Руководство

по использованию модуля

«Антирутин»

Версия модуля 20.0.0

18 апреля 2020г.

Содержание

Введение 3

Технические требования 4

Загрузка модуля на сайт 4

Использование модуля 7

- Начало работы 7
- Фильтр 8
- Действия 9
- Ручной запуск 9
- Профили 10

Для разработчиков 11

- Файл class.php 11
- Файл .form.php 12
- Файл script.js 13
- Файл style.css 13
- Файл .description.php 14
- Наследование плагинов 14

Логи модуля 15

Техподдержка 15

Введение

Модуль «Антирутин» создан для массовой работы с каталогом товаром, для решения таких задач, которые обычно занимают долгие часы, дни и даже недели работы:

- заполнение полей и свойств (а также, цен, остатков, SEO-данных и др) в товарах и разделах,
- копирование значений между полями и свойствами,
- заполнение цен, наценки, скидки, конвертация валюты,
- выполнение замен в тексте (простые замены, замены с регулярными выражениями, дописывание в конец и в начало),
- работа с привязками к разделам (перенос в раздел, добавление и удаление привязок, смена основного раздела),
- работа с изображениями (уменьшение изображений, обработка модулем «Обработчик изображений», и др),
- и многое-многое другое.

При этом модуль справляется со своими задача максимально быстро, обработка тысяч товаров может быть выполнена за несколько секунд.

Основные особенности «Антирутина»:

- подходит для любого сайта,
- прост в установке и использовании, обладает интуитивно понятным интерфейсом,
- гибок и универсален в настройке задач,
- адаптирован для эффективной ежедневной работы,
- не требователен к сайту, серверу и производительности.

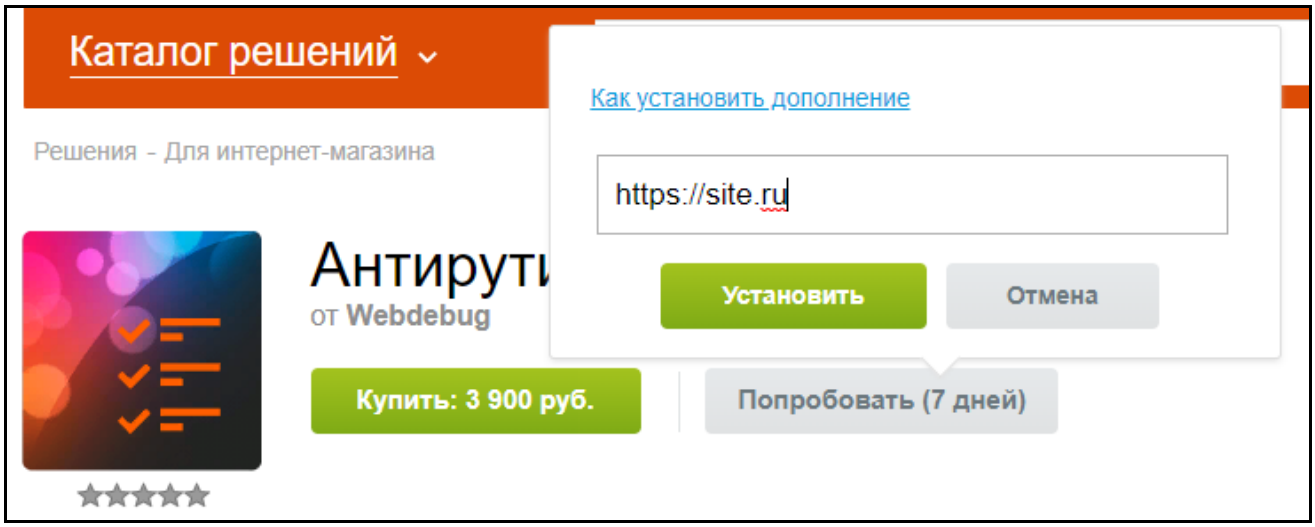
Все это делает модуль максимально полезным и эффективным инструментом, который, несомненно, будет важным приобретением для любого сайта.

