Import Statement

```
use
Magento\Catalog\Api\ProductRepositoryInterface;
use Magento\Catalog\Api\Data\ProductInterfaceF-
actory;
```

Constructor

```
/**
 * @var ProductRepositoryInterface
 */
protected $productRepositoryInterface;
/**
 * @var ProductInterfaceFactory
 */
protected $productInterfaceFactory;
/**
 * @param ProductRepositoryInterface $productRepo-
sitoryInterface
 */
public function __construct(
    ProductRepositoryInterface $productRepository-
Interface,
    ProductInterfaceFactory $productInterfaceF-
actory,
) {
    $this->productRepositoryInterface = $productR-
epositoryInterface;
    $this->productInterfaceFactory = $productInte-
rfaceFactory;
}
```

Product Types

```
\Magento\Catalog\Model\Product\Type::TYPE_SIMPLE
\Magento\Catalog\Model\Product\Type::TYPE_BUNDLE
\Magento\Catalog\Model\Product\Type::TYPE_VIRTUAL
\Magento\Downloadable\Model\Product\Type::TYPE_D-
OWNLOADABLE
\Magento\ConfigurableProduct\Model\Product\Type-
\Configurable::TYPE_CODE
\Magento\GroupedProduct\Model\Product\Type\Group-
ed::TYPE_CODE
```

Get info about product by product SKU

```
$product = $this->productRepositoryInterface-
>get($sku);
```

```
/**
* @param string $sku
* @param bool $editMode
* @param int|null $storeId
* @param bool $forceReload
* @return \Magento\Catalog\Api\Data\ProductInterface
* @throws \Magento\Framework\Exception\NoSuchEntityException
*/
```

Get info about product by product id

```
$product = $this->productRepositoryInterface-
>getById($productId);
```

```
/**
* @param int $productId
* @param bool $editMode
* @param int|null $storeId
* @param bool $forceReload
* @return \Magento\Catalog\Api\Data\ProductInterface
* @throws \Magento\Framework\Exception\NoSuchEntityException
*/
```

By **DominicXigen**

cheatography.com/dominicxigen/

Published 2nd April, 2020.
Last updated 2nd May, 2020.
Page 1 of 6.

Delete by SKU

```
$this->productRepositoryInterface-
>deleteById($sku);
```

```
/**
* @param string $sku
* @return bool Will returned True if deleted
* @throws \Magento\Framework\Exception\NoSuchEntityException
* @throws \Magento\Framework\Exception\StateException
*/
```

Create New Product

```
$product = $this->productInterfaceFactory
    ->create()
    ->setSku('SKU');
// other required attributes
$product = $this->productRepositoryInterface->s-
ave($product);
```

Update Existing Product

```
$product = $this->productRepositoryInterface-
>get('SKU');
$product->setStatus(\Magento\Catalog\Model\Produ-
ct\Attribute\Source\Status::STATUS_DISABLED);
$updated = $this->productRepositoryInterface->s-
ave($product);
```

Advanced Search Filter Group

```
/**
 * @var \Magento\Catalog\Api\ProductRepositoryI-
nterface
 */
private $productRepository;
/**
 * @var \Magento\Framework\Api\SearchCriteriaBu-
ilder
 */
private $searchCriteriaBuilder;
```

Advanced Search Filter Group (cont)

```
/**
 * @var \Magento\Framework\Api\FilterBuilder
 */
private $filterBuilder;
/**
 * @var \Magento\Framework\Api\Search\FilterGro-
upBuilder
 */
private $filterGroupBuilder;
/**
 * @var \Magento\Framework\Api\SortOrderFactory
 */
private $sortOrder;
/**
 * @param \Magento\Catalog\Api\ProductRepositoryI-
nterface $productRepository
 * @param \Magento\Framework\Api\SearchCriteriaBu-
ilder $searchCriteriaBuilder
 * @param \Magento\Framework\Api\FilterBuilder
$filterBuilder
 * @param \Magento\Framework\Api\Search\FilterGro-
upBuilder $filterGroupBuilder
 * @param \Magento\Framework\Api\SortOrderFactory
$sortOrder
 */
public function __construct(
    \Magento\Catalog\Api\ProductRepositoryInte-
rface $productRepository,
    \Magento\Framework\Api\SearchCriteriaBuilder
$searchCriteriaBuilder,
    \Magento\Framework\Api\FilterBuilder $filte-
rBuilder,
    \Magento\Framework\Api\Search\FilterGroupB-
uilder $filterGroupBuilder,
```

