычно применяется один из двух технологических процессов по интеграции дизайна в систему управления: либо разработчик сам верстает (т.е. переводит из графического эскиза в HTML) макет сайта, либо ему предоставляется уже готовая верстка, и он ее интегрирует в сайт. Вопросы создания верстки не входят в программу обучения **Bitrix Framework**, поэтому речь пойдет о готовом, сверстанном шаблоне.

Шаблон сайта и визуальное редактирование

Bitrix Framework имеет возможность создания шаблона с помощью встроенного визуального редактора. Но этой возможностью пользоваться не рекомендуется. Редактирование шаблона дизайна сайта в визуальном режиме будет происходить некорректно, если:

- в атрибутах HTML-тегов содержится php-код



- если строки и ячейки таблицы прерываются php-кодом при формировании таблицы.

Если в коде шаблона дизайна сайта есть такие особенности, то редактировать его следует только в режиме кода. Также не рекомендуется редактировать шаблон в визуальном редакторе при наличии сложной верстки.

Определение количества необходимых шаблонов

Перед началом работы необходимо определить, сколько различных шаблонов сайта понадобится. Обычно при разработке сайта прорисовываются все различные страницы или основные элементы сайта.

Bitrix Framework позволяет использовать неограниченное число шаблонов и назначать их по разным условиям. Рассмотрим простейший вариант, что на всех этих страницах простого сайта фактически меняется только контентная часть, а дизайн – не изменяется. Исключение составляет главная страница, у которой контентная область устроена по-другому (не содержит заголовка страницы) и разделена на две части. Это можно реализовать как дополнительными условиями в шаблоне сайта, так и созданием двух шаблонов. Рекомендуется использовать дополнительные условия, в этом случае потребуется всего один шаблон сайта.

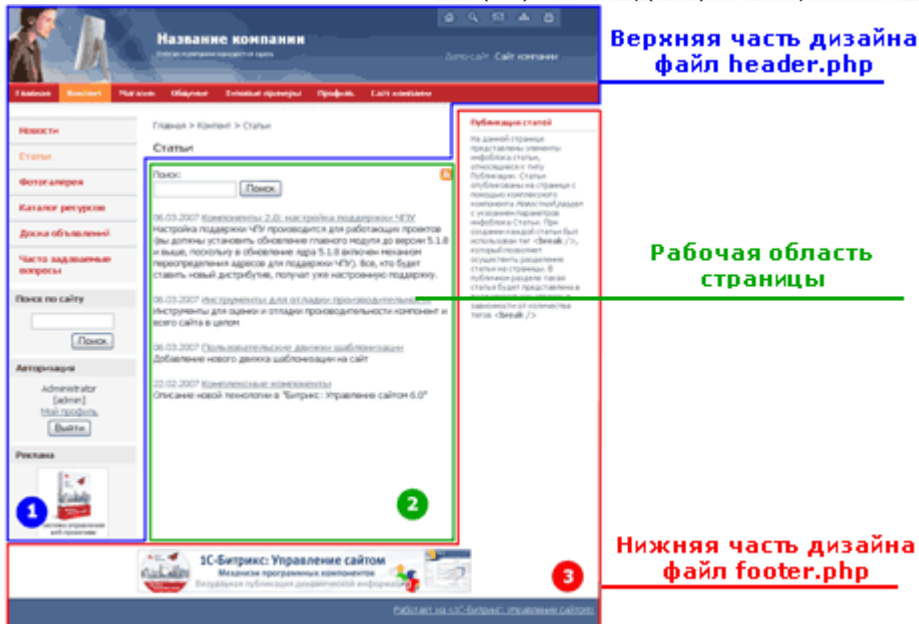
Редактирование шаблона

Перейти к редактированию шаблона можно любым из способов:

- Создав (открыв для редактирования) нужный шаблон (*Настройки > Настройки продукта > Сайты > Шаблоны Сайтов*);
- Выбрав **Шаблон** в меню **Пуск** (*Настройки > Настройки продукта > Сайты > Шаблоны Сайтов*);
- С помощью кнопки **Шаблон сайта** на **Панели управления** (*Шаблон сайта > В Панели управления > Редактировать шаблон*);
- Прямое редактирование файлов **header.php** и **footer.php** в папке шаблона.