Технические требования

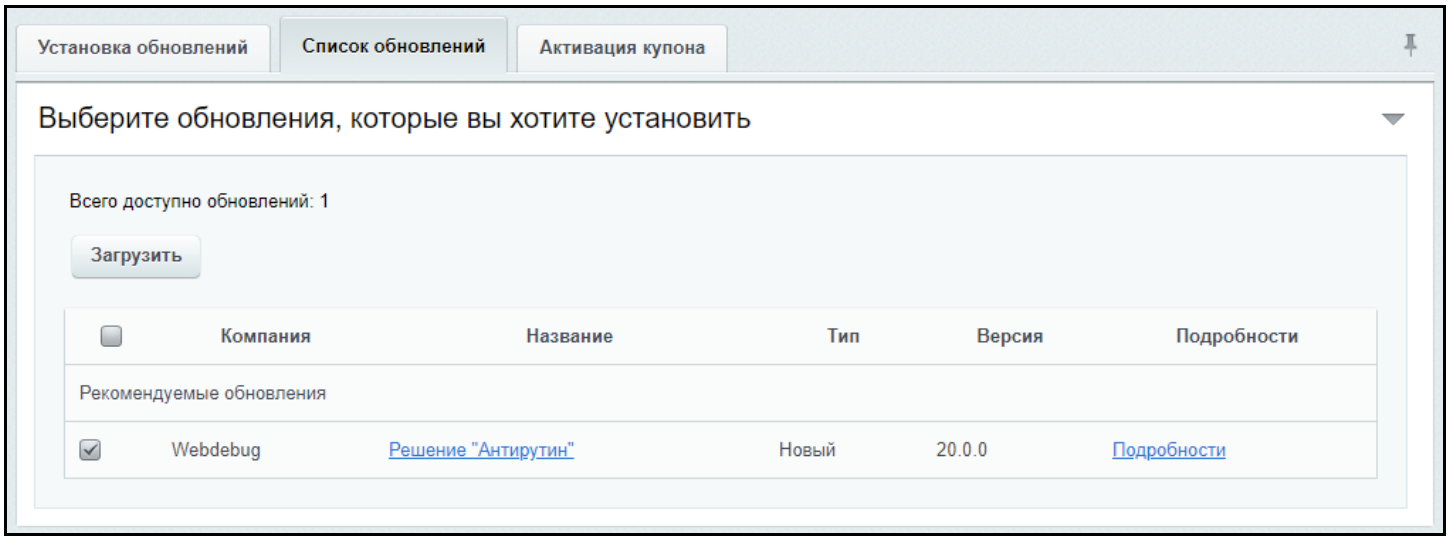
1. Система управления 1С-Битрикс любой редакции, в т.ч. «Первый сайт»,
2. Отсутствие ошибок при стандартной проверке сайта,
3. Дополнительные требования к хостингу не предъявляются, сайт одинаково хорошо работает как на обычно виртуальном хостинге, так и на VPS и выделенных серверах.

