



Content Management System



# C-GATOR

УПРАВЛЕНИЕ  
ВЕБ-ПРОЕКТАМИ

РУКОВОДСТВО  
ДИЗАЙНЕРА

 **Деловые программы**

## Содержание

<b>1. Введение</b> .....	<b>3</b>
1.1. Соглашения и обозначения.....	3
<b>2. Основные типы ресурсов</b> .....	<b>4</b>
2.1. Папка .....	4
2.2. Картинка.....	4
2.3. Файл .....	4
2.4. Страница.....	4
2.5. Текстовая страница .....	4
2.6. Текст .....	5
2.7. Шаблон.....	5
<b>3. Шаблоны и стили</b> .....	<b>6</b>
3.1. Шаблоны.....	6
3.1.1. Создание шаблона.....	6
3.1.2. Свойства шаблона .....	7
3.1.3. Применение шаблона.....	7
3.1.4. Удаление шаблона.....	7
3.2. Стили.....	7
3.2.1. Создание стиля .....	8
3.2.2. Применение стиля .....	8
<b>4. Динамический контент и декораторы</b> .....	<b>9</b>
4.1. Декораторы. Общее описание.....	9
4.1.2. Использование переменных .....	9
4.2. Декоратор GetVar .....	11
4.3. Декоратор SetVar.....	12
4.4. Декоратор ItemProperty.....	12
4.5. Декоратор MenuBuilder.....	16
4.6. Декоратор MenuItem .....	18
4.7. Декоратор ItemList.....	20
4.8. Декоратор ItemListPager.....	23
4.9. Декоратор FormPostBack .....	23
4.10. Декоратор ActionButton .....	24
4.11. Декоратор SendEmail.....	25
4.12. Декоратор GoogleSearch .....	26
4.13. Декоратор If .....	28
4.14. Декоратор IfHasRole .....	28
4.15. Декоратор ErrorLabel .....	29
<b>5. Приложения</b> .....	<b>30</b>
5.1. Требования к системе .....	30
5.2. Словарь терминов .....	31
5.3. Предметный указатель .....	33

# 1. Введение





---

Настоящий документ является руководством дизайнера к программному продукту **C-Gator: Управление веб-проектами** (далее «C-Gator» или «система»). Руководство содержит информацию для опытного пользователя, знающего язык разметки HTML и язык таблиц каскадных стилей CSS.

## 1.1. Соглашения и обозначения

Обозначения, используемые в тексте:

---

Свойства	Команды, которые выбираются где-либо в программе, обозначаются полужирным шрифтом
Меню ▶ Пункт	Символ '▶' означает, что команды или пункты меню выбираются одна за другой
Просмотр	Заголовки, которые вы можете видеть где-либо в программе, обозначаются курсивом.
Администратор	При описании или упоминании специальные термины обозначаются курсивом. В Приложении на стр. 31 вы найдете <a href="#">Словарь терминов</a>
Enter	Малые прописные обозначают клавишу.
Ctrl+Enter	Комбинация клавиш записывается через знак '+'. Символом  обозначается разного рода важные замечания
 <b>Внимание</b>	
 <b>Совет</b>	Символом  выделяются пояснения и рекомендации






---

В тексте данного руководства приводятся только русскоязычные названия пунктов меню и элементов управления. Таким образом, мы предполагаем, что пользователь работает в русскоязычной версии Windows и использует русскоязычный интерфейс при работе в C-Gator.

Администратор – одна из важнейших ролей в системе. Он наделен исключительными правами в рамках одного или нескольких доменов, которые поддерживаются системой. Он определяет базовые настройки системы, регистрирует пользователей и назначает им права на совершение тех или иных действий. Администратор системы управляет следующими объектами:

На уровне домена (область **Активный домен** в меню на зеленом фоне):






---

 <b>Пользователи</b>	Учетные записи пользователей системы
 <b>Группы</b>	Наборы пользователей с одинаковыми правами
 <b>Роли</b>	Набор прав пользователей на совершение действий
 <b>Сайты</b>	Задаёт главную страницу для определенного URL
 <b>Иконки</b>	Список иконок, использующихся в системе

---

На уровне системы в целом (область **Конфигурация системы**):

---

 <b>Администраторы системы</b>	Учетные записи администраторов, управляющих системой в целом
 <b>Домены</b>	Основные административные единицы системы
 <b>Имена сайтов</b>	Альтернативные имена сайтов, поддерживаемых системой
 <b>Ошибки</b>	Лог системных ошибок
 <b>Кэш</b>	Показатели динамического КЭШа системы

---

## 2. Основные типы ресурсов

---

C-Gator обладает большой гибкостью в возможности расширения набора доступных типов ресурсов. Существует несколько типов ресурсов, которые входят в базовую конфигурацию системы. Эти типы ресурсов, составляющие основу системы, мы будем называть *эталонными*. Рассмотрим, для чего предназначены и как используются эти типы ресурсов.

Часть из перечисленных ниже типов описана в Руководстве пользователя C-Gator. Здесь мы не будем останавливаться на них подробно.

### 2.1. Папка

Ресурсы типа «Папка» позволяют пользователю структурировать набор ресурсов таким образом, чтобы работать с ним привычным образом — подобно тому, как пользователь работает с деревом каталогов в Проводнике Windows. Папки могут быть вложены в другие папки, другие ресурсы могут размещаться в подпапках или в корне дерева.

Подробнее тип ресурса «Папка» описан в Руководстве пользователя C-Gator.

### 2.2. Картинка

В ресурсе типа «Картинка» хранится графическое изображение формата GIF, JPEG, PNG — для использования на страницах. Подробнее тип ресурса «Картинка» описан в Руководстве пользователя C-Gator.

### 2.3. Файл

Ресурс типа «Файл» предназначен для хранения данных в произвольном формате. Например — документ Word, текст в формате PDF, ZIP-архив, исполняемый EXE-файл и т.д. Подробнее тип ресурса «файл» описан в Руководстве пользователя C-Gator.

### 2.4. Страница

Системой C-Gator изначально поддерживаются три типа ресурсов, предназначенных для хранения и представления текстовой информации: «Страница», «Текстовая страница» и «Текст».

Ресурс типа «Страница» хранит обычные веб-страницы, которые могут редактироваться пользователями — причем как с помощью визуального редактора, так и в виде обычного HTML (на этот режим можно переключиться галочкой **Разметка** в визуальном редакторе). На странице могут использоваться декораторы и они будут учитываться при обработке страницы.

Для страницы, как правило, указывается какой-либо шаблон и таблица стилей, в визуальном редакторе описывается только содержательная часть страницы.

Подробнее тип ресурса «Страница» описан в Руководстве пользователя C-Gator.

### 2.5. Текстовая страница

Ресурс типа «Текстовая страница» в целом аналогичен ресурсу типа «Страница», но текстовую страницу можно редактировать только в режиме HTML-разметки, без возможности переключения в режим визуального редактора. Рекомендуется использовать этот тип ресурса в том случае, когда редактирование в режиме визуального редактора не имеет смысла и/или может нарушить разметку —

например, если большая часть страницы генерируется динамически с помощью декораторов. По этой причине, этот тип ресурса обычно используется для главной страницы сайта.

В процессе отображения «Текст» кэшируется на диске как aspx-файл, что повышает среднюю скорость вывода ресурса такого типа.

Текстовая страница обладает тем же набором дополнительных параметров, что и страница.

## 2.6. Текст

Ресурс типа «Текст» предназначен для хранения текстовой информации, которая, однако, не имеет прямого отображения в HTML-формат готовой страницы, выдаваемой в качестве ответа браузеру клиента. Например, в виде ресурса этого типа могут храниться JavaScript, таблицы стилей CSS, документы XML, шаблоны XSLT, фрагменты HTML для включения в страницы и т.п.

Чтобы задать конкретный тип ресурса «Текст», необходимо при его создании указать на вкладке **дополнительно** один из следующих типов:

- **text/plain** - текст
- **text/html** – текст в формате html
- **text/xml** - текст в формате xml
- **text/css** – стиль CSS
- **text/javascript** - javascript

Ресурс типа «Текст» по умолчанию выдается в формате text/html, если не выставлен отличный от text/html тип содержимого.

На вкладке **дополнительно** для ресурса данного типа можно определить политику кэширования:

- **Public** - файл будет кэшироваться браузером и проху-сервером
- **Private** - файл будет кэшироваться браузером, но не будет кэшироваться проху-сервером
- **Server** - файл будет кэшироваться сервером, на котором установлен C-Gator
- **ServerAndNoCache** - файл будет кэшироваться только сервером, кэширование браузером запрещено
- **ServerAndPrivate** - файл будет кэшироваться и сервером, и браузером
- **NoCache** - запрет на кэширование, файл не будет кэшироваться ни проху-сервером, ни браузером

## 2.7. Шаблон

Если страница содержит только содержательную часть страницы, то шаблон содержит лишь оформление, которое является общим для некоторого набора страниц — например, меню для перехода в другие разделы сайта, панель ссылок, копирайт и т.п. Это оформление описывается в виде HTML-разметки с использованием декораторов. В частности, один из декораторов указывает — в каком месте в этом тексте должен быть вставлен HTML-код содержательной части страницы.