Структурное деление шаблона

Проанализируйте шаблон и определите, какая часть кода должна относиться к **Прологу** (файл **header.php**), какая к **Эпилогу** (файл **footer.php**), а какая часть - к **Рабочей области страницы**. Выбранные части кода должны быть размещены в соответствующих файлах, а рабочая область должна быть отмечена тегом `#WORK_AREA#` в шаблоне сайта.



Добавьте соответствующий этим частям код в указанные файлы. Либо, если редактирование происходит в редакторе, разделите их тегом #WORK_AREA#, удалив из шаблона контентную часть.

Служебные директивы

Необходимо заменить некоторые части верстки на служебные директивы **Bitrix Framework** для создания шаблона:

- Заменить подключение стилей и, возможно, javascript файлов на директиву `<?APPLICATION->ShowHead() ?>`
- Заменить прописанный явно заголовок страницы на `<title><?APPLICATION->ShowTitle() ?></title>`
- Сразу после тэга добавить `<?APPLICATION->ShowPanel() ?>`. Если этого не сделать, **Панель управления** не появится
- Перед всеми картинками добавить путь к ним `/bitrix/templates/<? echo SITE_TEMPLATE_ID; ?>/images/`
- Заменить контент на специальный разделитель - #WORK_AREA#

Картинки и файлы стилей

Все изображения, относящиеся к шаблону размещаются в папке **/bitrix/templates/ID шаблона сайта/images/**. Описания стилей из представленной верстки переносятся в файл: **/bitrix/templates/ID шаблона сайта/styles.css**. Описания стилей собственно шаблона переносятся в файл **/bitrix/templates/ID шаблона сайта/template_styles.css**.

Список ссылок по теме:

- [Редактирование шаблона в визуальном редакторе](#) в курсе **Администратор. Базовый**.
- [Управление шаблоном дизайна сайта](#) в курсе **Интеграция**



Интеграция компонентов

Для создания полноценного сайта необходимо интегрировать в шаблон компоненты. Для этого необходимо выделить в дизайне сайта блоки, которые содержат динамическую информацию (вместо них будет размещен вызов компонентов) и блоки, информация в которых должна изменяться пользователем без изменения шаблона (это будут включаемые области).

Пример выделения функциональных областей:



Компоненты, которые должны выполнять функции в этих областях:



В зависимости от того, где должен быть расположен компонент в файле **header.php** или **footer.php**, удаляется html код, отображающий данную функциональную область и вставляется код компонента.

Если шаблон редактируется без визуального редактора, то возможны 2 варианта вставки кода визуальных компонентов:

- Из **Пользовательской документации**. В ней описан формат вызова всех компонентов.
- Вставкой кода в визуальном редакторе на какой-нибудь странице или в новом окне браузера при редактировании любого другого шаблона. Полученный код копируется в нужное место шаблона. Рекомендуется этот способ.

Примеры размещения компонентов

Добавление включаемых областей

Код для вставки компонента **Включаемая область (bitrix:main.include)**, с настройкой на вывод из файла, выглядит так:

```
<?APPLICATION->IncludeComponent (
    "bitrix:main.include",
    "",
    Array(
```



```
"AREA_FILE_SHOW" => "file",

"PATH" => $APPLICATION->GetTemplatePath("include_areas/company_name.php"),

"EDIT_TEMPLATE" => ""

)

);?>
```

В этом примере `include_areas/company_name.php` - файл с включаемой областью:

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

World Book

Все книги мира
```

В файле включаемой области не требуется подключать пролог и эпилог (файлы **header.php** и **footer.php**). Необходимо лишь проверить, что файл включаемой области подключен из системы, а не вызван напрямую. Это делает строка

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

Меню

Для подключения меню необходимо вставить в нужное место шаблона вызов компонента:

```
<?$APPLICATION->IncludeComponent(

    "bitrix:menu",

    "horizontal_multilevel",

    Array(

        "ROOT_MENU_TYPE" => "top",

        "MAX_LEVEL" => "3",

        "CHILD_MENU_TYPE" => "left",

        "USE_EXT" => "Y",

        "MENU_CACHE_TYPE" => "A",

        "MENU_CACHE_TIME" => "3600",

        "MENU_CACHE_USE_GROUPS" => "Y",

        "MENU_CACHE_GET_VARS" => Array()
```



```
)  
);?>
```

Это можно сделать как через визуальный редактор, так и через PHP-код шаблона.

Список ссылок по теме:

- [Пользовательская документация](#) с описанием компонентов.
- [Подключение компонентов](#) в курсе **Интеграция**.

Кастомизация шаблонов компонентов

Кастомизация шаблона компонента, как правило, преследует две цели:

- Приведение формы вывода данных компонента в соответствие с дизайном сайта;
- Организация вывода данных компонента в виде, недоступном в стандартном варианте.

На простых сайтах, как правило, шаблоны требуют кастомизации только при решении первой задачи. Для изменения системного шаблона компонента под конкретный сайт, его необходимо сначала целиком скопировать в папку шаблона сайта. После этого можно перейти к редактированию скопированного шаблона.

Пример редактирования шаблона на основе компонента меню

Выделите в HTML-верстке код, отвечающий за показ верхнего меню. Например:

```
<div class="topmenu">  
    <ul class="topmenu">  
        <li><a href="#" class="first">На главную</a></li>  
        <li><a href="#">Новости</a></li>  
        <li><a href="#" class="selected">Магазины</a></li>  
        <li><a href="#">Книги</a></li>  
        <li><a href="#">Форум</a></li>  
        <li><a href="#">О компании</a></li>  
        <li><a href="#">О Контакты</a></li>  
    </ul>  
</div>
```



В этом коде пункты меню представлены в виде списка, который обладает следующими нюансами:

- У первого пункта меню должен быть указан стиль **first**;
- У выделенного пункта меню должен быть указан стиль **selected**;
- Меню является одноуровневым.

Модифицировать будем код шаблона **Горизонтального многоуровневого меню (horizontal_multilevel)**. Скопируйте шаблон в собственное пространство имен и откройте его для редактирования.

Редактирование шаблона можно проводить как в форме **Bitrix Framework**, так и копированием кода и правкой его в другом редакторе. Использовать другой редактор удобнее в случае объемных текстов, так как форма редактирования в **Bitrix Framework** не поддерживает цветное выделение тегов. Для примера используйте **Notepad++**.

Код меню для этого шаблона выглядит так:

```
1         <?if          (!defined("B_PROLOG_INCLUDED"))          ||
B_PROLOG_INCLUDED!==true)die ();?>
2
3 <?if (!empty($arResult)):?>
4 <div class="gray-tabs-menu">
5     <ul>
6 <?foreach($arResult as $arResult):?>
7
8     <?if ($arResult["PERMISSION"] > "D"):?>
9         <li><a
href="<?=$arResult["LINK"]?>"><nobr><?=$arResult["TEXT"]?></nobr></a>
10     <?endif?>
11
12 <?endforeach?>
13
14     </ul>
15</div>
16<div class="menu-clear-left"></div>
17<?endif?>
```

Рассмотрим каждую строчку этого шаблона:

**Строки шаблона**

1. Проверка, вызван ли этот файл напрямую или нет. Если напрямую – прекратить работу.
3. Если массив с пунктами меню `$arResult` не пуст, то выполнять дальнейшие действия
- 4,5. Внешний блок и начало списка пунктов меню
6. Цикл по массиву с пунктами меню. В `$arResult` – текущий элемент цикла.
- 8-10. Если текущий пользователь обладает правами на просмотр данной страницы, вывести элемент списка с ссылкой на эту страницу. В полях **LINK** и **TEXT** содержится адрес страницы и название пункта меню, соответственно.
12. Конец цикла по массиву с пунктами меню.
- 14,15. Конец списка пунктов меню и конец блока
16. Специальный HTML-тэг, специфичный для использованной верстки
17. Конец условия на наличие пунктов меню (см. строку 3)

Таким образом, шаблон меню содержит:

- область пролога шаблона меню;
- область тела шаблона меню (вывод повторяющихся элементов);
- область эпилога шаблона меню.

После адаптации шаблон примет вид (зеленым цветом выделены изменения):

