Документация на модуль abcwww.image для CMS 1С-Битрикс

Поделиться ссылкой в соцсетях

Преимущества:

1. В качестве исходного изображения можно передавать разные варианты и модуль поймет их:
    - Локальная ссылка на изображение (относительно корня сайта)
    - Абсолютная ссылка на изображение (даже если файл лежит за пределами $_SERVER['DOCUMENT_ROOT'], главное что бы файл был доступен для PHP)
    - Относительная ссылка на изображение (относительно пути PHP-скрипта который вызвал метод)
    - ID-файла из таблицы зарегистрированных файлов в 1С-Битрикс
    - Массив описывающий файл, например $arResult['PREVIEW_PICTURE'] или массив возвращаемый методом \CFile::GetFileArray()
    - Массив возвращаемый методом \CFile::ResizeImageGet()
2. Возможность конвертации изображения в другой формат (который поддерживает библиотека GD той версии что установлена на вашем сервере)
3. Возможность пропорционального ресайза изображения, где в качестве размера достаточно указать всего один параметр (ширина или высота)
4. При конвертации сохраняется имя исходного файла (просто новый файл сохраняется в другую папку, но с таким же именем)
5. Конвертация происходит всего один раз, далее будет выводиться сконвертированный файл из кэша
6. Если изменится исходное изображение, то автоматически перегенерируется результирующее
7. Возможность вывода исходного файла без конвертации и ресайза
8. Автоматическая генерация тега <picture>
9. Поддержка media для различных разрешений экрана (можно вывести разные изображения для разных разрешений)
10. Поддержка плотности экрана (автоматически выведется нужное изображение (с учетом указанных плотностей) оптимальное для плотности экрана устройства)
11. Поддержка микроразметки schema.org/ImageObject

Установка:

Стандартная установка модуля из MarketPlace для CMS 1С-Битрикс (ссылка на модуль). Модуль не имеет ни каких настроек, он состоит из класса с методами для работы с изображениями.

Для того что бы были доступны методы модуля, его необходимо подключить.

```
<?\Bitrix\Main\Loader::includeModule("abcwww.image")?>
```

Если необходимо что бы он был доступен на всех страницах, то подключите его в init.php

Методы:

- \Bitrix\Abcwww\Image::setDefault($params = [])
- \Bitrix\Abcwww\Image::getPicture($image, $params = [])
- \Bitrix\Abcwww\Image::getImg($image, $params = [])
- \Bitrix\Abcwww\Image::getArResize($image, $params = [])
- \Bitrix\Abcwww\Image::getArWebp($image, $params = [])
- \Bitrix\Abcwww\Image::getResize($image, $params = [])
- \Bitrix\Abcwww\Image::getWebp($image, $params = [])

Метод \Bitrix\Abcwww\Image::setDefault($params = [])

Назад к методам

Задает дефолтные параметры, которые применятся к методам, если они не будут переданы при вызове методов.

В данный момент доступен единственный ключ QUALITY, который задаст дефолтное значение для качества генерируемых изображений, число от 1 до 100.

Если вы хотите что бы ваше дефолтное значение было установлено всегда, то вызовите этот метод сразу после подключения модуля, и этот вызов должен всегда присутствовать в коде.

Изначально качество изображения установлено 100%, вы можете поменять на любое другое:

```
<?=\Bitrix\Abcwww\Image::setDefault(["QUALITY" => 80])?>
```

ВНИМАНИЕ!!! для каждого сгенерированного изображения в папке /upload/image_cache/ создается файл, путь до которого зависит от указанного качества для него. Например если ранее качество было 100%, а вы сделали 80%, то сгенерируются новые файлы, поэтому рекомендуется очистить папку /upload/image_cache/ и сбросить кэш Битрикс. Это связано с тем что на разных страницах вы можете вывести одно и то же изображение но с разным качеством, поэтому и пути до картинок с разным качеством разные.

Метод \Bitrix\Abcwww\Image::getPicture($image, $params = [])

Назад к методам

$image Обязательный параметр: путь к изображению или массив данных изображения (описано в преимуществах, п.1).

$params Необязательный параметр: массив настроек обработки изображения.

В качестве результата вернется HTML-код тега <picture> с вложенными в него тегами <source> и <img>. Метод удобен для вывода изображений на странице с учетом адаптивной верстки и плотности экрана. Для совместимости со всеми браузерами, в теге <picture> будут лежать ресурсы как в формате webp, так и в формате оригинального изображения, на тот случай, что бы браузер выбрал сам нужный ресурс, формат которого он поддерживает.