Типу ресурса «шаблон» и особенностям его использования посвящена следующая глава Руководства.

## 3. Шаблоны и стили

---

Система C-Gator позволяет реализовывать дизайн сайта независимо от содержимого (контента). Таким образом, рядовой пользователь C-Gator может работать с контентом сайта, не касаясь дизайна и не рискуя нарушить общее оформление сайта. Дизайнер же, внося изменения в дизайн или даже полностью его меняя, не нарушает содержания сайта.

Реализация дизайна сайта в C-Gator осуществляется посредством шаблонов и стилей. К каждой странице сайта может быть привязан определенный шаблон, задающий правила ее отображения. В свою очередь, к каждому шаблону может быть привязан стиль, описывающий детали дизайна: цвета, шрифты, отступы и т.д.

### 3.1. Шаблоны

При помощи *шаблонов*, а также *стилей*, реализуется несколько важных возможностей системы C-Gator.

Во-первых, шаблоны дают нам способ отделить, абстрагировать содержательную часть сайта — текст новостей, содержание статей и другой контент — от его оформительской части — используемых шрифтов, цветов, способа разметки страницы, графических элементов оформления. Таким образом, содержание сайта описывается отдельно (ресурсами типа «Страница», «Текстовая страница», «Текст»), а его дизайн — отдельно (в виде шаблонов и стилей).

Во-вторых, поскольку дизайн описан отдельно от контента — появляется возможность полностью заменять дизайн, не затрагивая содержательной части — достаточно указать, что в качестве шаблона для страниц нужно использовать другой ресурс — и дизайн сайта будет заменен. Более того, работу над оформительской частью сайта можно вести параллельно работе над его содержательной частью: редактор пишет новую новость, а параллельно с этим дизайнер работает над новым вариантом дизайна — они работают независимо друг от друга и не мешая друг другу.

Шаблон — это документ в формате HTML, содержащий общую разметку страницы. В шаблоне задается, где находится заголовок страницы, главное меню, ссылки на другие разделы сайта, где размещается содержательная часть страницы. Статические части страницы описываются при помощи обычных тэгов HTML. Для описания динамических частей используются *декораторы*.

Декораторы — это специальные тэги, которые не попадают в готовую страницу в браузере, а обрабатываются на сервере — вместо декоратора подставляется какое-то содержимое и/или выполняется некоторое действие — какое, определяется названием и параметрами декоратора. Например, следующий декоратор в ходе обработки шаблона заменяется на содержимое данной страницы:

```
<decorator:PageContent />
```

Как работают шаблоны. Пользователь, используя браузер, выполняет на сайте какое-то действие — например, нажимает на ссылку. В систему C-Gator приходит запрос на получение страницы. Система определяет, что страница привязана к шаблону. Берется текст этого шаблона (а НЕ страницы), выполняется обработка всех тэгов (в том числе, возможно — вставка в нужное место разметки текста самой страницы), полученный таким образом HTML-документ выдается в качестве ответа браузеру — пользователь видит новую страницу.

Подробнее об использовании декораторов читайте на стр. 9.

#### 3.1.1. Создание шаблона

Для создания нового шаблона выполните действия:

1. Выберите в дереве ресурсов библиотеку или папку, в которой вы хотите создать новый шаблон.
2. Перейдите на вкладку **Ресурсы (Список ресурсов** — если вы выбрали папку)
3. Нажмите кнопку **Создать ресурс**
4. В списке **тип ресурса** выберите **шаблон**, нажмите **ОК** для продолжения
5. Задайте свойства ресурса, обязательно укажите **название**, например, «template.regular.html».

### 3.1.2. Свойства шаблона

---

<b>идентификатор (URI) ресурса</b>	Строка, идентифицирующая данный ресурс — для ссылки на него из других ресурсов
<b>название</b>	Название ресурса в дереве ресурсов
<b>идентификатор иконки</b>	Задаёт иконку при изображении данного ресурса в дереве ресурсов
<b>описание</b>	Ваши заметки об этом ресурсе, в произвольной форме
<b>заголовок</b>	
<b>стиль</b>	Идентификатор (URI) ресурса таблицы стилей (CSS), используемой с данным шаблоном
<b>содержимое тега header</b>	
<b>атрибуты тега body</b>	Дополнительные атрибуты для тега <body>

---


### 3.1.3. Применение шаблона

Для применения шаблона выполните действия:

1. Выберите в дереве ресурсов ресурс типа «Страница» или «Текстовая страница».
2. Перейдите на вкладку **дополнительно**.
3. Нажмите на иконку справа от поля **шаблон** и укажите шаблон для использования с данной страницей.
4. Нажмите кнопку **Сохранить** для сохранения изменений.

### 3.1.4. Удаление шаблона

Для удаления шаблона выполните действия:

1. Выберите этот шаблон в дереве ресурсов
2. Выберите вкладку **Свойства**
3. Нажмите на панели инструментов иконку  (**Удалить**).

## 3.2. Стили

Каскадные таблицы стилей — Cascading Style Sheets (CSS) — содержат определения стилей, которые применяются к тэгам HTML и описывают форматирование — например, цвет текста, размер и тип шрифта, выравнивание, границы страниц и т.п. Например, строка

```
H1 {text-align:center; color:red;}
```

говорит о том, что для текста внутри тэга H1 нужно использовать выравнивание по центру и красный цвет текста.

Помимо определения форматирования конкретных тэгов, CSS позволяет определять *классы*, которые можно применить практически к любому тэгу. Например, вот описание класса:

```
.myclass { font: 10pt Arial; }
```

и его применение:

```
<SPAN class="myclass">текст</SPAN>
```

Таким образом, таблицы стилей позволяют отделить описание форматирования текста от самого текста. При грамотном использовании стилей, в тексте не встречается описания форматирования, кроме ссылок на уже определенные стили.

- 📄 Официальную спецификацию CSS можно найти на странице <http://www.w3.org/Style/CSS/>

HTML позволяет определять набор стилей на самой странице — используя тэг <STYLE>. Однако, этот способ не согласуется с принципом отделения контента сайта от его оформления. Мы можем описывать стили внутри *шаблона* — также используя <STYLE>, но как правило это неудобно — например, если мы хотим использовать несколько разных шаблонов с одним набором стилей. Поэтому будет лучше, если стиль представлен в виде отдельного ресурса.

В системе C-Gator таблицы стилей могут быть представлены в виде ресурсов двух типов — «Файл» и «Текст». Первый тип удобнее, если вы редактируете таблицы стилей во внешнем редакторе и загружаете его в систему в виде файла. Ресурс типа «Текст» будет удобнее, если вы редактируете CSS непосредственно в системе C-Gator.

### 3.2.1. Создание стиля

Создание стиля в системе C-Gator достаточно тривиально.

Если вы хотите использовать для стиля ресурс типа «Файл» — просто создайте такой ресурс и загрузите в него CSS-файл, предварительно созданный с помощью стороннего редактора.

Если будет использоваться ресурс типа «Текст» — создайте такой ресурс, перейдите на вкладку **содержание** и опишите на ней нужный набор стилей.

### 3.2.2. Применение стиля

Стиль может быть задан как для отдельной страницы, так и для шаблона. Для применения стиля, выберите в дереве ресурсов страницу или шаблон, перейдите на вкладку **дополнительно** и укажите URI ресурса стиля в поле **стиль**, либо выберите ресурс, нажав на кнопку рядом с полем. После этого нажмите кнопку **Сохранить изменения**.

## 4. Динамический контент и декораторы

### 4.1. Декораторы. Общее описание

Декораторы — это специальные теги, вставляемые в отображаемые поля ресурсов для получения динамического контента. Общий формат записи декораторов выглядит следующим образом:

```
<decorator:DecoratorName parameter="value" runat="server" />
```

Все декораторы должны быть оформлены согласно спецификации XHTML 1.0. Декораторы имеют открывающий и закрывающий теги:

```
<decorator:DecoratorName runat="server"><decorator:DecoratorName />
```

Для декораторов не содержащих вложенные элементы допустимо не указывать закрывающий тег, достаточно в открывающем элементе в конце поставить символ `/`:

```
<decorator:DecoratorName runat="server" />.
```

Атрибуты декоратора, как и названия декораторов нечувствительны к регистру. Значения атрибутов обязательно заключаются в двойные кавычки. Список атрибутов может перечисляться одной строкой или переноситься на несколько:

```
<decorator:DecoratorName parameter1="value1" parameter2="value2"
parameter3="value3" parameter4="value4" parameter5="value5" runat="server" />
```

```
<decorator:DecoratorName
parameter1="value1"
parameter2="value2"
parameter3="value3"
parameter4="value4"
parameter5="value5"
runat="server"
/>
```

Атрибут `runat="server"` является обязательным. Если атрибут не указан, то декоратор не будет обработан.