Загрузка модуля на сайт

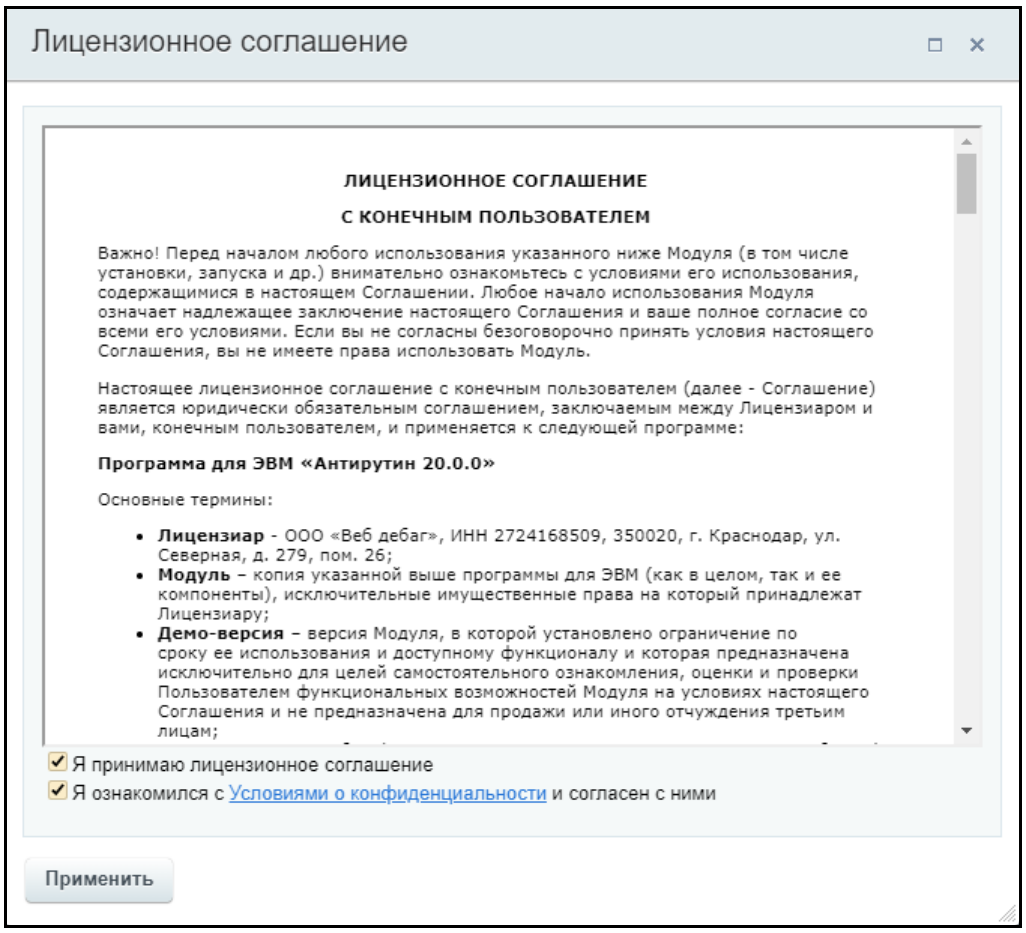
Установка модуля начинается с загрузки его на сайт. Загрузить модуль можно на его странице в сервисе 1С-Битрикс: Маркетплейс. Для этого нажмите кнопку «**Попробовать**», в открывшемся окне укажите адрес сайта, например, «https://www.site.ru» и нажмите «**Установить**»:



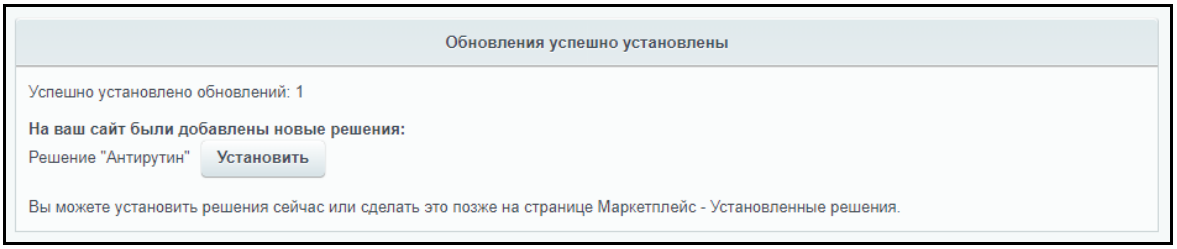
После этого откроется страница административного раздела Вашего сайта, где Вам предлагается загрузка указанного модуля. Нажмите кнопку «Загрузить».