By **DominicXigen**

cheatography.com/dominicxigen/

Published 2nd April, 2020.
Last updated 2nd May, 2020.
Page 2 of 6.

Advanced Search Filter Group (cont)

```
    \Magento\Framework\Api\SortOrderFactory
$sortOrder
) {
    $this->productRepository = $productRepository;
    $this->searchCriteriaBuilder = $searchCrite-
riaBuilder;
    $this->filterBuilder = $filterBuilder;
    $this->filterGroupBuilder = $filterGroupBui-
lder;
    $this->sortOrder = $sortOrder;
}
[...]
$filter1 = $this->filterBuilder
    ->setField("url")
    ->setValue("%magento.com")
    ->setConditionType("like")
    ->create();
$filter2 = $this->filterBuilder
    ->setField("store_id")
    ->setValue("1")
    ->setConditionType("eq")
    ->create();
$filterGroup1 = $this->filterGroupBuilder
    ->setFilters([$filter1, $filter2])
    ->create();
$filter3 = $this->filterBuilder
    ->setField("url_type")
```

Advanced Search Filter Group (cont)

```
    ->setValue(1)
    ->setConditionType("eq")
    ->create();
$filterGroup2 = $this->filterGroupBuilder
    ->setFilters([$filter3])
    ->create();
$searchCriteria = $this->searchCriteriaBuilder->-
create();
$searchCriteria->setFilterGroups([$filterGroup1,
$filterGroup2]);
$this->productRepository->getList($searchCriteri-
a)->getItems();
```

The code above creates a Search Criteria with the Filters put together in the following way: (url like %magento.com OR store_id eq 1) AND (url_type eq 1)

Search Criteria

```
/**
 * @var ProductRepositoryInterface
 */
private $productRepositoryInterface;
/**
 * @var SearchCriteriaBuilder
 */
private $searchCriteriaBuilder;
/**
 * @param ProductRepositoryInterface $productRepo-
sitoryInterface
 * @param SearchCriteriaBuilder $searchCriteria-
Builder
```

Search Criteria (cont)

```
 */
public function __construct(
    ProductRepositoryInterface $productRepository-
Interface,
    SearchCriteriaBuilder $searchCriteriaBuilder
) {
    $this->productRepository = $productRepository-
Interface;
    $this->searchCriteriaBuilder = $searchCrite-
riaBuilder;
}
/**
 * Get products by ids
 * @param array $productIds
 * @return ProductInterface[]
 */
public function getProducts(array $productIds):
array
{
    $this->searchCriteriaBuilder->addFilter('enti-
ty_id', $productIds, 'in');
    $searchCriteria = $this->searchCriteriaBuilde-
r->create();
    $products = $this->productRepositoryInterf-
ace->getList($searchCriteria)->getItems();
    $count = $products->getTotalCount();
    return $products;
}
```

Useful constants

```
\Magento\Catalog\Api\Data\ProductInterface::SKU
\Magento\Catalog\Api\Data\ProductInterface::NAME
\Magento\Catalog\Api\Data\ProductInterface::PRICE
\Magento\Catalog\Api\Data\ProductInterface::WEIGHT
\Magento\Catalog\Api\Data\ProductInterface::STATUS
\Magento\Catalog\Api\Data\ProductInterface::VISI-
BILITY
\Magento\Catalog\Api\Data\ProductInterface::ATTR-
IBUTE_SET_ID
\Magento\Catalog\Api\Data\ProductInterface::T-
YPE_ID
\Magento\Catalog\Api\Data\ProductInterface::CREA-
TED_AT
\Magento\Catalog\Api\Data\ProductInterface::UPDA-
TED_AT
\Magento\Catalog\Api\Data\ProductInterface::MEDI-
A_GALLERY
\Magento\Catalog\Api\Data\ProductInterface::TIER-
_PRICE
```

