Magento Core API

This file documents the Magento Core API for Magento Community Edition 1.6.1.

Resource: core_store

Method: list

```
$client->call($sessionId, 'core_store.list');
```

Implemented In

```
Mage_Core_Model_Store_Api::items
/Users/alanstorm/Sites2011/magento1point6point1.dev/app/code/core/Mage/Core/Model/Store/Api.php
```

Doc Comment

```
/**
 * Retrieve stores list
 *
 * @return array
 */
```

Method Implementation Definition

```
    public function items()
    {
        // Retrieve stores
        $stores = Mage::app()->getStores();

        // Make result array
        $result = array();
        foreach ($stores as $store) {
            $result[] = array(
                'store_id'    => $store->getId(),
                'code'        => $store->getCode(),
                'website_id'  => $store->getWebsiteId(),
                'group_id'    => $store->getGroupId(),
                'name'        => $store->getName(),
                'sort_order'  => $store->getSortOrder(),
                'is_active'   => $store->getIsActive()
            );
        }

        return $result;
    }
```

Method: info

```
$client->call($sessionId, 'core_store.info', $storeId);
```

Implemented In

```
Mage_Core_Model_Store_Api::info
/Users/alanstorm/Sites2011/magento1point6point1.dev/app/code/core/Mage/Core/Model/Store/Api.php
```

Doc Comment

```
/**
 * Retrieve store data
 *
 * @param string|int $storeId
 * @return array
 */
```

Method Implementation Definition

```
    public function info($storeId)
    {
        // Retrieve store info
```

```
        try {
            $store = Mage::app()->getStore($storeId);
        } catch (Mage_Core_Model_Store_Exception $e) {
            $this->_fault('store_not_exists');
        }

        if (!$store->getId()) {
            $this->_fault('store_not_exists');
        }

        // Basic store data
        $result = array();
        $result['store_id'] = $store->getId();
        $result['code'] = $store->getCode();
        $result['website_id'] = $store->getWebsiteId();
        $result['group_id'] = $store->getGroupId();
        $result['name'] = $store->getName();
        $result['sort_order'] = $store->getSortOrder();
        $result['is_active'] = $store->getIsActive();

        return $result;
    }
```

Resource: directory_country

Method: list

```
$client->call($sessionId, 'directory_country.list');
```

Implemented In

```
Mage_Directory_Model_Country_Api::items
/Users/alanstorm/Sites2011/magento1point6point1.dev/app/code/core/Mage/Directory/Model/Country/Api.php
```

Doc Comment

```
/**
 * Retrieve countries list
 *
 * @return array
 */
```

Method Implementation Definition

```
    public function items()
    {
        $collection = Mage::getModel('directory/country')->getCollection();

        $result = array();
        foreach ($collection as $country) {
            /* @var $country Mage_Directory_Model_Country */
            $country->getName(); // Loading name in default locale
            $result[] = $country->toArray(array('country_id', 'iso2_code', 'iso3_code', 'name'));
        }

        return $result;
    }
```

Resource: directory_region

Method: list

```
$client->call($sessionId, 'directory_region.list', $country);
```

Implemented In

Mage_Directory_Model_Region_Api::items
/Users/alanstorm/Sites2011/magento1point6point1.dev/app/code/core/Mage/Directory/Model/Region/Api.php

Doc Comment

```
/**
 * Retrieve regions list
 *
 * @param string $country
 * @return array
 */
```

Method Implementation Definition

```
    public function items($country)
    {
        try {
            $country = Mage::getModel('directory/country')->loadByCode($country);
        } catch (Mage_Core_Exception $e) {
            $this->_fault('country_not_exists', $e->getMessage());
        }

        if (!$country->getId()) {
            $this->_fault('country_not_exists');
        }

        $result = array();
        foreach ($country->getRegions() as $region) {
            $region->getName();
            $result[] = $region->toArray(array('region_id', 'code', 'name'));
        }

        return $result;
    }
```

Resource: customer

Method: list

```
$client->call($sessionId, 'customer.list', $filters);
```

Implemented In

Mage_Customer_Model_Customer_Api::items
/Users/alanstorm/Sites2011/magento1point6point1.dev/app/code/core/Mage/Customer/Model/Customer/Api.php

Doc Comment

```
/**
 * Retrieve cutomers data
 *
 * @param  array $filters
 * @return array
 */
```

Method Implementation Definition

```
    public function items($filters)
    {
        $collection = Mage::getModel('customer/customer')->getCollection()
            ->addAttributeToSelect('*');

        if (is_array($filters)) {
            try {
                foreach ($filters as $field => $value) {
                    if (isset($this->_mapAttributes[$field])) {
```

```
                    $field = $this->_mapAttributes[$field];
                }

                $collection->addFieldToFilter($field, $value);
            }
        } catch (Mage_Core_Exception $e) {
            $this->_fault('filters_invalid', $e->getMessage());
        }
    }

    $result = array();
    foreach ($collection as $customer) {
        $data = $customer->toArray();
        $row  = array();

        foreach ($this->_mapAttributes as $attributeAlias => $attributeCode) {
            $row[$attributeAlias] = (isset($data[$attributeCode]) ? $data[$attributeCode] : null);
        }

        foreach ($this->getAllowedAttributes($customer) as $attributeCode => $attribute) {
            if (isset($data[$attributeCode])) {
                $row[$attributeCode] = $data[$attributeCode];
            }
        }

        $result[] = $row;
    }

    return $result;
}
```

Method: create

```
$client->call($sessionId, 'customer.create', $customerData);
```

Implemented In

```
Mage_Customer_Model_Customer_Api::create
/Users/alanstorm/Sites2011/magento1point6point1.dev/app/code/core/Mage/Customer/Model/Customer/Api.php
```

Doc Comment

```
/**
 * Create new customer
 *
 * @param array $customerData
 * @return int
 */
```

Method Implementation Definition

```
    public function create($customerData)
    {
        $customerData = $this->_prepareData($customerData);
        try {
            $customer = Mage::getModel('customer/customer')
                ->setData($customerData)
                ->save();
        } catch (Mage_Core_Exception $e) {
            $this->_fault('data_invalid', $e->getMessage());
        }
        return $customer->getId();
    }
```

Method: info

```
$client->call($sessionId, 'customer.info', $customerId, $attributes);
```

Implemented In

```
Mage_Customer_Model_Customer_Api::info
/Users/alanstorm/Sites2011/magento1point6point1.dev/app/code/core/Mage/Customer/Model/Customer/Api.php
```

Doc Comment

```
/**
 * Retrieve customer data
 *
 * @param int $customerId
 * @param array $attributes
 * @return array
 */
```