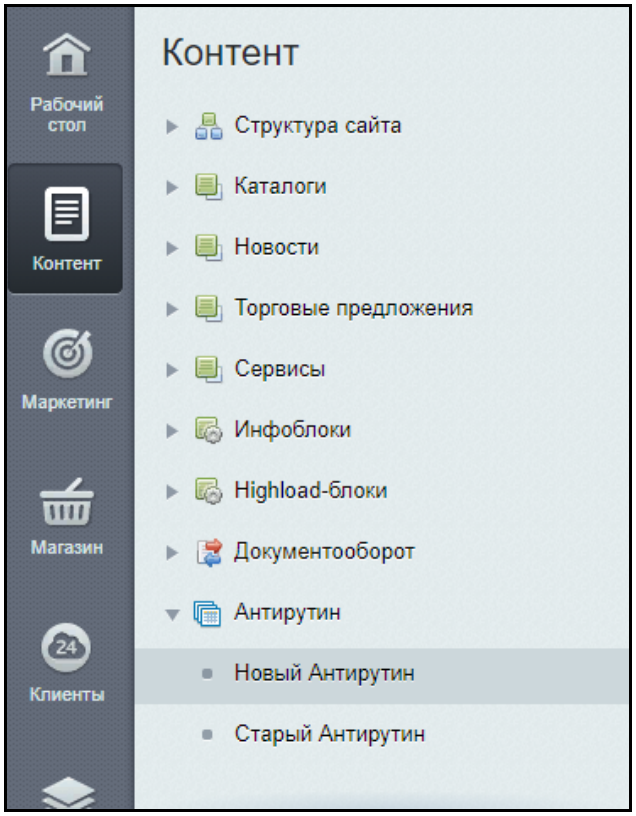
В появившемся окне с лицензионным соглашением отметьте двумя галочками свое согласие и нажмите «Применить»:



После загрузки модуля будет предложено его установить. Нажмите кнопку «Установить»:



Все! Теперь модуль установлен. Перейти на административные страницы модулей (как старого, так и нового), можно на странице в административном разделе: «Контент» – «Антирутин».



Использование модуля

Вся работа с модулем выполняется на одной странице (см. рисунок) - «Новый Антирутин». Ниже опишем устройство административной страницы и порядок работы с ней.

Начало работы

Прежде всего, необходимо выбрать с чем мы будем работать – **элементы** или **разделы** (блок «Что обрабатывать»). Это две абсолютно разные (и в то же время связанные между собой) сущности Битрикса, поэтому и в нашем модуле это разные сущности. Принципы работы для обоих вариантов практически одинаковые, но действия при этом разные (хотя, многие из них идентичные для обоих сущностей). В дальнейшем будет называть эти два варианта работы модуля как «**Режим обработки элементов**» и «**Режим обработки разделов**».

Далее, выбираем инфоблок с которым будем работать. Здесь также предлагается опция «Выбрать разделы» для уточнения разделов, с которыми будем работать. Здесь работа в обоих режимах аналогична: эти разделы служат для уточнения выборки. При режиме обработки разделов Вам может

показаться, что этот список – это именно то, какие разделы модуль будет обрабатывать, но это не так! В списке мы лишь выбираем родительские – т.е. модуль найдет все вложенные разделы, которые есть в выбранных в списке разделах. Также, важно, что в режиме обработки элементов имеется возможность выбора вложенных любого уровня, а в режиме обработки разделов модуль может выбрать только дочерние для непосредственно выбранных.

Выпадающий список «**Вложенность**» не влияет на обработку, он только помогает удобно отобразить дерево разделов – т.к. иногда каталог имеет высокий уровень вложенности, показывать который полностью не имеет смысла. В таком случае, если позволяет задача, удобно вывести для отображения только первые 1, 2 или 3 уровня.

Фильтр

Следующий этап настройки задачи – указание фильтра. Он необходим для случаев, когда выбор разделов не ограничивает условия поиска: например, нужно указать модулю чтобы он выбрал только активные товары, или только те товары, у которых в свойстве, например, «Распродажа» проставлено «Да», или, например, только те товары, у которых бренд задан как «Одежда+».