Передавать данные в декоратор можно несколькими способами:







- используя постоянные значения:  
`<decorator:GetVar name="1024" runat="server" />` — значение 1024;
- указав источник:  
`<decorator:GetVar name="get[id] " runat="server" />` — значение переменной `id` из GET запроса или пустая строка, если параметр не был передан;
- указав источник и значение по умолчанию:  
`<decorator:GetVar name="get[id]=1024" runat="server" />` — значение переменной `id` из GET запроса или 1024, если параметр не был передан.

Для декораторов, которые не осуществляют передачу данных на сервер рекомендуется устанавливать атрибут `EnableViewState="false"`, что позволит уменьшить размер страниц.

#### 4.1.2. Использование переменных

Переменные, используемые в декораторах, могут хранить в различных источниках.

Название	Описание
 Get	Переменные GET запроса. Только чтение.
 Post	Переменные POST запроса. Только чтение.
 Common	Системные переменные. Только чтение.

 Context	Переменные установленные в контексте приложения. Чтение и запись.
 Session	Переменные хранимые в сессии. Чтение и запись.
 Profile	Переменные профиля пользователя. Только чтение.
 Tail	Переменные из строки запроса.
 User	Персональные данные пользователя.
 Params	Переменные всех источников.

Формат строки доступа к переменным выглядит следующим образом:

`common[datetime]`, где

`common` — название источника

`datetime` — имя переменных.

## Переменные GET запроса

Переменные GET берутся из адресной строки браузера. Например, в строке запроса








`http://mysite.ru/news/view.html?id=123&type=all`

доступными переменными являются `id` и `type`.

## Переменные POST запроса

Переменными данного типа являются все параметры переданные формой при отправке с клиента на сервер.

## Системные переменные

Название	Описание
 DateTime	Дата и время <code>common[datetime]:0:dd.MM.yyy</code> , где <code>common</code> — название источника <code>datetime</code> — имя переменной <code>0</code> — часовой пояс <code>dd.MM.yyy</code> — формат отображения даты и времени Более подробно формат даты описан в <b>приложении 1</b> .
 Os	Операционная система.
 Browser	Название браузера.
 ServerIP	IP адрес сервера.
 ClientIP	IP адрес клиента.
 SiteName	Имя сайта.
 DomainID	Идентификатор домена.
 Country	Информация о стране <code>common[country]:code</code> — код страны (RU). <code>common[country]:name</code> — название страны (Russia).
 referer	Реферер страницы.

## Переменные установленные в контексте приложения

Областью видимости переменных данного источника является контекст текущей страницы. При переходе на другую страницу переменная будет потеряна. Для того, чтобы сохранять значение переменной при переходе между страницами следует пользоваться переменными хранимыми в сессии.

## Переменные, хранимые в сессии

Переменные в сессии доступны на всех страницах открытых в окне браузера и других дочерних окнах.

## Переменные профиля пользователя

Источник позволяет получить данные из расширенного профиля авторизованного пользователя.

Формат записи доступа к параметрам имеет следующий вид:

```
profile[group name]:param, где  
group name — имя группы  
param — название параметра
```

## Переменные из строки запроса

Параметры, передаваемые в строке запроса, имеющими вид:  
`http://mysite.ru/news.html/_t_/category=1/date=10.06.2005`  
доступны через источник `tail`.

Например:

```
tail[category]  
tail[date]
```

## Персональные данные пользователя

Возможен доступ к следующим параметрам:

```
login — логин пользователя  
firstname — имя  
lastname — фамилия  
shortname — короткое имя  
culture — локализация  
email — e-mail
```



## Переменные всех источников

Данный источник позволяет получать доступ к переменным сразу из нескольких источников в следующем порядке: `context`, `session`, `get`, `post`, `tail`.  
Возвращается первое найденное значение переменной из цепочки источников.


## 4.2. Декоратор GetVar

Декоратор **GetVar** позволяет получить значение переменной установленной методами GET и POST или из массива глобальных переменных контекста страницы.

### Атрибуты

Название	Описание
 <b>Runat</b>	Устанавливает признак того, что декоратор должен обрабатываться на сервере. Обязательный атрибут, значение которого должно быть установлено "server".
 <b>Name</b>	Устанавливает имя переменной, значение которой будет возвращено в результате работы декоратора.

### Шаблоны

Название	Описание
 <b>ErrorTemplate</b>	Шаблон для отображения ошибки. В теле шаблона можно вставлять произвольный текст и декораторы. Описание возникшей ошибки можно получить декоратором <b>ErrorLabel</b> . Если шаблон не указан — выводится стандартная системная

ошибка, при условии, что в свойствах сайта установлен параметр, позволяющий отображать ошибки.

## Примеры использования

Пример 1: Получение переменной из GET запроса

```
<decorator:GetVar runat="server" name="get [var]" />
```




Если строка запроса имеет вид `/index.html?var=1024`, то декоратор выдаст значение **1024**.

- Значение, установленное при помощи декоратора **SetVar**, может быть получено в дальнейшем декоратором **GetVar**, но только в контексте текущей страницы.


## 4.3. Декоратор SetVar

Декоратор SetVar устанавливает переменную и ее значение в массив глобальных переменных.

### Атрибуты

Название	Описание
 <b>Runat</b>	Устанавливает признак того, что декоратор должен обрабатываться на сервере. Обязательный атрибут, значение которого должно быть установлено "server".
 <b>Name</b>	Устанавливает имя переменной.
 <b>Value</b>	Устанавливает значение для переменной <b>Name</b> .

### Шаблоны

Название	Описание
 <b>ValueTemplate</b>	Шаблон для задания значения переменной. Внутри шаблона можно использовать другие декораторы.

## Примеры использования

Пример 1: Установка значения переменной

```
<decorator:GetVar runat="server" name="get [var]" />
```

Если строка запроса имеет вид `/index.html?var=1024`, то декоратор выдаст значение **1024**.

Если вставить декоратор

```
<decorator:SetVar runat="server" name="context [var]" value="2048" />
```

и затем снова получить значение переменной:

```
<decorator:GetVar runat="server" name="context [var]" />
```

то теперь оно будет **2048**.












- Значение установленное при помощи декоратора **SetVar** может быть получено в дальнейшем декоратором **GetVar**, но только в контексте текущей страницы.

## 4.4. Декоратор ItemProperty

Декоратор ItemProperty позволяет получать или устанавливать значения для свойств ресурсов. Наиболее частое применение декоратора — вставка контента других страниц.




### Атрибуты

Название	Описание
----------	----------














 <b>Runat</b>	Устанавливает признак того, что декоратор должен обрабатываться на сервере. Обязательный атрибут, значение которого должно быть установлено "server".
 <b>URI</b>	Устанавливает или возвращает Uri отображаемого ресурса приложения C-Gator. Если параметр не указан, то по умолчанию используется Uri из контекста текущей страницы (например, URL запрашиваемой страницы в строке браузера). В шаблоне для получения свойств текущей страницы необходимо размещать декоратор без указания атрибута <b>Uri</b> .
 <b>ItemID</b>	Устанавливает или возвращает код записи множественных свойств отображаемого ресурса приложения C-Gator.
 <b>FieldName</b>	Устанавливает или возвращает имя поля в записи множественных свойств или в самом отображаемом ресурсе приложения C-Gator. Более подробно значения поля описаны ниже.
 <b>FieldValue</b>	Устанавливает или возвращает значение поля в записи множественных свойств или в самом отображаемом ресурсе приложения C-Gator.
 <b>XML</b>	Устанавливает или возвращает признак того надо ли выводить значение поля в виде XML в записи множественных свойств или в самом отображаемом ресурсе приложения C-Gator. Если указан атрибут <b>FieldName</b> , то результатом работы декоратора будет XML строка, содержащая только одно поле в формате <code>&lt;имя поля&gt;содержимое поля&lt;/имя поля&gt;</code> . При пустом значении <b>FieldName</b> (или атрибут не указан) в XML формате будет выдан список всех полей. Поле может принимать значение <i>true</i> или <i>false</i> .
 <b>XSLT</b>	Устанавливает или возвращает строку со ссылкой на ресурс. Используется, если значение поля отображается в виде XML в записи множественных свойств или в самом отображаемом ресурсе приложения C-Gator. XSLT преобразование может выполняться как при указании атрибута <b>XML</b> , так и без него. В этом случае XSLT трансформация будет применена к содержимому поля <b>FieldName</b> , данные которого соответствуют XML формату.
 <b>FieldRenderName</b>	Устанавливает или возвращает название объекта отвечающего за форму отображения информации. Значения, которые может принимать поле, описаны ниже. Форма отображения значения поля должна быть совместима с его типом, например поле типа Boolean может быть отображено с одним из рендеров: <i>FieldBooleanRender</i> , <i>FieldIntegerRender</i> , <i>FieldTextRender</i> , <i>FieldChoiceRender</i> .
 <b>DateTimeFormat</b>	Устанавливает или возвращает формат отображения поля, содержащего дату и время. Формат задается только для режима (свойство <b>Mode</b> ="View"). Для режима "Edit" используется единый внутри системы формат отображения даты и времени. Подробно о формате даты и времени описано в <b>приложении 1</b> .
 <b>Mode</b>	Устанавливает или возвращает режим отображения информации. Возможны следующие значения параметра: <b>View</b> и <b>Edit</b> . При установленном значении <b>View</b> (по умолчанию) декоратор отображает запрашиваемое значение в режиме только для чтения; при <b>Edit</b> — в режиме для редактирования.
 <b>Source</b>	Устанавливает или возвращает строку с названием объекта источника данных для отображения. Существуют следующие

	источники: " <b>resource</b> ", " <b>properties</b> ", " <b>multiproperties</b> ". По умолчанию — " <b>properties</b> ".
 <b>CssClass</b>	Устанавливает или возвращает строку с названием класса стиля для отображения поля данных.