Структура $params:

```
<?
$params = [
    "WIDTH" => 1000, //число (ограничить по ширине)
    "HEIGHT" => 1000, //число (ограничить по высоте)
    "QUALITY" => 90, //число от 1 до 100 (качество изображения)
    "DENSITY" => [1.5, 2], //массив с дополнительными плотностями экрана
    "CONVERT" => "image/jpeg", //тип изображения в котором его вывести (не нужно указывать "image/webp", он и так будет выведен дополнительно, так же не нужно указывать "image/jpeg", если у вас исходное изображение "image/jpeg", просто не передавайте этот параметр)
    "ATTRIBUTES" => [ //атрибуты для тега img
        "class" => "my-class",
        "loading" => "lazy",
    ],
    "SHOW_SIZE" => true, //выведет у тега img атрибуты width и height с реальными данными
    "SCHEMA" => [
        "NAME" => "Название картинки", //значение для itemprop="name"
        "DESCRIPTION" => "Описание картинки", //значение для itemprop="description"
    ],
    "SHOW_ORIGINAL_SRC" => true, //выведет у тега img в src путь до оригинального изображения
    "RESPONSIVE" => [
        [
            "MEDIA" => "(max-width: 767px)", //медиа запрос
            "IMAGE" => "/img/phone.jpg", //изображение, формат как у параметра $image
            "WIDTH" => 200, //число (ограничить по ширине)
            "HEIGHT" => 200, //число (ограничить по высоте)
            "QUALITY" => 90, //число от 1 до 100 (качество изображения)
            "DENSITY" => [2], //массив с дополнительными плотностями экрана
        ],
        ...
        [
            ...
        ],
    ],
]
```

```
?>
```

Подробное описание каждого элемента массива $params:

WIDTH и HEIGHT задаются в виде целого числа (без px), это число означает размер в пикселях. Они могут использоваться как вместе, так и по отдельности. Если указан один из этих параметров, то произойдет сравнение с аналогичным размером исходного изображения. И если указанный параметр меньше значения оригинала, то произойдет пропорциональное уменьшение изображения, иначе размеры останутся оригинальными. Если указаны 2 параметра, то результирующее изображение уменьшится так же пропорционально, до таких размеров, что бы оно поместилось в прямоугольник с размерами WIDTH x HEIGHT. Работает аналогично стилю в CSS background-size: contain; Если ни один параметр не задан (WIDTH и HEIGHT), то результирующее изображение создастся с оригинальными размерами.

QUALITY качество результирующего изображения, число от 1 до 100, если не указан, то будет выбрано дефолтное значение.

DENSITY если вы хотите выводить изображения с учетом плотности экрана устройства, то задайте массив плотностей, например [1.5, 2]. Не рекомендуется задавать большое количество вариантов, учтите что для каждой плотности будет создано отдельное изображение. Рекомендую ограничиться одним значением [2]. Не нужно дополнительно добавлять значение 1, для него и так будут созданы изображения по умолчанию.

> DENSITY применится только если задан WIDTH или HEIGHT и они меньше аналогичных параметров исходного изображения. Иначе он проигнорируется.

CONVERT конвертирует результирующее изображение в другой формат. Например у вас исходное изображение в формате image/png, а вы указали в этом параметре image/jpeg, тогда в <source> попадут ссылки на изображения в форматах image/jpeg и image/webp, иначе (без указания этого параметра) image/png и image/webp.

> ВНИМАНИЕ!!! не нужно указывать image/webp, он и так будет выведен дополнительно.
>
> Если у вас изображение в том же формате, в котором вы хотите его вывести, то не передавайте этот параметр.

ATTRIBUTES задается массив атрибутов, в формате "ключ" => "значение" которые выведутся в теге <img>, например ["class" => "my-class", "loading" => "lazy"].

> Если информация о изображении хранится в базе, например это анонсное или детальное изображение элемента инфоблока или свойство тип файл элемента инфоблока и у него в поле с описанием задано значение, то вы можете указать чтобы это описание автоматически подставлялось в атрибут alt или title тега <img>. Для этого в качестве $image передавайте ID файла из таблицы b_file или массив аналогичный CFile::GetFileArray() (например $arResult['PREVIEW_PICTURE']), а в параметре $params['ATTRIBUTES']['alt'] или $params['ATTRIBUTES']['title'] укажите true.