Фильтр в модуле собственный, но он интуитивно понятен: добавляем, если необходимо, группу (группы нужны для группировки, для построения сложных условий с логикой И / ИЛИ), в группе добавляем поле, для каждого поля выбираем (нажимая на ссылку): поле/свойство, логику и значение. Выбор значений в отдельном всплывающем окне позволяет максимально гибко выбирать значения: например, для свойств типа «Строка» по умолчанию подгружается обычное текстовое поле, но для логики «в списке» подгружается поле с возможностью добавления дополнительных значений. Для свойств типа «привязка к элементам», «привязка к разделам», «справочник» и «список» подгружается выпадающий список значений.

При настройке фильтра обращайте внимание на логику – И / ИЛИ. Это разные варианты работы фильтра, приводящее к разным результатам.

При каждом изменении в фильтре модуль автоматически ищет товары и показывает количество найденных (поле «Найдено: 1»), если нажать на эту ссылку – откроется всплывающее окно со списком найденных элементов/разделов.

Фактически фильтр работает стандартным для 1С-Битрикс способом (для элементов это CIBlockSection::getList, для разделов – CIBlockSection::getList), поэтому фильтр в модуле всего лишь генерирует массив нужного формата, который будет использоваться при фильтрации. Просмотреть этот фильтр (это полезно для контроля того, что создает фильтр) можно при нажатии на ссылку «Показать массив».

Действия

Следующий этап настройки задачи – это указание действий, выполняемых в задаче. Действие – это некая операция по обработке элементов/разделов. Модуль поддерживает указание произвольного количества действий в одной задаче.

Физически каждое действие в модуле - это динамически подключаемый функционал, поэтому и здесь, и в других справочных материалах мы также называем действие плагином. Список плагинов в модуле ограничен, но модуль поддерживает подключение и пользовательских плагинов, т.е. каждый клиент модуля может разработать (конечно же, необходим опыт веб-разработки) собственное действие для модуля, которое будет выполнять с элементами и разделами самые разные операции.

При выборе действия модуль подгружает форму настроек. В большинстве действий в форме требуется выбрать некоторое поле/свойство для его обработки. Также в некоторых случаях предлагается выбор конечного поля/свойства – т.е. поля/свойства, куда будет сохранено обработанное значение.

Как уже говорилось выше, каждая задача может содержать любое количество разных действий. После того как вы добавили и настроили форму действия, вы можете его добавить в задачу – для этого нажмите кнопку «Закрепить». После этого можете добавить новое действие.

Каждое действие можно переименовать – чтобы при большом количестве действие на запутаться в них, и быстрее ориентироваться. Также, каждое действие можно свернуть/развернуть, нажимая на заголовок. А при нажатии кнопки «Справка» открывается информационное окно, содержащее краткую информацию по работе плагина.

Ручной запуск

После настройки задачи вы можете запустить ручное выполнение (также доступен автоматический запуск, для этого необходимо сохранить задачу в профиль – об этом в следующем разделе), нажав кнопку «Запустить». Доступна горячая клавиша для запуска – Alt+X.

При ручном запуске обработка выполняется по шагам, время шага ограничивается в настройках модуля (также можно ограничить время прямо в настройках задачи, нажав «Настройки» напротив заголовка «Действия»). На больших каталогах в некоторых случаях (напр., для повышения общей скорости выполнения) имеет смысл увеличить значение шага обработки, но максимальное значение в каждом случае разное – это зависит от таймаутов сервера, превышение этих таймаутов может привести к проблемам с выполнением задачи.

Профили

Профили – это сохраненная совокупность настроек и всех действий, относящихся к одной задаче. Чтобы создать профиль, настройте задачу (укажите инфоблок, разделы, фильтр, и действия с настройками) и нажмите кнопку «Сохранить профиль» (горячая клавиша Alt+S). При сохранении профиля нужно указать имя профиля, желательно также указать описание, чтобы в дальнейшем не запутаться и точно понимать для чего он был создан. Символьный код профиля необходимо указывать, если планируется работа с профилем через API.