## Шаблоны






Название	Описание
 <b>ErrorTemplate</b>	Шаблон для отображения ошибки. В теле шаблона можно вставлять произвольный текст и декораторы. Описание возникшей ошибки можно получить декоратором <b>ErrorLabel</b> . Если шаблон не указан — выводится стандартная системная ошибка, при условии, что в свойствах сайта установлен параметр, позволяющий отображать ошибки.
 <b>ValidatorTemplate</b>	Шаблон для вставки валидаторов. Подробное описание валидаторов находится в <b>приложении 2</b> .
 <b>ValueTemplate</b>	Шаблон для вставки значения. Используется только если атрибут <b>Mode="edit"</b> .

## Значения поля FieldRenderName

Значение	Описание
 <b>FieldBooleanRender</b>	Логическое выражение.
 <b>FieldCurrencyRender</b>	Представление валюты.
 <b>FieldDateTimeRender</b>	Дата и время.
 <b>FieldIntegerRender</b>	Целочисленное значение.
 <b>FieldNoteRender</b>	Текстовый блок, представленный в виде текстового поля ввода или WYSIWYG редактора.
 <b>FieldNumberRender</b>	Числовое значение.
 <b>FieldTextRender</b>	Текстовая строка.
 <b>FieldChoiceRender</b>	Выбор из списка.
 <b>FieldMultiChoiceRender</b>	Множественный выбор из списка.
 <b>FieldURLRender</b>	Редактор связанного URL ресурса.
 <b>FieldUserRender</b>	Редактор связанного пользователя.
 <b>FieldFileRender</b>	Редактор связанного файла.
 <b>FieldLookupRender</b>	Ссылка на поле ресурса.

## Значения поля fieldName

Значение	Описание
<b>Source="resource"</b>	
 <b>uriname</b>	Название ресурса, обязательное поле ресурса. Хранится только в латинском алфавите.
 <b>description</b>	Описание.
 <b>uri</b>	Полный uri ресурса, однозначно идентифицирующий ресурс в библиотеке.
 <b>id</b>	Код ресурса в формате Guid.
 <b>type</b>	Тип ресурса.

 status	Статус ресурса в диапазоне {На редактировании, Отправлено на доработку, Утверждено, Опубликовано}.
 status_comment	Список комментариев (вводятся при изменении статуса ресурса и хранятся в формате XML).
<b>Source="properties"</b>	
Поля определяются явно через библиотеку	
<b>Source="multiproperties"</b>	
 rmp_property_id	Код множественного свойства.
 status	Статус множественного свойства в диапазоне {На редактировании, Отправлено на доработку, Утверждено, Опубликовано}.
 status_comment	Список комментариев (вводятся при изменении статуса ресурса и хранятся в формате XML).

## Примеры использования

### Пример 1: Значение свойств

```
<decorator:ItemProperty Runat="server" URI="/common/menu.html"
FieldName="content"></decorator:ItemProperty>
```

В данном примере результатом работы будет содержимое поля content страницы /common/menu.html.

### Пример 2: Использование декоратора в атрибутах и скриптах

Результат работы декоратора можно использовать в атрибутах HTML тэгов или присваивать как значение скриптам, при этом декоратор должен быть заключен в одинарные кавычки, как в приведенных ниже примерах.

```
<a href='<decorator:ItemProperty Runat="server" FieldName="uri" />'>link</a>
```

```
<script>
var uri='<decorator:ItemProperty Runat="server" FieldName="uri" />';
</script>
```

### Пример 3: Получение данных в XML формате

Получить данные поля в XML формате можно таким образом:

```
<decorator:ItemProperty Runat="server" FieldName="title" uri="/test.html"
XML="true" />
```

Результатом работы этого примера будет XML строка

```
<item>
  <field name="title" title="заголовок" type="Text" />
</item>
```

### Пример 4: Изменение вида отображения

Для получения всех свойств ресурса достаточно исключить атрибут **FieldName**:

```
<decorator:ItemProperty Runat="server" uri="/test.html" XML="true" />
```

XML строка в этом случае будет приблизительно такой (зависит от набора полей типа ресурса):

```
<item>
  <field name="title" title="заголовок" type="Text" />
  <field name="cachepolicy" title="политика кэширования"
type="Choice">Client</field>
  <field name="style" title="стиль" type="Url" />
  <field name="template" title="шаблон" type="Url" />
  <field name="createForm" title="создавать форму"
type="Boolean">Нет</field>
  <field name="sslonly" title="доступ только через SSL"
type="Boolean">Нет</field>
```

```

        <field name="navigation" title="участие в навигации"
type="Boolean">Нет</field>
        <field name="redirectpage" title="редирект на страницу" type="Url" />
        <field name="contentplain" title="контент" type="Note">
<![CDATA[Hello World!!!]]>
</field>
        <field name="empty" title="пустое поле" type="Text" />
</item>

```

Указав атрибут XSLT возможно преобразовать данные и вывести в необходимом формате.

Установка атрибута **FieldRenderName** позволяет изменить вид отображения данных. Например, имеется поле типа Boolean, значение которого не установлено, т.е. равно "Нет". При использовании атрибута **FieldRenderName** можно получить следующие значения:

*FieldBooleanRender* — Нет

*FieldIntegerRender* — 0







*FieldTextRender* — False

- ✎ Недопустимо использование декоратора в свойстве ресурса, которое отображает содержимое этого же свойства этого же ресурса — это вызовет заикливание, обработка декоратора будет прекращена. Например, при размещении в поле content декоратора `<decorator:ItemProperty Runat="server" FieldName="content" />` отображение страницы будет приостановлено.

## 4.5. Декоратор MenuBuilder


Декоратор **MenuBuilder** позволяет получать структуру ресурсов в древовидной структуре и отображать ее на странице. Наиболее частое применение декоратора — отображение меню или карты сайта.

### Атрибуты


Название	Описание
 Runat	Устанавливает признак того, что декоратор должен обрабатываться на сервере. Обязательный атрибут, значение которого должно быть установлено "server".
 RootURI	Устанавливает стартовый Uri узла, относительно которого будет производиться выборка ресурсов. Если атрибут не указан или имеет пустое значение — по умолчанию устанавливается "/".
 QueryId	Устанавливает идентификатор запроса, используемый для выборки ресурсов.
 LibraryId	Устанавливает библиотеку, в которой будет производиться поиск ресурсов согласно условиям, установленным в запросе.
 XML	Устанавливает признак того надо ли выводить значение поля в виде XML в записи множественных свойств или в самом отображаемом ресурсе приложения C-Gator. Если указан атрибут <b>FieldName</b> , то результатом работы декоратора будет XML строка, содержащая только одно поле в формате <code>&lt;имя поля&gt;содержимое поля&lt;/имя поля&gt;</code> . При пустом значении <b>FieldName</b> (или атрибут не указан) в XML формате будет выдан список всех полей. Поле может принимать значение <i>true</i> или <i>false</i> .
 XSLT	Устанавливает строку со ссылкой на XSLT ресурс. Используется, если значение поля отображается в виде XML в записи множественных свойств или в самом отображаемом ресурсе приложения C-Gator. XSLT преобразование может

	выполняться как при указании атрибута <b>XML</b> , так и без него. В этом случае XSLT трансформация будет применена к содержимому поля <b>FieldName</b> , данные которого соответствуют XML формату.
 Depth	Устанавливает глубину поиска ресурсов относительно RootUri дальше которого выборка производится не будет. Значение атрибута может быть только целочисленным значением.
 Expand	Устанавливает поведение для отображения активных и неактивных веток дерева ресурсов. Атрибут принимает два значения <i>active</i> (по умолчанию) или <i>all</i> . В первом случае раскрываться будет только активная ветка дерева, во втором — все ветки.
 PageInfo	Устанавливает параметры для ограничения количества выводимых результатов.