Method Implementation Definition

```
    public function info($customerId, $attributes = null)
    {
        $customer = Mage::getModel('customer/customer')->load($customerId);

        if (!$customer->getId()) {
            $this->_fault('not_exists');
        }

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

        $result = array();

        foreach ($this->_mapAttributes as $attributeAlias=>$attributeCode) {
            $result[$attributeAlias] = $customer->getData($attributeCode);
        }

        foreach ($this->getAllowedAttributes($customer, $attributes) as $attributeCode=>$attribute) {
            $result[$attributeCode] = $customer->getData($attributeCode);
        }

        return $result;
    }
```

Method: update

```
$client->call($sessionId, 'customer.update', $customerId, $customerData);
```

Implemented In

```
Mage_Customer_Model_Customer_Api::update
/Users/alanstorm/Sites2011/magento1point6point1.dev/app/code/core/Mage/Customer/Model/Customer/Api.php
```

Doc Comment

```
/**
 * Update customer data
 *
 * @param int $customerId
 * @param array $customerData
 * @return boolean
 */
```

Method Implementation Definition

```
    public function update($customerId, $customerData)
    {
        $customerData = $this->_prepareData($customerData);
```

```
        $customer = Mage::getModel('customer/customer')->load($customerId);

        if (!$customer->getId()) {
            $this->_fault('not_exists');
        }

        foreach ($this->getAllowedAttributes($customer) as $attributeCode=>$attribute) {
            if (isset($customerData[$attributeCode])) {
                $customer->setData($attributeCode, $customerData[$attributeCode]);
            }
        }

        $customer->save();
        return true;
    }
```

Method: delete

```
$client->call($sessionId, 'customer.delete', $customerId);
```

Implemented In

```
Mage_Customer_Model_Customer_Api::delete
/Users/alanstorm/Sites2011/magento1point6point1.dev/app/code/core/Mage/Customer/Model/Customer/Api.php
```

Doc Comment

```
/**
 * Delete customer
 *
 * @param int $customerId
 * @return boolean
 */
```

Method Implementation Definition

```
    public function delete($customerId)
    {
        $customer = Mage::getModel('customer/customer')->load($customerId);

        if (!$customer->getId()) {
            $this->_fault('not_exists');
        }

        try {
            $customer->delete();
        } catch (Mage_Core_Exception $e) {
            $this->_fault('not_deleted', $e->getMessage());
        }

        return true;
    }
```

Resource: customer_group

Method: list

```
$client->call($sessionId, 'customer_group.list');
```

Implemented In

```
Mage_Customer_Model_Group_Api::items
/Users/alanstorm/Sites2011/magento1point6point1.dev/app/code/core/Mage/Customer/Model/Group/Api.php
```

Doc Comment

```
/**
 * Retrieve groups
 *
 * @return array
 */
```

Method Implementation Definition

```
    public function items()
    {
        $collection = Mage::getModel('customer/group')->getCollection();

        $result = array();
        foreach ($collection as $group) {
            /* @var $group Mage_Customer_Model_Group */
            $result[] = $group->toArray(array('customer_group_id', 'customer_group_code'));
        }

        return $result;
    }
```

Resource: customer_address

Method: list

```
$client->call($sessionId, 'customer_address.list', $customerId);
```

Implemented In

```
Mage_Customer_Model_Address_Api::items
/Users/alanstorm/Sites2011/magento1point6point1.dev/app/code/core/Mage/Customer/Model/Address/Api.php
```

Doc Comment

```
/**
 * Retrive customer addresses list
 *
 * @param int $customerId
 * @return array
 */
```

Method Implementation Definition

```
    public function items($customerId)
    {
        $customer = Mage::getModel('customer/customer')
            ->load($customerId);
        /* @var $customer Mage_Customer_Model_Customer */

        if (!$customer->getId()) {
            $this->_fault('customer_not_exists');
        }

        $result = array();
        foreach ($customer->getAddresses() as $address) {
            $data = $address->toArray();
            $row  = array();

            foreach ($this->_mapAttributes as $attributeAlias => $attributeCode) {
                $row[$attributeAlias] = isset($data[$attributeCode]) ? $data[$attributeCode] : null;
            }

            foreach ($this->getAllowedAttributes($address) as $attributeCode => $attribute) {
                if (isset($data[$attributeCode])) {
                    $row[$attributeCode] = $data[$attributeCode];
                }
```

```
            }

            $row['is_default_billing'] = $customer->getDefaultBilling() == $address->getId();
            $row['is_default_shipping'] = $customer->getDefaultShipping() == $address->getId();

            $result[] = $row;

        }

        return $result;
    }
```

Method: create

```
$client->call($sessionId, 'customer_address.create', $customerId, $addressData);
```

Implemented In

```
Mage_Customer_Model_Address_Api::create
/Users/alanstorm/Sites2011/magento1point6point1.dev/app/code/core/Mage/Customer/Model/Address/Api.php
```

Doc Comment

```
/**
 * Create new address for customer
 *
 * @param int $customerId
 * @param array $addressData
 * @return int
 */
```

Method Implementation Definition

```
    public function create($customerId, $addressData)
    {
        $customer = Mage::getModel('customer/customer')
            ->load($customerId);
        /* @var $customer Mage_Customer_Model_Customer */

        if (!$customer->getId()) {
            $this->_fault('customer_not_exists');
        }

        $address = Mage::getModel('customer/address');

        foreach ($this->getAllowedAttributes($address) as $attributeCode=>$attribute) {
            if (isset($addressData[$attributeCode])) {
                $address->setData($attributeCode, $addressData[$attributeCode]);
            }
        }

        if (isset($addressData['is_default_billing'])) {
            $address->setIsDefaultBilling($addressData['is_default_billing']);
        }

        if (isset($addressData['is_default_shipping'])) {
            $address->setIsDefaultShipping($addressData['is_default_shipping']);
        }

        $address->setCustomerId($customer->getId());

        $valid = $address->validate();

        if (is_array($valid)) {
            $this->_fault('data_invalid', implode("\n", $valid));
```

```
        }

        try {
            $address->save();
        } catch (Mage_Core_Exception $e) {
            $this->_fault('data_invalid', $e->getMessage());
        }

        return $address->getId();
    }
```

Method: info

```
$client->call($sessionId, 'customer_address.info', $addressId);
```

Implemented In

```
Mage_Customer_Model_Address_Api::info
/Users/alanstorm/Sites2011/magento1point6point1.dev/app/code/core/Mage/Customer/Model/Address/Api.php
```

Doc Comment

```
/**
 * Retrieve address data
 *
 * @param int $addressId
 * @return array
 */
```