Открыть список профилей можно, нажав кнопку «Управление профилями» (в верхней части страницы), либо горячей клавишей Alt+O. Здесь каждая строка таблицы – это отдельный профиль. Иконка в левой части показывает, на обработку чего настроен профиль – элементов или разделов. В правой части каждой строки имеются кнопки для настройки регулярного автозапуска, и для удаления профиля. Нажатие на любую часть строки загружает выбранный профиль (будьте осторожны, если текущие настройки не сохранены – все несохраненные настройки будут сброшены).

Нажатие на кнопку «Настроить автозапуск» открывает всплывающее окно для настройки автоматического запуска профиля. Поддерживается произвольное количество заданий для запуска. Каждое задание настраивается по времени в стиле планировщика Cron. Модуль сохраняет эти задачи в crontab сервера.

Несколько примеров настройки времени выполнения заданий:

- * * * * * - каждую минуту,
- */5 * * * * - каждые 5 минут,
- 0 7 * * * - каждый день в 7:00 утра,
- 0 1 */3 * * - каждые 3 дня в 1:00 ночи,
- 0 9,12,15,18 * * * - каждый день в 9 часов, 12 часов, 15 часов и 18 часов.

Внимание! Возможность настроить задания есть не на каждом сервере, примерно в 1% случаев такой возможности на сервере нет. И если у вас модуль сообщает, что такой возможности нет, то сначала проверьте настройки модуля («**Путь к PHP на сервере**», «Дополнительные конфиги PHP», «Добавлять параметры mbstring»), и если там все заполнено верно – значит, ваш сервер не поддерживает автоматическое добавление заданий в планировщик, и задания запуска необходимо настраивать вручную на сервере.

Для разработчиков

Модуль предоставляет разработчикам возможность создавать собственные плагины для модуля. Каждый плагин представляет собой php-файл class.php, а также некоторые дополнительные файлы.

/my_plugin – папка плагина

- /lang/ru/.form.php – языковые фразы для файла .form.php (в кодировке сайта),
- /lang/ru/class.php – языковые фразы для файла .form.php (в кодировке сайта),
- /.description.php – файл с описанием (всегда в UTF-8)
- /.form.php – файл с формой настроек,
- /class.php – основной класс плагина,
- /icon.png – картинка плагина размером 16x16 пикселей,
- /script.js – скрипты плагина,
- /style.css – стили плагина.

Собственный плагин подключается через обработчик модуля:

```
addEventHandler('webdebug.antirutin', 'OnFindPlugins', 'myOnFindPlugins');
function myOnFindPlugins($strEntityType, $strPluginsDir){
    require_once(__DIR__.'/include/wda_plugins/my_plugin/class.php');
}
```

При этом свой плагин нельзя класть в папку модуля. Можно положить его, например, в файл /bitrix/php_interface/include/antirutin_my_plugin.

Файл class.php

Общий вид плагина для элементов (наследуется от PluginElement):

```
namespace WD\Antirutin\Plugins\Element;

use
      \WD\Antirutin\Helper,
      \WD\Antirutin\IBlock,
      \WD\Antirutin\PluginElement;

class MyPlugin extends PluginElement {

      public function processElement($intElementId){
```

```
		//
	}

}
```

Общий вид плагина для разделов (наследуется от PluginSection):

```
namespace WD\Antirutin\Plugins\Section;

use
	\WD\Antirutin\Helper,
	\WD\Antirutin\IBlock,
	\WD\Antirutin\PluginSection;

class MyPlugin extends PluginSection {

	public function processSection($intSectionId){
		//
	}

}
```

Методы processElement и processSection должны возвращать true в случае успешной обработки, false – в случае ошибки (в случае ошибки необходимо задать текст ошибки: $this->setError('Текст ошибки')). В плагине для элементов метод processSection запрещен (и аналогично для разделов запрещен processElement).

В плагине возможно объявления переменной $arFieldsFilter, для дальнейшего использования ее в методе $this->getFields() – это требуется для построения списков доступных полей.