SHOW_SIZE если передан этот параметр со значением true то в тег <img> добавятся атрибуты width и height с реальными значениями ширины и высоты этого изображения.

SCHEMA если вы хотите добавить микроразметку schema.org/ImageObject для тега <picture>, то передайте массив в формате ["NAME" => "Название картинки", "DESCRIPTION" => "Описание картинки"].

SHOW_ORIGINAL_SRC если передан этот параметр со значением true то в атрибуте src тега <img> выведется путь до оригинального изображения. Это полезно когда вы добавляете микроразметку schema.org/ImageObject.

RESPONSIVE массив состоящий из массивов параметров для изображений которые будут выводиться для разных разрешений экрана.

Учтите что внутри тэга <picture> сначала будут выводится теги <source> на основе данных из массива $params['RESPONSIVE'] в том же порядке, а в самом низу на основе основных параметров $params, поэтому внимательно отнеситесь к этому параметру.

Каждый отдельный массив внутри RESPONSIVE может иметь следующие параметры:

MEDIA (обязательный) в нем указывается медиа запрос для разрешения экрана при котором вывести изображение с этими параметрами, например (max-width: 767px).

В теге <picture> условия MEDIA работают немного по другому чем в CSS. Если в CSS подходящие условия расположенные ниже по коду перебивают предыдущие, то в <picture> наоборот, выбирается самый первый подходящий вариант, а все остальные (что ниже) игнорируются.

Рекомендую всегда в MEDIA использовать условие max-width от меньшего к большему, это снизит нагрузку на телефонах т.к. условие для показа изображения для маленьких экранов будет в самом верху.

IMAGE структура такая же как и у основного параметра $image, если вам нужно вывести другое изображение (отличное от основного) то кажите этот параметр.

WIDTH, HEIGHT, QUALITY, DENSITY аналогично значениям основного параметра $params.

Если QUALITY не указан, то значение возьмется из основного параметра $params['QUALITY'].

Для удобства можно использовать сокращенные ключи для параметра $params, это первые буквы ключей массива.

Внимание!!! Сокращенные ключи не для всех, вот полный список:

```
<?
$params = [
   "W" => 1000, //число (ограничить по ширине)
   "H" => 1000, //число (ограничить по высоте)
   "Q" => 90, //число от 1 до 100 (качество изображения)
```

```
    "D" => [1.5, 2], //массив с дополнительными плотностями экрана
    "C" => "image/jpeg", //тип изображения
    "A" => [ //атрибуты для тега img
        "class" => "my-class",
        "loading" => "lazy",
    ],
    "SHOW_SIZE" => true, //выведет у тега img атрибуты width и height с реальными данными
    "S" => [
        "N" => "Название картинки", //значение для itemprop="name"
        "D" => "Описание картинки", //значение для itemprop="description"
    ],
    "SHOW_ORIGINAL_SRC" => true, //выведет у тега img в src путь до оригинального изображения
    "R" => [
        [
            "M" => "(max-width: 767px)", //медиа запрос
            "I" => "/img/phone.jpg", //изображение, формат как у параметра $image
            "W" => 200, //число (ограничить по ширине)
            "H" => 200, //число (ограничить по высоте)
            "Q" => 90, //число от 1 до 100 (качество изображения)
            "D" => [2], //массив с дополнительными плотностями экрана
        ],
        ...
        [
            ...
        ],
    ],
]

?>
```

Примеры использования \Bitrix\Abcwww\Image::getPicture($image, $params = []):

Самый простой вариант, выводим изображение как есть:

```
<?=\Bitrix\Abcwww\Image::getPicture("/example/abcwww-image/images/news-list.jpg")?>
```

Самый распространенный вариант, когда нужно просто вывести изображение ограничив его по ширине 300px, с указанием CSS-класса и alt и добавив двойную плотность экрана, что бы на телефонах изображение было более четким, например для списка новостей:

```
<?=\Bitrix\Abcwww\Image::getPicture("/example/abcwww-image/images/news-list.jpg", [
   "W" => 300,
   "D" => [2],
   "A" => [
      "alt" => "Пример использования abcwww.image",
      "class" => "my-class",
   ],

])?>
```