## Шаблоны

Название	Описание
 ErrorTemplate	Шаблон для отображения ошибки. В теле шаблона можно вставлять произвольный текст и декораторы. Описание возникшей ошибки можно получить декоратором <b>ErrorLabel</b> . Если шаблон не указан — выводится стандартная системная ошибка, при условии, что в свойствах сайта установлен параметр, позволяющий отображать ошибки.

## Вложенные декораторы

Название	Описание
 MenuItem	Декоратор, отвечающий за отображение элементов в указанном уровне вложенности. Подробнее см. описание декоратора <b>MenuItem</b> . Для каждого уровня вложенности дерева ресурсов необходимо задавать свой декоратор <b>MenuItem</b> определяющий параметры отображения.

## Примеры использования

### Пример 1: Использование декоратора для построения меню

```
<decorator:MenuBuilder libraryID="e33d9061-da97-4d58-b003-ef8136ae183a"
queryId="d08171f3-2d50-47b9-beeb-37efd4fa6ba9" runat="server" depth="2"
expand="all">
<ErrorTemplate>Произошла ошибка!</ErrorTemplate>
<decorator:MenuItem level="1">
<active>
<p><a class="active" href='<decorator:ItemProperty runat="server"
source="resource" FieldName="uri" />'><decorator:ItemProperty runat="server"
source="properties" FieldName="title" /></a></p>
</active>
<inactive>
<p><a href='<decorator:ItemProperty runat="server" source="resource"
FieldName="uri" />'><decorator:ItemProperty runat="server" source="properties"
FieldName="title" /></a></p>
</inactive>
</decorator:MenuItem>
<decorator:MenuItem level="2">
<active>
<p>&#151; <a class="active" href='<decorator:ItemProperty runat="server"
source="resource" FieldName="uri" />'><decorator:ItemProperty runat="server"
source="properties" FieldName="title" /></a></p>
</active>
<inactive>
```

```

<p>&#151; <a href='<decorator:ItemProperty runat="server" source="resource"
FieldName="uri" />'><decorator:ItemProperty runat="server" source="properties"
FieldName="title" /></a></p>
</inactive>
</decorator:MenuItem>
</decorator:MenuBuilder>

```

В этом примере будет отображено дерево ресурсов по условия, указанным в запросе начиная с узла "/". При этом развернуты будут все узлы дерева, поскольку атрибут **Expand="all"**. Глубина отображения дерева ресурсов — 2.

#### Пример 2: Построение пути по сайту

```

<decorator:MenuBuilder libraryID="e33d9061-da97-4d58-b003-ef8136ae183a"
queryId="d08171f3-2d50-47b9-beeb-37efd4fa6ba9" runat="server" depth="3"
expand="active">
<decorator:MenuItem level="1">
<active>
&nbsp;/&nbsp;/&nbsp;/&nbsp;<a href='<decorator:ItemProperty runat="server"
source="resource" FieldName="uri" />'><decorator:ItemProperty runat="server"
source="properties" FieldName="title" /></a>
</active>
</decorator:MenuItem>
<decorator:MenuItem level="2">
<active>
&nbsp;/&nbsp;/&nbsp;/&nbsp;<a href='<decorator:ItemProperty runat="server"
source="resource" FieldName="uri" />'><decorator:ItemProperty runat="server"
source="properties" FieldName="title" /></a>
</active>
</decorator:MenuItem>
<decorator:MenuItem level="3">
<active>
&nbsp;/&nbsp;/&nbsp;/&nbsp;<a href='<decorator:ItemProperty runat="server"
source="resource" FieldName="uri" />'><decorator:ItemProperty runat="server"
source="properties" FieldName="title" /></a>
</active>
</decorator:MenuItem>
</decorator:MenuBuilder>

```

Результатом работы данного примера будет строка приблизительно следующего вида:



#### [Главная](#) / [О компании](#) / [Контактная информация](#)

- ✎ Если атрибуту **Depth** присвоено значение, большее, чем максимальное значение **Level** в декораторе **MenuItem** будут отображаться только те уровни, для которых прописаны декораторы **MenuItem**. Необязательно включать декораторы **MenuItem** для всех уровней вложенности, например, если нужно вывести только третий уровень иерархии, достаточно лишь использовать `<decorator:MenuItem level="3" />`.








## 4.6. Декоратор MenuItem

Декоратор **MenuItem** является декоратором, используемым **MenuBuilder** для отображения элементов дерева ресурсов.

### Атрибуты

Название	Описание
 <b>Runat</b>	Устанавливает признак того, что декоратор должен обрабатываться на сервере. Обязательный атрибут, значение которого должно быть установлено "server".
 <b>Level</b>	Устанавливает уровень вложенности в структуре дерева ресурсов из которого будет произведена выборка ресурсов.

## Шаблоны

Название	Описание
 <b>ErrorTemplate</b>	Шаблон для отображения ошибки. В теле шаблона можно вставлять произвольный текст и декораторы. Описание возникшей ошибки можно получить декоратором <b>ErrorLabel</b> . Если шаблон не указан — выводится стандартная системная ошибка, при условии, что в свойствах сайта установлен параметр, позволяющий отображать ошибки.
 <b>Active</b>	Шаблон для отображения активного элемента в определенном уровне вложенности, заданным декоратором.
 <b>Inactive</b>	Шаблон для отображения неактивных элементов в определенном уровне вложенности, заданным декоратором.
 <b>LastActive</b>	Шаблон для отображения последнего активного элемента. Если шаблон указан, то действие шаблона <b>Active</b> игнорируется.
 <b>Separator</b>	Разделитель, вставляемый после между элементами. После последнего элемента шаблон не применяется.
 <b>Header</b>	Шаблон, содержимое которого вставляется перед выводом элементов.
 <b>Footer</b>	Шаблон, содержимое которого вставляется после вывода всех элементов.









## Примеры использования

### Пример 1: Использование декоратора для построения меню








```
<decorator:MenuBuilder libraryID="e33d9061-da97-4d58-b003-ef8136ae183a"
queryId="d08171f3-2d50-47b9-beeb-37efd4fa6ba9" runat="server" depth="2"
expand="all">
<ErrorTemplate>Произошла ошибка!</ErrorTemplate>
<decorator:MenuItem level="1">
<active>
<p><a class="active" href='<decorator:ItemProperty runat="server"
source="resource" fieldName="uri" />'><decorator:ItemProperty runat="server"
source="properties" fieldName="title" /></a></p>
</active>
<inactive>
<p><a href='<decorator:ItemProperty runat="server" source="resource"
fieldName="uri" />'><decorator:ItemProperty runat="server" source="properties"
fieldName="title" /></a></p>
</inactive>
</decorator:MenuItem>
<decorator:MenuItem level="2">
<active>
<p>&#151; <a class="active" href='<decorator:ItemProperty runat="server"
source="resource" fieldName="uri" />'><decorator:ItemProperty runat="server"
source="properties" fieldName="title" /></a></p>
</active>
<inactive>
<p>&#151; <a href='<decorator:ItemProperty runat="server" source="resource"
fieldName="uri" />'><decorator:ItemProperty runat="server" source="properties"
fieldName="title" /></a></p>
</inactive>
</decorator:MenuItem>
</decorator:MenuBuilder>
```



В этом примере будет отображено дерево ресурсов по условия, указанным в запросе начиная с узла "/". При этом развернуты будут все узлы дерева, поскольку атрибут *expand="all"*. Глубина отображения дерева ресурсов — 2.



	которого должно быть установлено "server".
 URI	Устанавливает или возвращает Uri отображаемого ресурса приложения C-Gator при отображении множественных свойств ресурса и Uri родительского ресурса, если отображается список ресурсов. Если параметр не указан, то по умолчанию используется Uri из контекста текущей страницы (например, URL запрашиваемой страницы в строке браузера). В шаблоне для получения свойств текущей страницы необходимо размещать декоратор без указания атрибута <b>Uri</b> .
 Columns	Количество столбцов при выводе результатов.
 Rows	Количество строк при выводе результатов.
 XML	Устанавливает признак того надо ли выводить результаты в виде XML или использовать для построения вывода шаблоны декоратора. Если значение установлено <i>true</i> , то все шаблоны игнорируются. Поле может принимать значение <i>true</i> или <i>false</i> .
 XSLT	Устанавливает или возвращает строку со ссылкой на XSLT ресурс. Используется, если значение поля отображается в виде XML в записи множественных свойств или в самом отображаемом ресурсе приложения C-Gator. XSLT преобразование может выполняться как при указании атрибута <b>XML</b> , так и без него. В этом случае XSLT трансформация будет применена к содержимому поля <b>FieldName</b> , данные которого соответствуют XML формату.
 Source	Устанавливает или возвращает строку с названием объекта источника данных для отображения. Существуют следующие источники: " <b>resource</b> ", " <b>properties</b> ", " <b>multiproperties</b> ". По умолчанию — " <b>properties</b> ".
 PageInfo	Устанавливает параметры для ограничения количества выводимых результатов.
 PagerControlName	Устанавливает идентификатор элемента, осуществляющего постраничный вывод результатов. Элемент должен быть декоратором <b>ItemListPager</b> .