Файл .form.php

Данный файл содержит верстку и код формы настроек. Поля формы необходимо создавать следующие образом:

```
<div class="plugin-form__field">
	<div class="plugin-form__field-title">
		<?=$this->fieldName('FIELD', true);?>
	</div>
	<div class="plugin-form__field-value">
		<div id="<?=$this->getId('select');?>">
			<?=IBlock::showAvailableFields($this->getFields(), $this->strEntityType,
$this->getInputName('field'),
				$this->get('field'), 'data-role="field"', true);?>
		</div>
	</div>
</div>
```

Как видно из примера, в файле доступна переменная $this (объект класса текущего плагина).

Файл script.js

Скрипты плагинов подключаются не стандартным способом (не через <script src='…'></script>). Подключение сделано более удобным способом: код из этого файла подключается прямо на странице после формы настроек, при этом в скрипте доступы следующие переменные: id (идентификатор текущего плагина), hid (идентификатор текущего плагина в виде #id), div (jquery-объект блока плагина), code (код плагина), title (заголовок плагина).

В плагине не допускается использования событие типа $(document).ready() и $(window).load() – для того чтобы выполнить какие-то действия при показе плагина, требуется добавить обработчик события pluginload, например:

```
$(document).delegate(hid, 'pluginload', function(e){
	$('select[data-role="my_select"]', div).trigger('change');
});
```

При этом события для элементов управления следует назначать только через $(document).delegate():

```
$(document).delegate(hid+' input[data-role="my_button"]', 'click', function(e){
	e.preventDefault();
	alert('Test');
});
```

Файл style.css

Как и script.js, этот файл подключается не стандартным способом – он подключается перед формой настроек плагина.

Учитывая, что на странице может быть показано сразу несколько форм различных плагинов, критически важно стили не пересекались, поэтому все стили необходимо прописывать в собственном пространстве имен, например:

```
div[data-plugin-form="SET_VALUES"] textarea {
	font-family:'Courier New', 'Courier', monospace;
	min-height:60px;
	min-width:300px;
	resize:vertical;
}
```

Здесь важно учитывать, что коды плагинов преобразуются, т.е., например, если класс плагина MySuperPlugin, то код будет MY_SUPER_PLUGIN.

Файл .description.php

Этот файл содержит html-содержимое для описания плагина (то, что появляется при нажатии кнопки «Справка»). Содержимое файла всегда в кодировке UTF-8, даже на сайтах, работающих в кодировке windows-1251. Учитывая, что файл подключается в статическом режиме, переменная $this здесь недоступна, обратиться можно только к нестатическим плагинам, например:

```
static::getName();
```

Наследование плагинов

В модуле имеется возможность наследования плагинов. Это необходимо редко, но все же может пригодиться. Например, если разработать плагин, унаследовав его от плагина логирования, можно создать свой плагин для логирования своих данных. Например, таким способом можно искать битые картинки: получается, что алгоритм поиска будет свой, а сохранение в файл уже готово и это не потребуется разрабатывать.

Пример наследования:

```
namespace WD\Antirutin\Plugins\Element;

use
	\WD\Antirutin\Helper,
	\WD\Antirutin\IBlock,
	\WD\Antirutin\PluginElement;

require_once($_SERVER['DOCUMENT_ROOT'].
'/bitrix/modules/webdebug.antirutin/plugins/element/logger/class.php');

class LoggerX extends Logger {

	protected function processElement($intElementId){
		return parent:: processElement($intElementId);
	}


	protected function prepareCss(&$strCss){
		$strCss = str_replace('div[data-plugin-form="LOGGER"]', 'div[data-plugin-
form="LOGGER_X"]', $strCss);
	}

}
```

Логи модуля

Модуль хранит свои логи в папке /upload/webdebug.antirutin/log (по каждому профилю свой лог-файл). Логируются только ошибки.

Техподдержка

В случае возникновения каких-либо вопросов или проблем – пожалуйста, обращайтесь к нам:

Email: info@webdebug.ru
Skype: webdebug
Telegram: webdebug
WhatsApp/Viber: +79882410850
ICQ: 932767

Задать вопрос