Method Implementation Definition

```
    public function info($addressId)
    {
        $address = Mage::getModel('customer/address')
            ->load($addressId);

        if (!$address->getId()) {
            $this->_fault('not_exists');
        }

        $result = array();

        foreach ($this->_mapAttributes as $attributeAlias => $attributeCode) {
            $result[$attributeAlias] = $address->getData($attributeCode);
        }

        foreach ($this->getAllowedAttributes($address) as $attributeCode => $attribute) {
            $result[$attributeCode] = $address->getData($attributeCode);
        }


        if ($customer = $address->getCustomer()) {
            $result['is_default_billing']  = $customer->getDefaultBilling() == $address->getId();
            $result['is_default_shipping'] = $customer->getDefaultShipping() == $address->getId();
        }

        return $result;
    }
```

Method: update

```
$client->call($sessionId, 'customer_address.update', $addressId, $addressData);
```

Implemented In

```
Mage_Customer_Model_Address_Api::update
```

/Users/alanstorm/Sites2011/magento1point6point1.dev/app/code/core/Mage/Customer/Model/Address/Api.php

Doc Comment

```
/**
 * Update address data
 *
 * @param int $addressId
 * @param array $addressData
 * @return boolean
 */
```

Method Implementation Definition

```
    public function update($addressId, $addressData)
    {
        $address = Mage::getModel('customer/address')
            ->load($addressId);

        if (!$address->getId()) {
            $this->_fault('not_exists');
        }

        foreach ($this->getAllowedAttributes($address) as $attributeCode=>$attribute) {
            if (isset($addressData[$attributeCode])) {
                $address->setData($attributeCode, $addressData[$attributeCode]);
            }
        }

        if (isset($addressData['is_default_billing'])) {
            $address->setIsDefaultBilling($addressData['is_default_billing']);
        }

        if (isset($addressData['is_default_shipping'])) {
            $address->setIsDefaultShipping($addressData['is_default_shipping']);
        }

        $valid = $address->validate();
        if (is_array($valid)) {
            $this->_fault('data_invalid', implode("\n", $valid));
        }

        try {
            $address->save();
        } catch (Mage_Core_Exception $e) {
            $this->_fault('data_invalid', $e->getMessage());
        }

        return true;
    }
```

Method: delete

```
$client->call($sessionId, 'customer_address.delete', $addressId);
```

Implemented In

Mage_Customer_Model_Address_Api::delete
/Users/alanstorm/Sites2011/magento1point6point1.dev/app/code/core/Mage/Customer/Model/Address/Api.php

Doc Comment

```
/**
 * Delete address
 *
 * @param int $addressId
```

```
 * @return boolean
 */
```

Method Implementation Definition

```
    public function delete($addressId)
    {
        $address = Mage::getModel('customer/address')
            ->load($addressId);

        if (!$address->getId()) {
            $this->_fault('not_exists');
        }

        try {
            $address->delete();
        } catch (Mage_Core_Exception $e) {
            $this->_fault('not_deleted', $e->getMessage());
        }

        return true;
    }
```

Resource: catalog_category

Method: currentStore

```
$client->call($sessionId, 'catalog_category.currentStore', $store);
```

Implemented In

```
Mage_Catalog_Model_Category_Api::currentStore
/Users/alanstorm/Sites2011/magento1point6point1.dev/app/code/core/Mage/Catalog/Model/Api/Resource.php
```

Doc Comment

```
/**
 * Set current store for catalog.
 *
 * @param string|int $store
 * @return int
 */
```

Method Implementation Definition

```
    public function currentStore($store=null)
    {
        if (!is_null($store)) {
            try {
                $storeId = Mage::app()->getStore($store)->getId();
            } catch (Mage_Core_Model_Store_Exception $e) {
                $this->_fault('store_not_exists');
            }

            $this->_getSession()->setData($this->_storeIdSessionField, $storeId);
        }

        return $this->_getStoreId();
    }
```

Method: tree

```
$client->call($sessionId, 'catalog_category.tree', $parentId, $store);
```

Implemented In

```
Mage_Catalog_Model_Category_Api::tree
```

/Users/alanstorm/Sites2011/magento1point6point1.dev/app/code/core/Mage/Catalog/Model/Category/Api.php

Doc Comment

```
/**
 * Retrieve category tree
 *
 * @param int $parent
 * @param string|int $store
 * @return array
 */
```

Method Implementation Definition

```
    public function tree($parentId = null, $store = null)
    {
        if (is_null($parentId) && !is_null($store)) {
            $parentId = Mage::app()->getStore($this->_getStoreId($store))->getRootCategoryId();
        } elseif (is_null($parentId)) {
            $parentId = 1;
        }

        /* @var $tree Mage_Catalog_Model_Resource_Eav_Mysql4_Category_Tree */
        $tree = Mage::getResourceSingleton('catalog/category_tree')
            ->load();

        $root = $tree->getNodeById($parentId);

        if($root && $root->getId() == 1) {
            $root->setName(Mage::helper('catalog')->__('Root'));
        }

        $collection = Mage::getModel('catalog/category')->getCollection()
            ->setStoreId($this->_getStoreId($store))
            ->addAttributeToSelect('name')
            ->addAttributeToSelect('is_active');

        $tree->addCollectionData($collection, true);

        return $this->_nodeToArray($root);
    }
```

Method: level

```
$client->call($sessionId, 'catalog_category.level', $website, $store, $categoryId);
```

Implemented In

Mage_Catalog_Model_Category_Api::level
/Users/alanstorm/Sites2011/magento1point6point1.dev/app/code/core/Mage/Catalog/Model/Category/Api.php

Doc Comment

```
/**
 * Retrieve level of categories for category/store view/website
 *
 * @param string|int|null $website
 * @param string|int|null $store
 * @param int|null $categoryId
 * @return array
 */
```

Method Implementation Definition

```
    public function level($website = null, $store = null, $categoryId = null)
    {
        $ids = array();
        $storeId = Mage_Catalog_Model_Category::DEFAULT_STORE_ID;
```

```
// load root categories of website
if (null !== $website) {
    try {
        $website = Mage::app()->getWebsite($website);
        if (null === $store) {
            if (null === $categoryId) {
                foreach ($website->getStores() as $store) {
                    /* @var $store Mage_Core_Model_Store */
                    $ids[] = $store->getRootCategoryId();
                }
            } else {
                $ids = $categoryId;
            }
        } elseif (in_array($store, $website->getStoreIds())) {
            $storeId = Mage::app()->getStore($store)->getId();
            $ids = (null === $categoryId)? $store->getRootCategoryId() : $categoryId;
        } else {
            $this->_fault('store_not_exists');
        }
    } catch (Mage_Core_Exception $e) {
        $this->_fault('website_not_exists', $e->getMessage());
    }
}
elseif (null !== $store) {
    // load children of root category of store
    if (null === $categoryId) {
        try {
            $store = Mage::app()->getStore($store);
            $storeId = $store->getId();
            $ids = $store->getRootCategoryId();
        } catch (Mage_Core_Model_Store_Exception $e) {
            $this->_fault('store_not_exists');
        }
    }
    // load children of specified category id
    else {
        $storeId = $this->_getStoreId($store);
        $ids = (int)$categoryId;
    }
}
// load all root categories
else {
    $ids = (null === $categoryId)? Mage_Catalog_Model_Category::TREE_ROOT_ID : $categoryId;
}

$collection = Mage::getModel('catalog/category')->getCollection()
    ->setStoreId($storeId)
    ->addAttributeToSelect('name')
    ->addAttributeToSelect('is_active');

if (is_array($ids)) {
    $collection->addFieldToFilter('entity_id', array('in' => $ids));
} else {
    $collection->addFieldToFilter('parent_id', $ids);
}

// Only basic category data
$result = array();
foreach ($collection as $category) {
    /* @var $category Mage_Catalog_Model_Category */
```

```
            $result[] = array(
                'category_id' => $category->getId(),
                'parent_id'   => $category->getParentId(),
                'name'        => $category->getName(),
                'is_active'   => $category->getIsActive(),
                'position'    => $category->getPosition(),
                'level'       => $category->getLevel()
            );
        }

        return $result;
    }
```