## Шаблоны

Название	Описание
 ErrorTemplate	Шаблон для отображения ошибки. В теле шаблона можно вставлять произвольный текст и декораторы. Описание возникшей ошибки можно получить декоратором <b>ErrorLabel</b> . Если шаблон не указан — выводится стандартная системная ошибка, при условии, что в свойствах сайта установлен параметр, позволяющий отображать ошибки.
 ItemTemplate	Шаблон для отображения элемента.
 ItemSeparator	Разделитель, вставляемый после шаблона <b>ItemTemplate</b> и перед отображением следующего элемента. После последнего элемента шаблон не применяется.
 Header	Шаблон, содержимое которого вставляется перед выводом элементов.
 Footer	Шаблон, содержимое которого вставляется после вывода всех элементов.
 RowHeader	Шаблон, вставляемый перед выводом новой строки.
 RowFooter	Шаблон, вставляемый по окончании вывода строки, длина которой ограничена атрибутом <b>Columns</b> .

 ColumnHeader	Шаблон, вставляемый перед началом вывода колонки.
 ColumnFooter	Шаблон, вставляемый по окончании вывода колонки.

Структура вывода элементов в режиме **XML="false"**:

```
<Header />
  <RowHeader />
    <ColumnHeader />
      <ItemTemplate />
    <ColumnFooter />
      <ItemSeparator />
    <ColumnHeader />
      <ItemTemplate />
    <ColumnFooter />
  <RowFooter />
<RowHeader />
  <ColumnHeader />
    <ItemTemplate />
  <ColumnFooter />
    <ItemSeparator />
  <ColumnHeader />
    <ItemTemplate />
  <ColumnFooter />
<RowFooter />
<Footer />
```

## Примеры использования

### Пример 1: Вывод новостей

```
<decorator:ItemList LibraryId="af05f0f8-e6dc-46d6-bf7f-f5b0e1461430"
runat="server" uri="/news/news" Source="multiproperties" >
<Header>
<b>Заголовок</b>
</Header>
<ItemTemplate>
<p style="margin: 5px 0px 5px 0px;">
<decorator:ItemProperty runat="server" Source="multiproperties"
FieldName="date">
<ErrorTemplate>Ошибка: <decorator:ErrorLabel runat="server" /></ErrorTemplate>
</decorator:ItemProperty>
<a href='/news.html/view.html?id=<decorator:ItemProperty runat="server"
Source="multiproperties" FieldName="multi_id" />'><decorator:ItemProperty
runat="server" Source="multiproperties" FieldName="title" />
</a></p>
</ItemTemplate>
<Footer>
<a href="/news.html">Все новости</a>
</Footer>
</decorator:ItemList>
```

Отобразится список новостей в оформлении, заданном в шаблоне **ItemTemplate**. Вверху и внизу новостей будет отображаться строка постраничной навигации по списку новостей.

### Пример 2: Вывод данных в табличной форме

```
<decorator:ItemList LibraryId="af05f0f8-e6dc-46d6-bf7f-f5b0e1461430"
runat="server" uri="/catalog/computer" Source="multiproperties" Columns="3"
Rows="2">
<Header><table border="1" cellpadding="5" cellspacing="0"></Header>
<RowHeader><tr></RowHeader>
<ColumnHeader><td></ColumnHeader>
<ItemTemplate>
<decorator:ItemProperty runat="server" Source="multiproperties"
FieldName="title" />
</ItemTemplate>
```

```
<ColumnFooter></td></ColumnFooter>
<RowFooter></tr></RowFooter>
<Footer></table></Footer>
</decorator:ItemList>
```

Декоратор выдаст таблицу размером 3 столбца на 2 ячейки, в каждой ячейке будет отображаться содержимое поля *title*.





#### Пример 3: Получение результатов в виде XML

```
<decorator:ItemList LibraryId="af05f0f8-e6dc-46d6-bf7f-f5b0e1461430"
runat="server" uri="/catalog/computer" Source="resource" XML="true" />
```


## 4.8. Декоратор ItemListPager

Декоратор **ItemListPager** создает постраничную навигацию по результатам работы декоратора ItemList.

### Атрибуты

Название	Описание
 Runat	Устанавливает признак того, что декоратор должен обрабатываться на сервере. Обязательный атрибут, значение которого должно быть установлено "server".
 DataControlName	Устанавливает идентификатор элемента к которому будет применяться пейджинг.
 Method	Устанавливает тип формирования ссылок. <b>get</b> — через источник <code>get (?pagenumber=1)</code> <b>tail</b> — через источник <code>tail (/_t_/pagenumber=1)</code>
 PageSize	Устанавливает список полей разделенных запятой для проверки по регулярным выражениям.

### Шаблоны

Название	Описание
 ErrorTemplate	Шаблон для отображения ошибки. В теле шаблона можно вставлять произвольный текст и декораторы. Описание возникшей ошибки можно получить декоратором <b>ErrorLabel</b> . Если шаблон не указан — выводится стандартная системная ошибка, при условии, что в свойствах сайта установлен параметр, позволяющий отображать ошибки.

### Примеры использования


#### Пример 1: Создание пейджинга




```
<decorator:ItemListPager Runat="server" ID="ctlTest" DataControlName="ilList1"
Method="tail" PageSize="1"/>
```

## 4.9. Декоратор FormPostBack




Декоратор **FormPostBack** позволяет отправлять данные с клиента на сервер для совершения серверных операций. Используется для изменения свойств ресурса или множественных свойств

### Атрибуты

Название	Описание
 Runat	Устанавливает признак того, что декоратор должен обрабатываться на сервере. Обязательный атрибут, значение которого должно быть установлено "server".

 Uri	Устанавливает Uri ресурса.
 LibraryId	Устанавливает идентификатор библиотеки.
 Source	Устанавливает или возвращает строку с названием объекта источника данных для отображения. Существуют следующие источники: " <b>resource</b> ", " <b>properties</b> ", " <b>multiproperties</b> ". По умолчанию — " <b>properties</b> ".

## Шаблоны

Название	Описание
 FormTemplate	Шаблон для размещения формы.
 SuccessTemplate	Шаблон для отображения сообщения в случае успешного проведения операции на сервере.
 ErrorTemplate	Шаблон для отображения ошибки. В теле шаблона можно вставлять произвольный текст и декораторы. Описание возникшей ошибки можно получить декоратором <b>ErrorLabel</b> . Если шаблон не указан — выводится стандартная системная ошибка, при условии, что в свойствах сайта установлен параметр, позволяющий отображать ошибки.

## Примеры использования





### Пример 1: Добавление множественного свойства





```
<decorator:FormPostBack Uri="/guestbook/guestbook/" LibraryId="abcd9517-78c4-4782-8dbb-a3d81ebf1fd1" Runat="server" Source="multiproperties">
<FormTemplate>
<decorator:ItemProperty FieldName="postdate" FieldRenderName="FieldVarRender"
Runat="server" Source="multiproperties" Mode="Edit">
<ValueTemplate><decorator:GetVar Name="common[datetime]"
Runat="server"/></ValueTemplate>
</decorator:ItemProperty>
<br/>
<decorator:ItemProperty FieldName="name" FieldRenderName="FieldVarRender"
Runat="server" Source="multiproperties" Mode="Edit">
<ValueTemplate><decorator:GetVar Name="get[name]"
Runat="server"/></ValueTemplate>
</decorator:ItemProperty>
</FormTemplate>
</decorator:FormPostBack>
```

## 4.10. Декоратор ActionButton


Декоратор **ActionButton** создает кнопку для совершения указанного действия в декораторе FormPostBack.

### Атрибуты

Название	Описание
 Runat	Устанавливает признак того, что декоратор должен обрабатываться на сервере. Обязательный атрибут, значение которого должно быть установлено "server".
 SensorFields	Устанавливает список полей разделенных запятой для проверки по регулярным выражениям.
 Source	Устанавливает или возвращает строку с названием объекта источника данных для отображения. Существуют следующие источники: " <b>resource</b> ", " <b>properties</b> ", " <b>multiproperties</b> ". По умолчанию — " <b>properties</b> ".
 Action	Устанавливает тип действия.

 <b>Text</b>	Устанавливает отображаемый текст на кнопке.
 <b>DataBindControl</b>	Устанавливает признак производить или нет DataBind при отправке формы.
 <b>ControlName</b>	Устанавливает имя связанного элемента.
 <b>CheckIP</b>	Устанавливает признак проверки IP адреса.