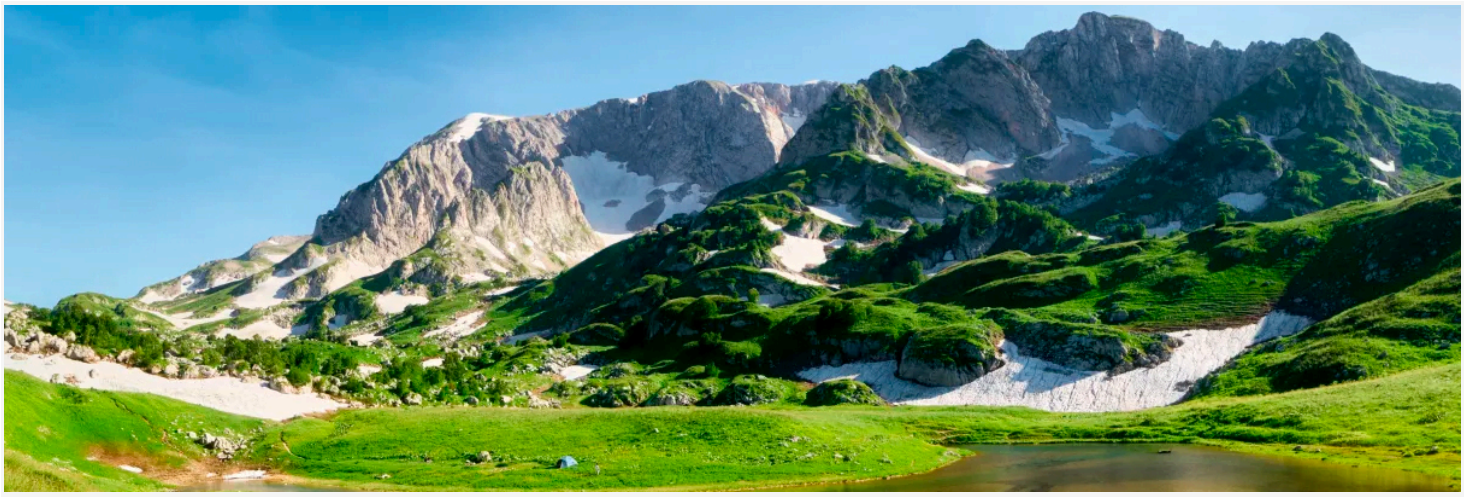
Сложный вариант, с такими условиями:

- Покажем для телефонов изображение отличное от десктопа
- Укажем для телефонов вариант с двойной плотностью экрана
- Для устройств с шириной экрана ниже 1024px покажем изображение для десктопа, так же зададим двойную плотность
- Изображение для десктопа у нас image/png а мы хотим вывести image/jpeg
- Выведем атрибуты width и height
- Сделаем качество всех картинок 90%
- Выведем микроразметку
- Сделаем так что бы в микроразметку попал путь до оригинального изображения, т.к. оно самое качественное и большое

```
<?=\Bitrix\Abcwww\Image::getPicture("/example/abcwww-image/images/pc.png", [
  "R" => [
    [
      "M" => "(max-width: 425px)", //Покажем это изображение для экранов 425px и ниже
      "I" => "/example/abcwww-image/images/phone.jpg", //Изображение для телефона отличное от десктопа
      "W" => 425, //Ограничиваем до 425px
      "D" => [2], //Двойная плотность экрана
    ],
    [
      //Т.к. "I" не задан то возьмётся "/example/abcwww-image/images/pc.png"
      "M" => "(max-width: 1023px)", //Покажем это изображение для экранов ниже 1024px
      "W" => 1023, //Ограничиваем до 1023px
      "D" => [2], //Двойная плотность экрана
    ],
  ],
  "W" => 1920, //Ограничиваем до 1920px
```

```
  "Q" => 90, //Качество 90%
  "C" => "image/jpeg", //Конвертируем в JPEG
  "A" => [ //Атрибуты для тега img
    "alt" => "Пример использования abcwww.image", //Выведем атрибут alt
    "class" => "my-class", //Установим CSS-класс для img
    "loading" => "lazy", //Ленивая загрузка
  ],
  "SHOW_SIZE" => true, //Выведем атрибуты width и height у img
  "SHOW_ORIGINAL_SRC" => true, //Подставим для img атрибут src
"/example/abcwww-image/images/pc.png"
  "S" => [ //Выведем микроразметку
    "N" => "Пример использования abcwww.image", //itemprop="name"
    "D" => "Демонстрация возможностей модуля abcwww.image", //itemprop="description"
  ],

])?>
```



Метод \Bitrix\Abcwww\Image::getImg($image, $params = [])

Назад к методам

Параметры аналогичные параметрам метода \Bitrix\Abcwww\Image::getPicture($image, $params = []), только набор ключей массива $params немного меньше.