Method: info

```
$client->call($sessionId, 'catalog_category.info', $categoryId, $store, $attributes);
```

Implemented In

```
Mage_Catalog_Model_Category_Api::info
/Users/alanstorm/Sites2011/magento1point6point1.dev/app/code/core/Mage/Catalog/Model/Category/Api.php
```

Doc Comment

```
/**
 * Retrieve category data
 *
 * @param int $categoryId
 * @param string|int $store
 * @param array $attributes
 * @return array
 */
```

Method Implementation Definition

```
    public function info($categoryId, $store = null, $attributes = null)
    {
        $category = $this->_initCategory($categoryId, $store);

        // Basic category data
        $result = array();
        $result['category_id'] = $category->getId();

        $result['is_active']   = $category->getIsActive();
        $result['position']    = $category->getPosition();
        $result['level']       = $category->getLevel();

        foreach ($category->getAttributes() as $attribute) {
            if ($this->_isAllowedAttribute($attribute, $attributes)) {
                $result[$attribute->getAttributeCode()] = $category->getData($attribute->getAttributeCode());
            }
        }
        $result['parent_id']   = $category->getParentId();
        $result['children']           = $category->getChildren();
        $result['all_children']       = $category->getAllChildren();

        return $result;
    }
```

Method: create

```
$client->call($sessionId, 'catalog_category.create', $parentId, $categoryData, $store);
```

Implemented In

```
Mage_Catalog_Model_Category_Api::create
```

/Users/alanstorm/Sites2011/magento1point6point1.dev/app/code/core/Mage/Catalog/Model/Category/Api.php

Doc Comment

```
/**
 * Create new category
 *
 * @param int $parentId
 * @param array $categoryData
 * @param int|null|string $store
 * @return int
 */
```

Method Implementation Definition

```
    public function create($parentId, $categoryData, $store = null)
    {
        $parent_category = $this->_initCategory($parentId, $store);

        /** @var $category Mage_Catalog_Model_Category */
        $category = Mage::getModel('catalog/category')
            ->setStoreId($this->_getStoreId($store));

        $category->addData(array('path'=>implode('/', $parent_category->getPathIds())));
        $category->setAttributeSetId($category->getDefaultAttributeSetId());

        $useConfig = array();
        foreach ($category->getAttributes() as $attribute) {
            if ($this->_isAllowedAttribute($attribute)
                && isset($categoryData[$attribute->getAttributeCode()])
            ) {
                // check whether value is 'use_config'
                $attrCode = $attribute->getAttributeCode();
                $categoryDataValue = $categoryData[$attrCode];
                if ('use_config' === $categoryDataValue ||
                    (is_array($categoryDataValue) &&
                    count($categoryDataValue) == 1 &&
                    'use_config' === $categoryDataValue[0])
                ) {
                    $useConfig[] = $attrCode;
                    $category->setData($attrCode, null);
                } else {
                    $category->setData($attrCode, $categoryDataValue);
                }
            }
        }

        $category->setParentId($parent_category->getId());

        /**
         * Proceed with $useConfig set into category model for processing through validation
         */
        if (count($useConfig) > 0) {
            $category->setData("use_post_data_config", $useConfig);
        }

        try {
            $validate = $category->validate();
            if ($validate !== true) {
                foreach ($validate as $code => $error) {
                    if ($error === true) {
                        Mage::throwException(Mage::helper('catalog')->__('Attribute "%s" is required.', $code));
                    } else {
                        Mage::throwException($error);
```

```
                }
            }
        }

        $category->save();
    }
    catch (Mage_Core_Exception $e) {
        $this->_fault('data_invalid', $e->getMessage());
    }
    catch (Exception $e) {
        $this->_fault('data_invalid', $e->getMessage());
    }

    return $category->getId();
}
```

Method: update

```
$client->call($sessionId, 'catalog_category.update', $categoryId, $categoryData, $store);
```

Implemented In

```
Mage_Catalog_Model_Category_Api::update
/Users/alanstorm/Sites2011/magento1point6point1.dev/app/code/core/Mage/Catalog/Model/Category/Api.php
```

Doc Comment

```
/**
 * Update category data
 *
 * @param int $categoryId
 * @param array $categoryData
 * @param string|int $store
 * @return boolean
 */
```

Method Implementation Definition

```
public function update($categoryId, $categoryData, $store = null)
{
    $category = $this->_initCategory($categoryId, $store);

    foreach ($category->getAttributes() as $attribute) {
        if ($this->_isAllowedAttribute($attribute)
            && isset($categoryData[$attribute->getAttributeCode()])) {
            $category->setData(
                $attribute->getAttributeCode(),
                $categoryData[$attribute->getAttributeCode()]
            );
        }
    }

    try {
        $validate = $category->validate();
        if ($validate !== true) {
            foreach ($validate as $code => $error) {
                if ($error === true) {
                    Mage::throwException(Mage::helper('catalog')->__('Attribute "%s" is required.', $code));
                }
                else {
                    Mage::throwException($error);
                }
            }
        }
```

```
            $category->save();
        }
        catch (Mage_Core_Exception $e) {
            $this->_fault('data_invalid', $e->getMessage());
        }

        return true;
    }
```

Method: move

```
$client->call($sessionId, 'catalog_category.move', $categoryId, $parentId, $afterId);
```

Implemented In

```
Mage_Catalog_Model_Category_Api::move
/Users/alanstorm/Sites2011/magento1point6point1.dev/app/code/core/Mage/Catalog/Model/Category/Api.php
```

Doc Comment

```
/**
 * Move category in tree
 *
 * @param int $categoryId
 * @param int $parentId
 * @param int $afterId
 * @return boolean
 */
```