```
1      <?if          (!defined("B_PROLOG_INCLUDED"))          ||
B_PROLOG_INCLUDED!==true)die ();?>
2
3 <?if (!empty($arResult)):?>
4 <div class="topmenu">
5     <ul class="topmenu">
5a <? $cnt=0; ?>
6 <?foreach($arResult as $arResult):?>
7
```



```
8 <?if ($arItem["PERMISSION"] > "D") :?>
8a <?if ($arItem["SELECTED"]): ?>
8b <li><a href="<?=$arItem["LINK"] ?>"
class="selected"><?=$arItem["TEXT"] ?></a></li>
8c <?else:??>
8d <?if ($cnt==0):?>
8e <li><a href="<?=$arItem["LINK"] ?>"
class="first"><?=$arItem["TEXT"] ?></a></li>
8f <?else:??>
9 <li><a href="<?=$arItem["LINK"] ?>"><?=$arItem["TEXT"] ?></a></li>
10a <?endif??>
10b <?endif??>
10c <?$cnt++; ?>
10 <?endif??>
11
12 <?endforeach??>
13
14 </ul>
15</div>
16
17<?endif??>
```

Итак, что мы сделали?

- В строках 4,5 заменили стили у блока и у списка.
- В строке 5а ввели переменную \$cnt с единственной целью – отследить первый элемент списка – в верстке он задается другим стилем. Эта переменная используется в строке 8d b в строке 10с.
- В строках 8-10 добавили условие проверки активного пункта меню и первого пункта меню.
- И, наконец, удалили специфику верстки из 16 строки.
- Кроме того, у нас нет необходимости в специализированных стилях и картинках для этого шаблона. Поэтому нужно удалить из каталога **/bitrix/templates/default/components/menu/top_menu/** файл **style.css** и папку **/images/**.



⚠ Внимание! Вы можете использовать стили компонента и каталог шаблона компонента для хранения стилей и дополнительных файлов. Это позволит вам переносить шаблон компонента между проектами.

Создание структуры

Создание структуры сайта производится в соответствии с ТЗ на сайт.

На сайте должна быть представлена статическая (о компании, контакты) и динамическая (новости, каталоги, форум) [информация](#).

Создайте требуемую для нас структуру файлов и папок. При создании структуры нельзя забывать про служебные страницы - поиск и карту сайта.

Пример структуры сайта:

Раздел сайта	Каталог/файл	Тип информации	Используемые компоненты
Главная	/index.php	Страница динамической информацией	с bitrix:news.list bitrix:catalog.top
Новости	/news	Страница динамической информацией	с bitrix:news
Магазины	/shops	Статическая страница	нет
Книги	/books/	Страница динамической информацией	с bitrix:catalog
Форум	/forum	Страница динамической информацией	с bitrix:forum
О компании	/about/	Статическая страница	нет
Контакты	/contacts/	Статическая страница	нет
<i>Карта сайта (служебная страница)</i>	/search/map.php	Страница динамической информацией	с bitrix:main.map



Результаты поиска (служебная страница)		/search/index.php	Страница динамической информацией	с	bitrix:search.page
Обработчик ошибки	404	/404.php	Страница динамической информацией	с	bitrix:main.map
обработчик ошибки	500-й	/500.html	Статическая страница (не Framework)	HTML Bitrix нет	

Список ссылок по теме:

- [Создание новых страниц и разделов в интерфейсе Эрмитаж](#) в курсе **Контент-менеджер**
- [Управление структурой проекта](#) в курсе **Интеграция**

Управление служебными данными**Кодировка страниц и формат отображения дат**

Bitrix Framework поддерживает кодировку UTF-8, в которой на одной странице могут сочетаться различные языки – от русского до иероглифов, что позволяет не заботиться о кодировке страниц. Рекомендуем вам разработку сайтов именно в кодировке UTF-8, что позволит вам избежать проблем с настройками однобайтных кодировок.

Настройка кодировки выполняется отдельно для административного и публичного раздела:

- Настройка кодировки, используемой в публичном разделе, выполняется для каждого сайта: *Настройки > Настройки продукта > Сайты > Список сайтов*.

Кодировка задается исходя из языка, используемого на сайте. Также при настройке параметров языка можно задать формат времени и даты, что позволит правильно выводить эти данные в публичном разделе (например, при показе новостей, товаров каталога и т.д.)

Параметры:

*Язык:

*Формат даты:
(например: DD.MM.YYYY)

*Формат даты и времени:
(например: DD.MM.YYYY HH:MI:SS)

*Кодировка:

Название веб-сайта:



⚠ Примечание. В этой форме производится настройка не языка, на котором будет выводиться информация в публичной части сайта, а языка системных сообщений системы, выводимое в публичной части. Например, сообщение: **Элемент не найден или доступ к нему запрещен**, которое выводится при отсутствии элемента инфоблока или запрете его просмотра для данного пользователя.

- Настройка кодировки для административного раздела сайта выполняется через форму управления параметрами языков, используемых в системе: *Настройки > Настройки продукта > Языки интерфейса*.

<input type="checkbox"/>	ID	Акт.	Сорт.	Название	По умолчан.
<input type="checkbox"/>	ru	Да	1	Russian	Да
<input type="checkbox"/>	en	Да	2	English	Нет

- Также при настройке параметров языка можно определить формат времени и даты. Указанный формат будет использоваться при отображении даты и времени в административном разделе сайта.

⚠ Примечание: язык административной части и язык (кодировка) публичной части сайта независимы. Т.е. вы можете создать англоязычный сайт с русским административным интерфейсом.

Управление метаданными

Основной целью использования метаданных является оптимизация сайта для поисковых систем. Поисковые системы используют метаданные для индексации документов сайта.

Примером управления метаданными в продукте может служить механизм задания ключевых слов и описаний для страниц и разделов сайта. По умолчанию в дистрибутиве продукта настроено управление именно этими двумя типами метаданных (по аналогичной схеме список возможных вариантов может быть дополнен).

Набор свойств может быть задан отдельно для каждого сайта, работающего под управлением системы.

Чтобы иметь возможность управлять значениями метаданных, предварительно необходимо создать соответствующие свойства в настройках модуля **Управление структурой** (*Настройки > Настройки продукта > Настройки модулей > Управление структурой*):



Типы свойств:	Тип	Название
	description	Описание
	keywords	Ключевые слова
	title	Дополнительный заголовок
	author	Автор
	not_show_nav_chain	Не показывать навигационную цепь
	adv_desired_target_keyw	Ключевые слова для рекламы (желательно)

⚠ Внимание! Названия типов свойств, используемых для управления метаданными страниц, должны совпадать с названиями мета-тегов в языке HTML. Например типы свойств, ключевые слова и описание, должны совпадать именами (*name*) соответствующих мета-тегов: **keywords** и **description**.

Для вывода значений метаданных в коде страницы нужно воспользоваться функцией `ShowMeta()`, размещаемой в прологе шаблона дизайна сайта:

```
<head>
...
< ?$APPLICATION->ShowMeta("keywords") ?>
< ?$APPLICATION->ShowMeta("description") ?>
...
</head>
```

Пример: Предположим, что для страницы заданы следующие значения свойств ключевые слова и описание:

- Ключевые слова (**keywords**): веб, разработка, программирование
- Описание (**description**): Система управления сайтом

С помощью функции `SetPageProperty()` значения данных свойств будут применены к странице:

```
< ?
$APPLICATION->SetPageProperty("keywords", "веб, разработка,
программирование");
$APPLICATION->SetPageProperty("description", "Система управления
сайтом");
?>
```



Тогда в результате работы функции ShowMeta() или ShowHead() в код страницы будет вставлен следующий HTML-код:

```
<meta name="keywords" content=" веб, разработка, программирование ">
<meta name="description" content=" Система управления сайтом ">
```

Если для самой страницы значение свойства не задано, то будет взято значение свойства вышестоящего раздела (рекурсивно до корня сайта). Если значение свойства не определено, то значение соответствующего мета-тэга останется незадаанным. Свойства страницы могут быть установлены динамически из кода компонентов, расположенных на странице.

Управление заголовком документа

Одним из самых важных элементов сайта на сегодняшний день с позиции SEO-продвижения является заголовок окна браузера (тэг <title>). Практически повсеместно требуется, чтобы заголовок окна браузера отличался от заголовка страницы (тэг <H1>). В «1С-Битрикс» включена поддержка этого требования.

Установка заголовка может производиться как с помощью свойств страницы, так и из визуальных компонент, размещенных на странице. Установка дополнительного заголовка окна веб-браузера осуществляется с помощью зарезервированого в продукте свойства title.

Заголовок страницы в коде может быть установлен следующими способами:

- При редактировании документа с помощью встроенного редактора. В этом случае заголовок страницы задается путем подстановки в код документа следующей функции:

```
<?
$APPLICATION->SetTitle("О компании");
?>
```

- Заголовок документа может устанавливаться динамически одним из компонентов, расположенным на странице. В приведенном ниже примере в качестве заголовка страницы используется имя текущего информационного блока:

```
<?
$arIBlock = GetIBlock($ID, "news")
$APPLICATION->SetTitle($arIBlock["NAME"]);
...
?>
```



- Вывод заголовка документа выполняется с помощью размещения функции `ShowTitle()` в месте показа заголовка страницы:

```
<H1><?$APPLICATION->ShowTitle() ?></H1>
```

Если при выводе заголовка страницы функция `ShowTitle()` использует параметр `false`, это означает, что для установки заголовка страницы не нужно проверять значение свойства `title`.

Вывод заголовка окна веб-браузера выполняется с помощью кода `ShowTitle()`, размещенного в области `<lt;head>>` шаблона дизайна сайта:


```
<head><title><?$APPLICATION->ShowTitle() ?></title></head>
```

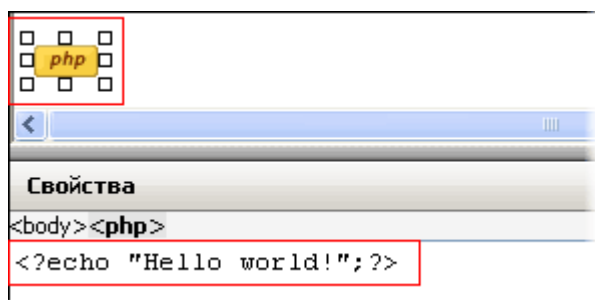
Заголовок окна веб-браузера может быть установлен с использованием различных механизмов. По умолчанию заголовок устанавливается в свойстве страницы `title`. Если значение данного свойства не указано, то заголовок окна браузера будет установлен равным текущему заголовку страницы.

Список ссылок по теме:

- [Свойства страниц и разделов сайта](#) в курсе **Интеграция**
- [Управление служебными данными шаблона](#) в курсе **Интеграция**
- [Управление интерфейсом](#) в курсе **Администратор. Базовый**.

Добавление произвольного PHP кода

В рабочую область страницы также может быть добавлен произвольный PHP-код, который в режиме редактирования в визуальном редакторе будет отображаться в виде значка . Если установить курсор на этот значок, то в панели **Свойства** отобразится непосредственно сам код:



Таким образом, также можно реализовывать дополнительный функционал проекта.

Список ссылок по теме:

- [Работа с PHP скриптом](#) в курсе **Администратор. Базовый**.