В качестве результата вернется HTML-код тега <img>.

Структура $params:

```
<?
$params = [
  "WIDTH" => 1000, //число (ограничить по ширине)
  "HEIGHT" => 1000, //число (ограничить по высоте)
```

```
    "QUALITY" => 90, //число от 1 до 100 (качество изображения)
    "DENSITY" => [1.5, 2], //массив с дополнительными плотностями экрана
    "CONVERT" => "image/jpeg", //тип изображения в котором его вывести
    "ATTRIBUTES" => [ //атрибуты для тега img
        "class" => "my-class",
        "loading" => "lazy",
    ],
    "SHOW_SIZE" => true, //выведет у тега img атрибуты width и height с реальными данными
  ]

?>
```

Пример использования \Bitrix\Abcwww\Image::getImg($image, $params = []):

```
<?=\Bitrix\Abcwww\Image::getImg("/example/abcwww-image/images/news-list.jpg", [
  "W" => 300,
  "D" => [2],
  "A" => [
      "alt" => "Пример использования abcwww.image",
      "class" => "my-class",
  ],

])?>
```

Метод \Bitrix\Abcwww\Image::getArResize($image, $params = [])

Назад к методам

Параметры аналогичные параметрам метода \Bitrix\Abcwww\Image::getImg($image, $params = []), только набор ключей массива $params немного меньше.

В качестве результата вернется массив описывающий файл.

Структура $params:

```
<?
$params = [
    "WIDTH" => 1000, //число (ограничить по ширине)
    "HEIGHT" => 1000, //число (ограничить по высоте)
    "QUALITY" => 90, //число от 1 до 100 (качество изображения)
    "CONVERT" => "image/jpeg", //тип изображения в котором его вывести
    "SHOW_SIZE" => true, //вернет в массиве атрибуты width и height с реальными данными
]

?>
```

Пример использования \Bitrix\Abcwww\Image::getArResize($image, $params = []):

```
<?
$arr = \Bitrix\Abcwww\Image::getArResize("/example/abcwww-image/images/news-list.png", [
    "W" => 300,
    "Q" => 90,
    "C" => "image/jpeg",
    "SHOW_SIZE" => true,
]);

?>
```

Вернется массив с данными:

```
<?
array(9) {
  ["SRC"]=>
  string(78) "/upload/image_cache/8e/8ee5e7b963dbf78b2eefc9f0f39ce829_w300_q90/news-list.jpg"
  ["SRC_ABS"]=>
  string(109)
"/data/current/web-tool.org/docs/upload/image_cache/8e/8ee5e7b963dbf78b2eefc9f0f39ce829_w300_q90/
news-list.jpg"
  ["CONTENT_TYPE"]=>
  string(10) "image/jpeg"
  ["ORIGINAL_SRC"]=>
  string(42) "/example/abcwww-image/images/news-list.png"
  ["ORIGINAL_SRC_ABS"]=>
  string(73) "/data/current/web-tool.org/docs/example/abcwww-image/images/news-list.png"
  ["DESCRIPTION"]=>
```

```
  string(0) ""
  ["TIMESTAMP"]=>
  int(1746148139)
  ["FILE_SIZE"]=>
  int(45387)
  ["ATTRIBUTES"]=>
  array(2) {
    ["width"]=>
    int(300)
    ["height"]=>
    int(211)
  }
}

?>
```

Хотя в названии метода присутствует слово Resize, это не означает что он предназначен только для ресайза. Если не задать ограничение по размерам, или задать но больше чем у оригинала, так же не задать качество или задать 100% и не задавать конвертацию, то метод выведет информацию оригинального файла.

```
<?
$arr = \Bitrix\Abcwww\Image::getArResize("/example/abcwww-image/images/news-list.jpg", [
    "SHOW_SIZE" => true,
]);

?>
```

Вернется массив с данными:

```
<?
array(9) {
  ["SRC"]=>
  string(42) "/example/abcwww-image/images/news-list.jpg"
  ["SRC_ABS"]=>
  string(73) "/data/current/web-tool.org/docs/example/abcwww-image/images/news-list.jpg"
  ["CONTENT_TYPE"]=>
  string(10) "image/jpeg"
  ["ORIGINAL_SRC"]=>
  string(42) "/example/abcwww-image/images/news-list.jpg"
  ["ORIGINAL_SRC_ABS"]=>
  string(73) "/data/current/web-tool.org/docs/example/abcwww-image/images/news-list.jpg"
  ["DESCRIPTION"]=>
```

```
  string(0) ""
  ["TIMESTAMP"]=>
  int(1721541039)
  ["FILE_SIZE"]=>
  int(661138)
  ["ATTRIBUTES"]=>
  array(2) {
    ["width"]=>
    int(768)
    ["height"]=>
    int(540)
  }
}

?>
```

Метод \Bitrix\Abcwww\Image::getArWebp($image, $params = [])

Назад к методам

Частный случай метода \Bitrix\Abcwww\Image::getArResize($image, $params = []).