Method Implementation Definition

```
    public function move($categoryId, $parentId, $afterId = null)
    {
        $category = $this->_initCategory($categoryId);
        $parent_category = $this->_initCategory($parentId);

        // if $afterId is null - move category to the down
        if ($afterId === null && $parent_category->hasChildren()) {
            $parentChildren = $parent_category->getChildren();
            $afterId = array_pop(explode(',', $parentChildren));
        }

        if( strpos($parent_category->getPath(), $category->getPath()) === 0) {
            $this->_fault('not_moved', "Operation do not allow to move a parent category to any of children
category");
        }

        try {
            $category->move($parentId, $afterId);
        } catch (Mage_Core_Exception $e) {
            $this->_fault('not_moved', $e->getMessage());
        }

        return true;
    }
```

Method: delete

```
$client->call($sessionId, 'catalog_category.delete', $categoryId);
```

Implemented In

```
Mage_Catalog_Model_Category_Api::delete
/Users/alanstorm/Sites2011/magento1point6point1.dev/app/code/core/Mage/Catalog/Model/Category/Api.php
```

Doc Comment

```
/**
 * Delete category
 *
 * @param int $categoryId
 * @return boolean
 */
```

Method Implementation Definition

```
    public function delete($categoryId)
    {
        $category = $this->_initCategory($categoryId);

        try {
            $category->delete();
        } catch (Mage_Core_Exception $e) {
            $this->_fault('not_deleted', $e->getMessage());
        }

        return true;
    }
```

Method: assignedProducts

```
$client->call($sessionId, 'catalog_category.assignedProducts', $categoryId, $store);
```

Implemented In

Mage_Catalog_Model_Category_Api::assignedProducts
/Users/alanstorm/Sites2011/magento1point6point1.dev/app/code/core/Mage/Catalog/Model/Category/Api.php

Doc Comment

```
/**
 * Retrieve list of assigned products to category
 *
 * @param int $categoryId
 * @param string|int $store
 * @return array
 */
```

Method Implementation Definition

```
    public function assignedProducts($categoryId, $store = null)
    {
        $category = $this->_initCategory($categoryId);

        $storeId = $this->_getStoreId($store);
        $collection = $category->setStoreId($storeId)->getProductCollection();
        ($storeId == 0)? $collection->addOrder('position', 'asc') : $collection->setOrder('position', 'asc');;

        $result = array();

        foreach ($collection as $product) {
            $result[] = array(
                'product_id' => $product->getId(),
                'type'       => $product->getTypeId(),
                'set'        => $product->getAttributeSetId(),
                'sku'        => $product->getSku(),
                'position'   => $product->getPosition()
            );
        }

        return $result;
    }
```

Method: assignProduct

```
$client->call($sessionId, 'catalog_category.assignProduct', $categoryId, $productId, $position, $identifierType);
```

Implemented In

```
Mage_Catalog_Model_Category_Api::assignProduct
/Users/alanstorm/Sites2011/magento1point6point1.dev/app/code/core/Mage/Catalog/Model/Category/Api.php
```

Doc Comment

```
/**
 * Assign product to category
 *
 * @param int $categoryId
 * @param int $productId
 * @param int $position
 * @return boolean
 */
```

Method Implementation Definition

```
    public function assignProduct($categoryId, $productId, $position = null, $identifierType = null)
    {
        $category = $this->_initCategory($categoryId);
        $positions = $category->getProductsPosition();
        $productId = $this->_getProductId($productId);
        $positions[$productId] = $position;
        $category->setPostedProducts($positions);

        try {
            $category->save();
        } catch (Mage_Core_Exception $e) {
            $this->_fault('data_invalid', $e->getMessage());
        }

        return true;
    }
```

Method: updateProduct

```
$client->call($sessionId, 'catalog_category.updateProduct', $categoryId, $productId, $position, $identifierType);
```

Implemented In

```
Mage_Catalog_Model_Category_Api::updateProduct
/Users/alanstorm/Sites2011/magento1point6point1.dev/app/code/core/Mage/Catalog/Model/Category/Api.php
```

Doc Comment

```
/**
 * Update product assignment
 *
 * @param int $categoryId
 * @param int $productId
 * @param int $position
 * @return boolean
 */
```

Method Implementation Definition

```
    public function updateProduct($categoryId, $productId, $position = null, $identifierType = null)
    {
        $category = $this->_initCategory($categoryId);
        $positions = $category->getProductsPosition();
        $productId = $this->_getProductId($productId, $identifierType);
        if (!isset($positions[$productId])) {
```

```
            $this->_fault('product_not_assigned');
        }
        $positions[$productId] = $position;
        $category->setPostedProducts($positions);

        try {
            $category->save();
        } catch (Mage_Core_Exception $e) {
            $this->_fault('data_invalid', $e->getMessage());
        }

        return true;
    }
```

Method: removeProduct

```
$client->call($sessionId, 'catalog_category.removeProduct', $categoryId, $productId, $identifierType);
```

Implemented In

```
Mage_Catalog_Model_Category_Api::removeProduct
/Users/alanstorm/Sites2011/magento1point6point1.dev/app/code/core/Mage/Catalog/Model/Category/Api.php
```

Doc Comment

```
/**
 * Remove product assignment from category
 *
 * @param int $categoryId
 * @param int $productId
 * @return boolean
 */
```

Method Implementation Definition

```
    public function removeProduct($categoryId, $productId, $identifierType = null)
    {
        $category = $this->_initCategory($categoryId);
        $positions = $category->getProductsPosition();
        $productId = $this->_getProductId($productId, $identifierType);
        if (!isset($positions[$productId])) {
            $this->_fault('product_not_assigned');
        }

        unset($positions[$productId]);
        $category->setPostedProducts($positions);

        try {
            $category->save();
        } catch (Mage_Core_Exception $e) {
            $this->_fault('data_invalid', $e->getMessage());
        }

        return true;
    }
```

Resource: catalog_category_attribute

Method: currentStore

```
$client->call($sessionId, 'catalog_category_attribute.currentStore', $store);
```

Implemented In

```
Mage_Catalog_Model_Category_Attribute_Api::currentStore
```

/Users/alanstorm/Sites2011/magento1point6point1.dev/app/code/core/Mage/Catalog/Model/Api/Resource.php

Doc Comment

```
/**
 * Set current store for catalog.
 *
 * @param string|int $store
 * @return int
 */
```

Method Implementation Definition

```
    public function currentStore($store=null)
    {
        if (!is_null($store)) {
            try {
                $storeId = Mage::app()->getStore($store)->getId();
            } catch (Mage_Core_Model_Store_Exception $e) {
                $this->_fault('store_not_exists');
            }

            $this->_getSession()->setData($this->_storeIdSessionField, $storeId);
        }

        return $this->_getStoreId();
    }
```