## Шаблоны

Название	Описание
 <b>ErrorTemplate</b>	Шаблон для отображения ошибки. В теле шаблона можно вставлять произвольный текст и декораторы. Описание возникшей ошибки можно получить декоратором <b>ErrorLabel</b> . Если шаблон не указан — выводится стандартная системная ошибка, при условии, что в свойствах сайта установлен параметр, позволяющий отображать ошибки.

## Примеры использования


### Пример 1: Создание кнопки

```
<decorator:ActionButton CensorFields="theme,message" Runat="server"
Source="multiproperties" Action="CreateMultiProperty" Text="Add"
DataBindControl="true" ControlName="ilList1" CheckIP="true" />
```



## 4.11. Декоратор SendEmail



Декоратор **SendEmail** осуществляет отправку данных на указанный e-mail.

## Атрибуты

Название	Описание
 <b>Runat</b>	Устанавливает признак того, что декоратор должен обрабатываться на сервере. Обязательный атрибут, значение которого должно быть установлено "server".
 <b>SendTo</b>	Устанавливает значение e-mail куда будет отправлено письмо.
 <b>ReturnAddress</b>	Устанавливает значение e-mail адреса для ответа на письмо.
 <b>SmtpServer</b>	Устанавливает значение SMTP сервера для отправки письма.
 <b>Name</b>	Устанавливает имя отправителя.
 <b>Subject</b>	Устанавливает имя переменной, значение которой будет возвращено в результате работы декоратора.
 <b>ShowFormOnPostBack</b>	Устанавливает показывать или нет форму сообщения после отправки ее на сервер.
 <b>ContentType</b>	Устанавливает значение Content-Type письма. Может принимать значения <b>text</b> (Content-Type: text/plain) или <b>html</b> (Content-Type: text/html).
 <b>Message</b>	Устанавливает значение текста сообщения.
 <b>ButtonName</b>	Устанавливает имя кнопки при нажатии на которую происходит отправка письма.

## Шаблоны

Название	Описание
 <b>EmailTemplate</b>	Шаблон для создания текста письма.
 <b>ActionForm</b>	Шаблон для отображения формы ввода данных.

 <b>ErrorTemplate</b>	Шаблон для отображения ошибки. В теле шаблона можно вставлять произвольный текст и декораторы. Описание возникшей ошибки можно получить декоратором <b>ErrorLabel</b> . Если шаблон не указан — выводится стандартная системная ошибка, при условии, что в свойствах сайта установлен параметр, позволяющий отображать ошибки.
 <b>SuccessTemplate</b>	Шаблон для отображения текста в случае успешной отправки письма.

## Примеры использования








### Пример 1: Отправка сообщения

```
<decorator:SendEmail SendTo="name@mailserver.net "
ReturnAddress="name@mailserver.net " SmtпServer="mailserver.net "
Name="@post[name]" ShowFormOnPostBack="false" Subject="Feedback form"
ContentType="html" Message="@post[message]" ButtonName="btnSend"
Runat="server">
<EmailTemplate>
from #name# <#sendto#>:
#message#
mailto: #returnaddress#
</EmailTemplate>
<ActionForm>
    <textarea id="message" name="message" rows="10" style="width: 100%"
wrap="virtual"></textarea><br>
    <asp:button id="btnSend" Text="Отправить"
runat="server"></asp:button><br>
</ActionForm>
</decorator:SendEmail>
```




## 4.12. Декоратор GoogleSearch


Декоратор GoogleSearch осуществляет поиск по сайту используя поисковый сервер Google. Для поиска по сайту необходимо чтобы сайт был проиндексирован системой.

### Атрибуты

Название	Описание
 <b>Runat</b>	Устанавливает признак того, что декоратор должен обрабатываться на сервере. Обязательный атрибут, значение которого должно быть установлено "server".
 <b>SiteUri</b>	URI сайта по которому осуществляется поиск.
 <b>NextLикContent</b>	Текстовое содержание для ссылки на следующую страницу результатов поиска.
 <b>PrevLinkContent</b>	Текстовое содержание для ссылки на предыдущую страницу результатов поиска.
 <b>DocPerPage</b>	Количество результатов на страницу.
 <b>PageRank</b>	RANK страницы.
 <b>ShowPage</b>	Номер страницы для отображения.

### Шаблоны

Название	Описание
 <b>HeaderTemplate</b>	Шаблон для заголовка вывода результатов.
 <b>FooterTemplate</b>	Шаблон для подвала вывода результатов.
 <b>ItemTemplate</b>	Шаблон для вывода результатов поиска.

 <b>ErrorTemplate</b>	Шаблон для отображения ошибки. В теле шаблона можно вставлять произвольный текст и декораторы. Описание возникшей ошибки можно получить декоратором <b>ErrorLabel</b> . Если шаблон не указан — выводится стандартная системная ошибка, при условии, что в свойствах сайта установлен параметр, позволяющий отображать ошибки.
--	--

## Метапеременные для HeaderTemplate и FooterTemplate

#text# — строка поиска  
#pageCount# — количество страниц  
#pager# — пейджер  
#startIndex# — начальный индекс  
#endIndex# — конечный индекс  
#searchComments# — комментарий поиска  
#searchQuery# — строка запроса  
#googleSearchQuery# — строка запроса  
#searchTime# — время затраченное на поиск  
#searchTips# — поисковая подсказка

## Метапеременные для ItemTemplate

#number# — порядковый номер результата поиска  
#cachedSize# — размер страницы  
#directoryCategory# — категория директории  
#directoryTitle# — заголовок директории  
#hostName# — имя хоста  
#relatedInformationPresent#  
#snippet#  
#summary#  
#title# — заголовок  
#URL# — ссылка на страницу

## Пример использования

```




<decorator:GoogleSearch
  SiteURI="udmgossovet.ru"
  Text="@text"
  NextLinkContent="След. &#8250;&#8250;"
  PrevLinkContent="&#8249;&#8249;&#8249; пред. "
  DocPerPage="15"
  PageRank="10"
  ShowPage="@showpage"
  Runat="server"
>
<HeaderTemplate>
  <b style="font-size: 120%; ">Результаты поиска:
  &#0171;#searchQuery#&#0187;</b>
  <p>Документы <b>#startIndex#</b>&#0151;<b>#endIndex#</b>&#8201;из
  <b>#pageCount#</b>&#8201;найденных документов.
  <ol start="&#8201;#startIndex#">
</HeaderTemplate>
<ItemTemplate>
  <p><li><a href="&#8201;#URL#"><b>#title#</b></a>
  <span class=fontDarkGray> (&#8201;#cachedSize#)</span><br/>
  &#8201;#snippet#
</ItemTemplate>
<FooterTemplate>
  </ol>
  Другие страницы найденных документов: &#8201;#pager#
</FooterTemplate>
</decorator:GoogleSearch>

```





## 4.13. Декоратор If

Декоратор If осуществляет проверку по заданному условию.

### Атрибуты

Название	Описание
 Runat	Устанавливает признак того, что декоратор должен обрабатываться на сервере. Обязательный атрибут, значение которого должно быть установлено "server".
 Cond	Устанавливает условие проверки. В условии используются параметры с именами @param1...@param5, значения которых задаются через атрибуты декоратора.
 Param1...Param5	Параметры декоратора.

### Шаблоны

Название	Описание
 ErrorTemplate	Шаблон для отображения ошибки. В теле шаблона можно вставлять произвольный текст и декораторы. Описание возникшей ошибки можно получить декоратором <b>ErrorLabel</b> . Если шаблон не указан — выводится стандартная системная ошибка, при условии, что в свойствах сайта установлен параметр, позволяющий отображать ошибки.
 Then	Шаблон для вывода результата в случае успешной проверки.
 Else	Шаблон для вывода результата в случае несовпадения условия проверки.
 ElseIf	Шаблон для вывода результата в случае несовпадения условия проверки и задания нового условия проверки. В шаблоне используются те же атрибуты, что и в декораторе If.

### Примеры использования



Пример 1: Проверка даты. День и месяц должны равняться 1.

```
<decorator:If cond="@param1=1 and @param2=1 " runat="server" param1="@get [day]"
param2="@get [month]">
  <then>>true</then>
  <else>>false</else>
</decorator:If>
```


## 4.14. Декоратор IfHasRole



Декоратор IfHasRole проверяет у авторизованного пользователя наличие роли в одной из входящих его групп.

### Атрибуты

Название	Описание
 Runat	Устанавливает признак того, что декоратор должен обрабатываться на сервере. Обязательный атрибут, значение которого должно быть установлено "server".
 Roles	Устанавливает имя роли.

### Шаблоны


Название	Описание
 Allow	Шаблон для вывода результата в случае успешной проверки.

 Deny	Шаблон для вывода результата в случае отсутствия у пользователя указанной роли.
 ErrorTemplate	Шаблон для отображения ошибки. В теле шаблона можно вставлять произвольный текст и декораторы. Описание возникшей ошибки можно получить декоратором <b>ErrorLabel</b> . Если шаблон не указан — выводится стандартная системная ошибка, при условии, что в свойствах сайта установлен параметр, позволяющий отображать ошибки.

