Фильтр производителей

```
<?php

defined('HOSTCMS') || exit('HostCMS: access denied.');

/**
* v. 1.2
*для HostCMS v.6x
* @author KAD
* http://www.artemkuts.ru/
* artem.kuts@gmail.com
*/

/*
*В ТДС перед вызовом show()
* Kad_Shop_Producer_Filter::instance()->execute($Shop_Controller_Show);
*
*В xsl для флажков

  <!-- Шаблон для фильтра производителей ФЛАЖКИ-->
    <xsl:template match="producers/shop_producer">

        <xsl:variable name="id" select="@id"/>
        <input type="checkbox" name="producer[]" value="{@id}"
id="producer_{@id}">

            <xsl:if
test="/shop/filters/used/producers[producer=$id]/node()">
                <xsl:attribute name="checked">checked</xsl:attribute>
            </xsl:if>

        </input>
    <label for="producer_{@id}"><a href="javascript:void(0)" onclick="
$('[id^=producer_]').prop('checked', false);
$('#producer_{@id}').prop('checked',
true);$('#producer_{@id}').closest('form').submit();"><xsl:value-of
select="name"/></a></label>

    </xsl:template>

*В xsl для списка

  <!-- Шаблон для фильтра производителей СПИСОК-->
    <xsl:template match="producers/shop_producer">

        <xsl:variable name="id" select="@id"/>
        <option value="{@id}">
```

```
            <xsl:if
test="/shop/filters/used/producers[producer=$id]/node()">
                <xsl:attribute name="selected">selected</xsl:attribute>
            </xsl:if>
            <xsl:value-of select="name"/>
        </option>
    </xsl:template>

* Вызов в шаблоне

    <select name="producer">
        <option>...</option>
        <xsl:apply-templates select="/shop/producers/shop_producer"/>
    </select>

* Далее заменяем код переменной filter

<!-- Передаем фильтр -->
<xsl:variable name="filter"><xsl:if
test="/shop/filter/node()">?filter=1&amp;sorting=<xsl:value-of
select="/shop/sorting"/>&amp;price_from=<xsl:value-of
select="/shop/price_from"/>&amp;price_to=<xsl:value-of
select="/shop/price_to"/><xsl:for-each select="/shop/*"><xsl:if
test="starts-with(name(), 'property_')">&amp;<xsl:value-of
select="name()"/>=<xsl:value-of select="."/></xsl:if></xsl:for-
each></xsl:if><xsl:if test="/shop/filters/used/producers/node()"><xsl:for-
each select="/shop/filters/used/producers/producer">&amp;<xsl:value-of
select="name()"/>[]=<xsl:value-of select="."/></xsl:for-
each></xsl:if></xsl:variable>

*
*/
class Kad_Shop_Producer_Filter
{

    public $input_name = 'producer';

    /**
     * The singleton instances.
     * @var mixed
     */
    static public $instance = NULL;

    /**
     * Register an existing instance as a singleton.
     * @return object
     */
    static public function instance()
    {
        if (is_null(self::$instance))
        {
```

```
            self::$instance = new self();
        }

        return self::$instance;
    }

    public function __construct()
    {

    }

    /*
  * Получаем производителей
    */
    public function getProducers()
    {
        $aShop_Producers =
Kad_Shop_Producer::getAll($this->Shop_Controller_Show->group, true);

        return $aShop_Producers;
    }

    /*
  * Добавляем производителей к xml
    */
    public function addToXml($aShopProducers, $aActiveProducers)
    {
        $oProducersXmlEntity =
Core::factory('Core_Xml_Entity')->name('producers');

        $aShopProducers = $this->getProducers();
        foreach ($aShopProducers as $oShopProducer)
        {
            $oProducersXmlEntity->addEntity(
                $oShopProducer->clearEntities()
            );
        }

        $this->Shop_Controller_Show->addEntity($oProducersXmlEntity);

        if ($aActiveProducers)
        {
            $xmlFilters = Core::factory('Core_Xml_Entity')->name('filters');
            $xmlUsed = Core::factory('Core_Xml_Entity')->name('used');
            $oActiveProducersXmlEntity =
Core::factory('Core_Xml_Entity')->name('producers');

            foreach ($aActiveProducers as $ActiveProducer)
            {
                $oProducerXml =
Core::factory('Core_Xml_Entity')->name($this->input_name)->value($ActiveProd
```

```
ucer);
                $oActiveProducersXmlEntity->addEntity($oProducerXml);
            }

$this->Shop_Controller_Show->addEntity($xmlFilters->addEntity($xmlUsed->addE
ntity($oActiveProducersXmlEntity)));
        }

        return true;
    }

    /*
  * Фильтр по производителям
    */
    public function filterProducers($aProducers = array())
    {

        $this->Shop_Controller_Show->shopItems()
            ->queryBuilder()
            ->where('shop_items.shop_producer_id', 'IN', $aProducers);

        $this->Shop_Controller_Show->addCacheSignature( 'producer=' .
implode(',', $aProducers));
    }

    public function execute($Shop_Controller_Show = Shop_Controller_Show)
    {
        $this->Shop_Controller_Show = $Shop_Controller_Show;

        $aProducers = Core_Array::getGet($this->input_name);

        if ($aProducers && !is_array($aProducers))
        {
            $aProducers = array($aProducers);
        }

        if (count($aProducers))
        {
            $this->filterProducers($aProducers);
        }

        $aShop_Producers =
Kad_Shop_Producer::getAll($this->Shop_Controller_Show->group);
        $this->addToXml($aShop_Producers, $aProducers);

    }
}
```