Method: list

```
$client->call($sessionId, 'catalog_category_attribute.list');
```

Implemented In

Mage_Catalog_Model_Category_Attribute_Api::items
/Users/alanstorm/Sites2011/magento1point6point1.dev/app/code/core/Mage/Catalog/Model/Category/Attribute/Api.php

Doc Comment

```
/**
 * Retrieve category attributes
 *
 * @return array
 */
```

Method Implementation Definition

```
    public function items()
    {
        $attributes = Mage::getModel('catalog/category')->getAttributes();
        $result = array();

        foreach ($attributes as $attribute) {
            /* @var $attribute Mage_Catalog_Model_Resource_Eav_Attribute */
            if ($this->_isAllowedAttribute($attribute)) {
                if (!$attribute->getId() || $attribute->isScopeGlobal()) {
                    $scope = 'global';
                } elseif ($attribute->isScopeWebsite()) {
                    $scope = 'website';
                } else {
                    $scope = 'store';
                }

                $result[] = array(
                    'attribute_id' => $attribute->getId(),
                    'code'         => $attribute->getAttributeCode(),
                    'type'         => $attribute->getFrontendInput(),
                    'required'     => $attribute->getIsRequired(),
```

```
                'scope'        => $scope
            );
        }
    }

    return $result;
}
```

Method: options

```
$client->call($sessionId, 'catalog_category_attribute.options', $attributeId, $store);
```

Implemented In

```
Mage_Catalog_Model_Category_Attribute_Api::options
/Users/alanstorm/Sites2011/magento1point6point1.dev/app/code/core/Mage/Catalog/Model/Category/Attribute/Api.php
```

Doc Comment

```
/**
 * Retrieve category attribute options
 *
 * @param int|string $attributeId
 * @param string|int $store
 * @return array
 */
```

Method Implementation Definition

```
    public function options($attributeId, $store = null)
    {
        $attribute = Mage::getModel('catalog/category')
            ->setStoreId($this->_getStoreId($store))
            ->getResource()
            ->getAttribute($attributeId);

        if (!$attribute) {
            $this->_fault('not_exists');
        }

        $result = array();
        if ($attribute->usesSource()) {
            foreach ($attribute->getSource()->getAllOptions(false) as $optionId=>$optionValue) {
                if (is_array($optionValue)) {
                    $result[] = $optionValue;
                } else {
                    $result[] = array(
                        'value' => $optionId,
                        'label' => $optionValue
                    );
                }
            }
        }

        return $result;
    }
```

Resource: catalog_product

Method: currentStore

```
$client->call($sessionId, 'catalog_product.currentStore', $store);
```

Implemented In

Mage_Catalog_Model_Product_Api::currentStore
/Users/alanstorm/Sites2011/magento1point6point1.dev/app/code/core/Mage/Catalog/Model/Api/Resource.php

Doc Comment

```
/**
 * Set current store for catalog.
 *
 * @param string|int $store
 * @return int
 */
```

Method Implementation Definition

```
    public function currentStore($store=null)
    {
        if (!is_null($store)) {
            try {
                $storeId = Mage::app()->getStore($store)->getId();
            } catch (Mage_Core_Model_Store_Exception $e) {
                $this->_fault('store_not_exists');
            }

            $this->_getSession()->setData($this->_storeIdSessionField, $storeId);
        }

        return $this->_getStoreId();
    }
```

Method: list

```
$client->call($sessionId, 'catalog_product.list', $filters, $store);
```

Implemented In

Mage_Catalog_Model_Product_Api::items
/Users/alanstorm/Sites2011/magento1point6point1.dev/app/code/core/Mage/Catalog/Model/Product/Api.php

Doc Comment

```
/**
 * Retrieve list of products with basic info (id, sku, type, set, name)
 *
 * @param array $filters
 * @param string|int $store
 * @return array
 */
```

Method Implementation Definition

```
    public function items($filters = null, $store = null)
    {
        $collection = Mage::getModel('catalog/product')->getCollection()
            ->addStoreFilter($this->_getStoreId($store))
            ->addAttributeToSelect('name');

        if (is_array($filters)) {
            try {
                foreach ($filters as $field => $value) {
                    if (isset($this->_filtersMap[$field])) {
                        $field = $this->_filtersMap[$field];
                    }

                    $collection->addFieldToFilter($field, $value);
                }
            } catch (Mage_Core_Exception $e) {
                $this->_fault('filters_invalid', $e->getMessage());
            }
```

```
        }

        $result = array();

        foreach ($collection as $product) {
//            $result[] = $product->getData();
            $result[] = array( // Basic product data
                'product_id' => $product->getId(),
                'sku'        => $product->getSku(),
                'name'       => $product->getName(),
                'set'        => $product->getAttributeSetId(),
                'type'       => $product->getTypeId(),
                'category_ids'       => $product->getCategoryIds()
            );
        }

        return $result;
    }
```

Method: info

```
$client->call($sessionId, 'catalog_product.info', $productId, $store, $attributes, $identifierType);
```

Implemented In

```
Mage_Catalog_Model_Product_Api::info
/Users/alanstorm/Sites2011/magento1point6point1.dev/app/code/core/Mage/Catalog/Model/Product/Api.php
```

Doc Comment

```
/**
 * Retrieve product info
 *
 * @param int|string $productId
 * @param string|int $store
 * @param array $attributes
 * @return array
 */
```

Method Implementation Definition

```
    public function info($productId, $store = null, $attributes = null, $identifierType = null)
    {
        $product = $this->_getProduct($productId, $store, $identifierType);


        $result = array( // Basic product data
            'product_id' => $product->getId(),
            'sku'        => $product->getSku(),
            'set'        => $product->getAttributeSetId(),
            'type'       => $product->getTypeId(),
            'categories' => $product->getCategoryIds(),
            'websites'   => $product->getWebsiteIds()
        );

        foreach ($product->getTypeInstance(true)->getEditableAttributes($product) as $attribute) {
            if ($this->_isAllowedAttribute($attribute, $attributes)) {
                $result[$attribute->getAttributeCode()] = $product->getData(
                                                                $attribute->getAttributeCode());
            }
        }

        return $result;
    }
```

Method: create

```
$client->call($sessionId, 'catalog_product.create', $type, $set, $sku, $productData, $store);
```

Implemented In

```
Mage_Catalog_Model_Product_Api::create
/Users/alanstorm/Sites2011/magento1point6point1.dev/app/code/core/Mage/Catalog/Model/Product/Api.php
```

Doc Comment

```
/**
 * Create new product.
 *
 * @param string $type
 * @param int $set
 * @param string $sku
 * @param array $productData
 * @param string $store
 * @return int
 */
```