## Примеры использования

### Пример 1: Установка значения переменной


```
<decorator:IfHasRole runat="server" Roles="RoleName">
<allow>Просмотр страницы разрешен!</allow>
<deny>Просмотр страницы не разрешен!</deny>
</decorator:IfHasRole>
```

-  Устанавливая разным группам свой набор ролей можно создавать персональные закрытые разделы.

## 4.15. Декоратор ErrorLabel

Декоратор **ErrorLabel** позволяет получить содержание текста ошибки, возникшей при работе декоратора. Наиболее распространенное применение декоратора в шаблоне **ErrorTemplate**, включенного во все декораторы.

### Атрибуты

Название	Описание
 Runat	Устанавливает признак того, что декоратор должен обрабатываться на сервере. Обязательный атрибут, значение которого должно быть установлено "server".

## Примеры использования

### Пример 1: Использование декоратора для построения меню

```
<decorator:ItemProperty runat="server" source="properties"
FieldName="notexistingfield">
<ErrorTemplate>
    Произошла ошибка: <decorator>ErrorLabel runat="server" />
</ErrorTemplate>
</decorator:ItemProperty>
```

При запросе несуществующего поля *notexistingfield* будет отображена ошибка примерного содержания:

**Произошла ошибка: Поле notexistingfield не существует.**

## 5. Приложения

---

### 5.1. Требования к системе

---

#### **Сервер**

Компьютер	PC
Операционная система	Windows NT Server
База данных	MS SQL Server
Компоненты и приложения	MS Internet Information Server 5.0
Серверный SSL 3.0 сертификат для обеспечения безопасности передачи данных	

---

#### **Клиент, работающий с административной частью системы**

Компьютер	PC
Операционная система	Windows 98/Me/2000 или Windows XP
Браузер	Internet Explorer версии 5.5 или выше

---

#### **Клиент**

Требования зависят только от содержимого страниц, помещенных в систему

---

## 5.2. Словарь терминов

### **HTML (HyperText Markup Language)**

На этом языке браузеру сообщается, какой именно текст и другие элементы (картинки, таблицы, формы) и каким образом нужно отображать на странице. На языке HTML не программируют, а верстают - особым образом размечают текст для публикации в интернете.

Язык HTML позволяет связывать страницы между собой с помощью ссылок (линков). Наличие ссылок является фундаментальным свойством веб-страниц. Ссылкой может являться не только некоторая часть текста, но и картинка или ее часть.

### **URL (Universal Resource Locator)**

Адрес страницы в интернете. URL состоит из доменного имени, пути к странице на сайте и имени файла страницы. Например:  
[www.mysom.ru/pub/stories/13.htm](http://www.mysom.ru/pub/stories/13.htm).  
Здесь [www.mysom.ru](http://www.mysom.ru) - доменное имя сайта, [/pub/stories/](http://www.mysom.ru/pub/stories/) - путь и [13.htm](http://www.mysom.ru/pub/stories/13.htm) - имя файла.

### **WYSIWYG (What You See Is What You Get)**

Стиль редактирования материала (в частности - текста или графики), при котором то, что вы видите на экране при редактировании, в точности соответствует тому что вы получите в результате (при просмотре, на печатной копии и т.п.).

### **Администратор системы**

Пользователь домена, занимающийся настройкой системы, определением прав и ролей порользователей.

### **Декоратор**

Специальный тэг, позволяющий формировать облик страницы динамически. Применяется в шаблонах.

### **Домен**

Организационная единица C-Gator. Каждый домен имеет свой набор пользователей, групп и сайтов. Каждый домен также имеет свое хранилище ресурсов. Изначально в системе имеется только административный домен.

### **Группа**

Набор пользователей, объединенных под одним именем. Каждой группе назначен набор ролей, который задает права пользователей группы. Группа принадлежит определенному домену и может включать пользователей только этого домена.

### **Картинка**

Один из типов ресурсов. Ресурс такого типа сохраняет картинку для вставки на страницы сайта.

### **Каталог товаров**

Один из типов ресурсов. Позволяет формировать на сайте список товаров с заданными параметрами

### **Кэш**

Подготовленные копии уже просмотренных ресурсов (например, страниц или картинок). При повторном запросе к кэшированным ресурсам, они не подготавливаются заново, а берутся из кеша. Применение кеша снижает нагрузку на интернет и повышает скорость загрузки ресурсов.

### **Лента новостей**

Один из типов ресурсов, предназначенный для подготовки новостей к публикации.

Интернет в высшей степени удобен для публикации новостей, поэтому новости обычно присутствуют не только в интернет-СМИ, но и на корпоративных сайтах. Чаще всего новости выглядят как список заголовков, расположенных друг под другом, и похожи на бесконечную ленту - свежие события появляются сверху, а старые уходят в архив.

### **Логин**

Имя пользователя, под которым пользователь входит в систему. При входе пользователь вводит имя (логин) и пароль, что позволяет идентифицировать его, определить к какому домену принадлежит пользователь и определить набор его прав в системе. Имя пользователя (логин) уникально в рамках системы.

**Папка**

Один из типов ресурсов. Предназначен для организации ресурсов в древовидную структуру - каталог может объединять в себе ресурсы любых типов.

**Пользователь**

Учетная запись, содержащая информацию, необходимую для авторизации реального пользователя при входе в систему и определения его прав при работе в системе. Содержит логин и пароль, задает принадлежность пользователя к группам.

**Приоритет**

Число, характеризующее для пользователя важность (срочность) работы над данным ресурсом.

**Ресурс**

Элемент дерева ресурсов. Набор операций над ресурсом определяется типом ресурса. Информация о ресурсе содержится в хранилище ресурсов домена.

**Роль**

Административная единица системы, определяющая набор разрешенных действий. Каждой группе пользователей назначается набор ролей, в результате чего пользователи, входящие в данную группу, получают права, определяемые ролями группы. Как правило, набор ролей и разрешенных для них действий задается администратором однократно при установке системы.

**Сайт**

Запись, связывающая имя Internet-домена с главной страницей, соответствующей этому Internet-домену. Имя сайта является именем домена Internet (например, www.mycompany.com). Каждый домен системы может включать в себя несколько сайтов.

**Список рассылки**

Один из типов ресурсов. С помощью списка рассылки организуется рассылка почтовой сообщений некоторой группе пользователей.

**Статистика сайта**

Владелец веб-сайта обычно хочет знать, сколько на сайте посетителей, какие разделы пользуются наибольшей популярностью и т.д. Такая информация собирается при анализе протокола посещений (т.н. лог).

**Страница**

Один из типов ресурсов. Содержит текстовую информацию в формате HTML.

**Супервизор**

Пользователь, входящий в административный домен, администратор системы. Учетные записи супервизоров используются только для настройки системы, рядовую работу выполняют пользователи других доменов.

**Тип ресурса**

Системная запись, определяющая один из видов ресурсов, доступных для создания и изменения. Тип ресурса связывает между собой класс, обеспечивающий хранение ресурса, и элемент управления, т.е. способ представления ресурса для пользователя. Тип ресурса задает также набор доменов, в которых можно создавать, редактировать и использовать ресурсы данного типа.

Типы ресурсов задаются супервизором при установке/настройке системы.

**Шаблон**

Страница используемая в качестве образца для создания других страниц. В тексте шаблона могут использоваться декораторы, с помощью которых можно гибко управлять содержимым страницы.

### 5.3. Предметный указатель

- CMS-система, 3
- SSL, 29
- Администратор, 6, 7, 14
  - регистрация, 14
  - удаление, 14
- Библиотека, 15
- Браузер, 29
- Группа, 8, 9
  - набор пользователей, 10
  - назначение прав, 9
  - создание, 9
  - удаление, 11
- Домен, 14
  - администрирование, 6
  - создание, 15
  - удаление, 15
- Запрос, 23
  - описание, 24
  - примеры, 28
  - создание, 24
- Иконки, 12
  - добавление, 12
  - удаление, 13
- Кэш, 17
  - очистка, 18
  - размер, 18
  - список объектов, 18
- Ошибки, 16
  - детальная информация, 17
  - очистка протокола, 17
  - просмотр ошибки, 17
  - просмотр протокола, 16
  - удаление, 17
- Пользователь, 6
  - блокирование, 8
  - назначение прав, 8
  - регистрация, 7
  - удаление, 8
- Поля, 20
  - добавление, 20
  - дополнительные параметры, 21
  - типы полей, 20
  - удаление, 22
- Роли
  - типовой набор, 10
- Сайт, 11
  - имена сайтов, 11, 16
  - создание, 11
  - удаление, 12
- Система безопасности, 9
- Тип ресурса, 18
  - запросы, 23
  - множественные поля, 23
  - обработчик типа, 19
  - поля, 20
  - список закладок, 23
- Учетная запись, 6, 8