Advanced Search

```
/**
 * @var \Magento\Catalog\Api\ProductRepositoryI-
nterface
 */
private $productRepository;
/**
 * @var \Magento\Framework\Api\SearchCriteriaBu-
ilder
 */
private $searchCriteriaBuilder;
/**
 * @var \Magento\Framework\Api\FilterBuilder
 */
```

Advanced Search (cont)

```
private $filterBuilder;
/**
 * @var \Magento\Framework\Api\SortOrderFactory
 */
private $sortOrder;
/**
 * @param \Magento\Catalog\Api\ProductRepositoryI-
nterface $productRepository
 * @param \Magento\Framework\Api\SearchCriteriaBu-
ilder $searchCriteriaBuilder
 * @param \Magento\Framework\Api\FilterBuilder
$filterBuilder
 * @param \Magento\Framework\Api\SortOrderFactory
$sortOrder
 */
public function __construct(
    \Magento\Catalog\Api\ProductRepositoryInte-
rface $productRepository,
    \Magento\Framework\Api\SearchCriteriaBuilder
$searchCriteriaBuilder,
    \Magento\Framework\Api\FilterBuilder $filte-
rBuilder,
    \Magento\Framework\Api\SortOrderFactory
$sortOrder
) {
    $this->productRepository = $productRepository;
    $this->searchCriteriaBuilder = $searchCrite-
riaBuilder;
    $this->filterBuilder = $filterBuilder;
    $this->sortOrder = $sortOrder;
}
/**
 * Retrieves products to be sent
```

Advanced Search (cont)

```
 * @param array $excludedIds
 * @param array $filterIds
 * @param int $page
 * @param int $pageSize
 * @return ProductInterface[]
 * @throws \Magento\Framework\Exception\InputEx-
ception
 */
public function retrieve(
    array $excludedIds,
    array $filterIds,
    int $page = 1,
    int $pageSize = 100
) : array {
    $this->searchCriteriaBuilder->addFilter('type-
_id', [
        \Magento\Catalog\Model\Product\Type::TY-
PE_SIMPLE,
        \Magento\Downloadable\Model\Product\Typ-
e::TYPE_DOWNLOADABLE
    ], 'in');
    if (count($filterIds)) {
        $this->searchCriteriaBuilder->addFilter-
('entity_id', $filterIds, 'in');
    }
    if (count($excludedIds)) {
        $this->searchCriteriaBuilder->addFilter-
('entity_id', $excludedIds, 'nin');
    }
    $sortOrder = $this->sortOrder->setField('up-
dated_at')
```

By **DominicXigen**

cheatography.com/dominicxigen/

Published 2nd April, 2020.
Last updated 2nd May, 2020.
Page 5 of 6.

Advanced Search (cont)

```
        ->setDirection(\Magento\Framework\Api\Sort-
Order::SORT_DESC);
    $this->searchCriteriaBuilder->setSortOrders([-
$sortOrder]);
    $this->searchCriteriaBuilder->setCurrentPage(-
$page);
    $this->searchCriteriaBuilder->setPageSize($pa-
geSize);
    $searchCriteria = $this->searchCriteriaBuilder-
>create();
    return $this->productRepository->getList($sear-
chCriteria)->getItems();
}
```

Useful Constants

```
\Magento\Framework\Api\SortOrder::SORT_ASC
\Magento\Framework\Api\SortOrder::SORT_DESC
```

Print Query String

```
$products = $this->productRepositoryInterface-
>getList($searchCriteria)
echo $products->getSelect()->assemble();
```