Method Implementation Definition

```
    public function create($type, $set, $sku, $productData, $store = null)
    {
        if (!$type || !$set || !$sku) {
            $this->_fault('data_invalid');
        }

        $this->_checkProductTypeExists($type);
        $this->_checkProductAttributeSet($set);

        /** @var $product Mage_Catalog_Model_Product */
        $product = Mage::getModel('catalog/product');
        $product->setStoreId($this->_getStoreId($store))
            ->setAttributeSetId($set)
            ->setTypeId($type)
            ->setSku($sku);

        $this->_prepareDataForSave($product, $productData);

        try {
            /**
             * @todo implement full validation process with errors returning which are ignoring now
             * @todo see Mage_Catalog_Model_Product::validate()
             */
            if (is_array($errors = $product->validate())) {
                $strErrors = array();
                foreach($errors as $code => $error) {
                    if ($error === true) {
                        $error = Mage::helper('catalog')->__('Attribute "%s" is invalid.', $code);
                    }
                    $strErrors[] = $error;
                }
                $this->_fault('data_invalid', implode("\n", $strErrors));
            }

            $product->save();
        } catch (Mage_Core_Exception $e) {
            $this->_fault('data_invalid', $e->getMessage());
        }

        return $product->getId();
    }
```

Method: update

```
$client->call($sessionId, 'catalog_product.update', $productId, $productData, $store, $identifierType);
```

Implemented In

```
Mage_Catalog_Model_Product_Api::update
/Users/alanstorm/Sites2011/magento1point6point1.dev/app/code/core/Mage/Catalog/Model/Product/Api.php
```

Doc Comment

```
/**
 * Update product data
 *
 * @param int|string $productId
 * @param array $productData
 * @param string|int $store
 * @return boolean
 */
```

Method Implementation Definition

```
    public function update($productId, $productData, $store = null, $identifierType = null)
    {
        $product = $this->_getProduct($productId, $store, $identifierType);

        $this->_prepareDataForSave($product, $productData);

        try {
            /**
             * @todo implement full validation process with errors returning which are ignoring now
             * @todo see Mage_Catalog_Model_Product::validate()
             */
            if (is_array($errors = $product->validate())) {
                $strErrors = array();
                foreach($errors as $code => $error) {
                    if ($error === true) {
                        $error = Mage::helper('catalog')->__('Value for "%s" is invalid.', $code);
                    } else {
                        $error = Mage::helper('catalog')->__('Value for "%s" is invalid: %s', $code, $error);
                    }
                    $strErrors[] = $error;
                }
                $this->_fault('data_invalid', implode("\n", $strErrors));
            }

            $product->save();
        } catch (Mage_Core_Exception $e) {
            $this->_fault('data_invalid', $e->getMessage());
        }

        return true;
    }
```

Method: delete

```
$client->call($sessionId, 'catalog_product.delete', $productId, $identifierType);
```

Implemented In

```
Mage_Catalog_Model_Product_Api::delete
/Users/alanstorm/Sites2011/magento1point6point1.dev/app/code/core/Mage/Catalog/Model/Product/Api.php
```

Doc Comment

```
/**
```

```
 * Delete product
 *
 * @param int|string $productId
 * @return boolean
 */
```

Method Implementation Definition

```
    public function delete($productId, $identifierType = null)
    {
        $product = $this->_getProduct($productId, null, $identifierType);

        try {
            $product->delete();
        } catch (Mage_Core_Exception $e) {
            $this->_fault('not_deleted', $e->getMessage());
        }

        return true;
    }
```

Method: getSpecialPrice

```
$client->call($sessionId, 'catalog_product.getSpecialPrice', $productId, $store);
```

Implemented In

```
Mage_Catalog_Model_Product_Api::getSpecialPrice
/Users/alanstorm/Sites2011/magento1point6point1.dev/app/code/core/Mage/Catalog/Model/Product/Api.php
```

Doc Comment

```
/**
 * Retrieve product special price
 *
 * @param int|string $productId
 * @param string|int $store
 * @return array
 */
```

Method Implementation Definition

```
    public function getSpecialPrice($productId, $store = null)
    {
        return $this->info($productId, $store, array('special_price', 'special_from_date', 'special_to_date'));
    }
```

Method: setSpecialPrice

```
$client->call($sessionId, 'catalog_product.setSpecialPrice', $productId, $specialPrice, $fromDate, $toDate,
$store);
```

Implemented In

```
Mage_Catalog_Model_Product_Api::setSpecialPrice
/Users/alanstorm/Sites2011/magento1point6point1.dev/app/code/core/Mage/Catalog/Model/Product/Api.php
```

Doc Comment

```
/**
 * Update product special price
 *
 * @param int|string $productId
 * @param float $specialPrice
 * @param string $fromDate
 * @param string $toDate
 * @param string|int $store
 * @return boolean
 */
```

Method Implementation Definition

```
    public function setSpecialPrice($productId, $specialPrice = null, $fromDate = null, $toDate = null, $store =
null)
    {
        return $this->update($productId, array(
            'special_price'     => $specialPrice,
            'special_from_date' => $fromDate,
            'special_to_date'   => $toDate
        ), $store);
    }
```

Method: listOfAdditionalAttributes

```
$client->call($sessionId, 'catalog_product.listOfAdditionalAttributes', $productType, $attributeSetId);
```

Implemented In

```
Mage_Catalog_Model_Product_Api::getAdditionalAttributes
/Users/alanstorm/Sites2011/magento1point6point1.dev/app/code/core/Mage/Catalog/Model/Product/Api.php
```

Doc Comment

```
/**
* Get list of additional attributes which are not in default create/update list
*
* @param  $productType
* @param  $attributeSetId
* @return array
*/
```

Method Implementation Definition

```
    public function getAdditionalAttributes($productType, $attributeSetId)
    {
        $this->_checkProductTypeExists($productType);
        $this->_checkProductAttributeSet($attributeSetId);

        /** @var $product Mage_Catalog_Model_Product */
        $productAttributes = Mage::getModel('catalog/product')
            ->setAttributeSetId($attributeSetId)
            ->setTypeId($productType)
            ->getTypeInstance(false)
            ->getEditableAttributes();

        $result = array();
        foreach ($productAttributes as $attribute) {
            /* @var $attribute Mage_Catalog_Model_Resource_Eav_Attribute */
            if ($attribute->isInSet($attributeSetId) && $this->_isAllowedAttribute($attribute)
                && !in_array($attribute->getAttributeCode(), $this->_defaultProductAttributeList)) {

                if ($attribute->isScopeGlobal()) {
                    $scope = 'global';
                } elseif ($attribute->isScopeWebsite()) {
                    $scope = 'website';
                } else {
                    $scope = 'store';
                }

                $result[] = array(
                    'attribute_id' => $attribute->getId(),
                    'code' => $attribute->getAttributeCode(),
                    'type' => $attribute->getFrontendInput(),
                    'required' => $attribute->getIsRequired(),
                    'scope' => $scope
```

```
                );
            }
        }

        return $result;
    }
```