Сконвертирует изображение в формат WebP и вернет массив описывающий файл.

Структура $params:

```
<?
$params = [
    "WIDTH" => 1000, //число (ограничить по ширине)
    "HEIGHT" => 1000, //число (ограничить по высоте)
    "QUALITY" => 90, //число от 1 до 100 (качество изображения)
    "SHOW_SIZE" => true, //вернет в массиве атрибуты width и height с реальными данными
]

?>
```

Пример использования \Bitrix\Abcwww\Image::getArWebp($image, $params = []):

```
<?
$arr = \Bitrix\Abcwww\Image::getArWebp("/example/abcwww-image/images/news-list.jpg", [
    "W" => 300,
    "Q" => 90,
    "SHOW_SIZE" => true,
```

```
]);

?>
```

Вернется массив с данными:

```
<?
array(9) {
 ["SRC"]=>
 string(79) "/upload/image_cache/bc/bce9fae19329b98d934288ae4f3e6a2b_w300_q90/news-list.webp"
 ["SRC_ABS"]=>
 string(110)
"/data/current/web-tool.org/docs/upload/image_cache/bc/bce9fae19329b98d934288ae4f3e6a2b_w300_q9
0/news-list.webp"
 ["CONTENT_TYPE"]=>
 string(10) "image/webp"
 ["ORIGINAL_SRC"]=>
 string(42) "/example/abcwww-image/images/news-list.jpg"
 ["ORIGINAL_SRC_ABS"]=>
 string(73) "/data/current/web-tool.org/docs/example/abcwww-image/images/news-list.jpg"
 ["DESCRIPTION"]=>
 string(0) ""
 ["TIMESTAMP"]=>
 int(1746067928)
 ["FILE_SIZE"]=>
 int(29580)
 ["ATTRIBUTES"]=>
 array(2) {
   ["width"]=>
   int(300)
   ["height"]=>
   int(211)
 }
}

?>
```

Метод \Bitrix\Abcwww\Image::getResize($image, $params = [])

Назад к методам

Вернет путь до изображения.

Структура $params:

```php
<?
$params = [
    "WIDTH" => 1000, //число (ограничить по ширине)
    "HEIGHT" => 1000, //число (ограничить по высоте)
    "QUALITY" => 90, //число от 1 до 100 (качество изображения)
    "CONVERT" => "image/jpeg", //тип изображения в котором его вывести
]

?>
```

Пример использования \Bitrix\Abcwww\Image::getResize($image, $params = []):

```php
<?
$arr = \Bitrix\Abcwww\Image::getResize("/example/abcwww-image/images/news-list.png", [
    "W" => 300,
    "Q" => 90,
    "C" => "image/jpeg",
]);

?>
```

Вернется путь до изображения:

```php
<?
string(78) "/upload/image_cache/8e/8ee5e7b963dbf78b2eefc9f0f39ce829_w300_q90/news-list.jpg"

?>
```

Метод \Bitrix\Abcwww\Image::getWebp($image, $params = [])

Назад к методам

Частный случай метода \Bitrix\Abcwww\Image::getResize($image, $params = []).

Сконвертирует изображение в формат WebP и вернет до него путь.

Структура $params:

```php
<?
$params = [
    "WIDTH" => 1000, //число (ограничить по ширине)
    "HEIGHT" => 1000, //число (ограничить по высоте)
    "QUALITY" => 90, //число от 1 до 100 (качество изображения)
]
```

```
?>
```

Пример использования \Bitrix\Abcwww\Image::getWebp($image, $params = []):

```
<?
$arr = \Bitrix\Abcwww\Image::getWebp("/example/abcwww-image/images/news-list.jpg", [
    "W" => 300,
    "Q" => 90,
]);

?>
```

Вернется путь до WebP:

```
<?
string(79) "/upload/image_cache/bc/bce9fae19329b98d934288ae4f3e6a2b_w300_q90/news-list.webp"

?>
```