Resource: catalog_product_attribute

Method: currentStore

```
$client->call($sessionId, 'catalog_product_attribute.currentStore', $store);
```

Implemented In

```
Mage_Catalog_Model_Product_Attribute_Api::currentStore
/Users/alanstorm/Sites2011/magento1point6point1.dev/app/code/core/Mage/Catalog/Model/Api/Resource.php
```

Doc Comment

```
/**
 * Set current store for catalog.
 *
 * @param string|int $store
 * @return int
 */
```

Method Implementation Definition

```
    public function currentStore($store=null)
    {
        if (!is_null($store)) {
            try {
                $storeId = Mage::app()->getStore($store)->getId();
            } catch (Mage_Core_Model_Store_Exception $e) {
                $this->_fault('store_not_exists');
            }

            $this->_getSession()->setData($this->_storeIdSessionField, $storeId);
        }

        return $this->_getStoreId();
    }
```

Method: list

```
$client->call($sessionId, 'catalog_product_attribute.list', $setId);
```

Implemented In

```
Mage_Catalog_Model_Product_Attribute_Api::items
/Users/alanstorm/Sites2011/magento1point6point1.dev/app/code/core/Mage/Catalog/Model/Product/Attribute/Api.php
```

Doc Comment

```
/**
 * Retrieve attributes from specified attribute set
 *
 * @param int $setId
 * @return array
 */
```

Method Implementation Definition

```
    public function items($setId)
    {
        $attributes = Mage::getModel('catalog/product')->getResource()
                ->loadAllAttributes()
```

```
                ->getSortedAttributes($setId);
        $result = array();

        foreach ($attributes as $attribute) {
            /* @var $attribute Mage_Catalog_Model_Resource_Eav_Attribute */
            if ((!$attribute->getId() || $attribute->isInSet($setId))
                    && $this->_isAllowedAttribute($attribute)) {

                if (!$attribute->getId() || $attribute->isScopeGlobal()) {
                    $scope = 'global';
                } elseif ($attribute->isScopeWebsite()) {
                    $scope = 'website';
                } else {
                    $scope = 'store';
                }

                $result[] = array(
                    'attribute_id' => $attribute->getId(),
                    'code' => $attribute->getAttributeCode(),
                    'type' => $attribute->getFrontendInput(),
                    'required' => $attribute->getIsRequired(),
                    'scope' => $scope
                );
            }
        }

        return $result;
    }
```

Method: options

```
$client->call($sessionId, 'catalog_product_attribute.options', $attributeId, $store);
```

Implemented In

Mage_Catalog_Model_Product_Attribute_Api::options
/Users/alanstorm/Sites2011/magento1point6point1.dev/app/code/core/Mage/Catalog/Model/Product/Attribute/Api.php

Doc Comment

```
/**
 * Retrieve attribute options
 *
 * @param int $attributeId
 * @param string|int $store
 * @return array
 */
```

Method Implementation Definition

```
    public function options($attributeId, $store = null)
    {
        $storeId = $this->_getStoreId($store);
        $attribute = Mage::getModel('catalog/product')
                ->setStoreId($storeId)
                ->getResource()
                ->getAttribute($attributeId);

        /* @var $attribute Mage_Catalog_Model_Entity_Attribute */
        if (!$attribute) {
            $this->_fault('not_exists');
        }
        $options = array();
        if ($attribute->usesSource()) {
            foreach ($attribute->getSource()->getAllOptions() as $optionId => $optionValue) {
```

```
                if (is_array($optionValue)) {
                    $options[] = $optionValue;
                } else {
                    $options[] = array(
                        'value' => $optionId,
                        'label' => $optionValue
                    );
                }
            }
        }

        return $options;
    }
```

Method: types

```
$client->call($sessionId, 'catalog_product_attribute.types');
```

Implemented In

```
Mage_Catalog_Model_Product_Attribute_Api::types
/Users/alanstorm/Sites2011/magento1point6point1.dev/app/code/core/Mage/Catalog/Model/Product/Attribute/Api.php
```

Doc Comment

```
/**
 * Retrieve list of possible attribute types
 *
 * @return array
 */
```

Method Implementation Definition

```
    public function types()
    {
        return Mage::getModel('catalog/product_attribute_source_inputtype')->toOptionArray();
    }
```

Method: create

```
$client->call($sessionId, 'catalog_product_attribute.create', $data);
```

Implemented In

```
Mage_Catalog_Model_Product_Attribute_Api::create
/Users/alanstorm/Sites2011/magento1point6point1.dev/app/code/core/Mage/Catalog/Model/Product/Attribute/Api.php
```

Doc Comment

```
/**
 * Create new product attribute
 *
 * @param array $data input data
 * @return integer
 */
```

Method Implementation Definition

```
    public function create($data)
    {
        /** @var $model Mage_Catalog_Model_Resource_Eav_Attribute */
        $model = Mage::getModel('catalog/resource_eav_attribute');
        /** @var $helper Mage_Catalog_Helper_Product */
        $helper = Mage::helper('catalog/product');

        if (empty($data['attribute_code']) || !is_array($data['frontend_label'])) {
            $this->_fault('invalid_parameters');
        }
```

```
        //validate attribute_code
        if (!preg_match('/^[a-z][a-z_0-9]{0,254}$/', $data['attribute_code'])) {
            $this->_fault('invalid_code');
        }

        //validate frontend_input
        $allowedTypes = array();
        foreach ($this->types() as $type) {
            $allowedTypes[] = $type['value'];
        }
        if (!in_array($data['frontend_input'], $allowedTypes)) {
            $this->_fault('invalid_frontend_input');
        }

        $data['source_model'] = $helper->getAttributeSourceModelByInputType($data['frontend_input']);
        $data['backend_model'] = $helper->getAttributeBackendModelByInputType($data['frontend_input']);
        if (is_null($model->getIsUserDefined()) || $model->getIsUserDefined() != 0) {
            $data['backend_type'] = $model->getBackendTypeByInput($data['frontend_input']);
        }

        $this->_prepareDataForSave($data);

        $model->addData($data);
        $model->setEntityTypeId($this->_entityTypeId);
        $model->setIsUserDefined(1);

        try {
            $model->save();
            // clear translation cache because attribute labels are stored in translation
            Mage::app()->cleanCache(array(Mage_Core_Model_Translate::CACHE_TAG));
        } catch (Exception $e) {
            $this->_fault('unable_to_save', $e->getMessage());
        }

        return (int) $model->getId();
    }
```

Method: update

```
$client->call($sessionId, 'catalog_product_attribute.update', $attribute, $data);
```

Implemented In

```
Mage_Catalog_Model_Product_Attribute_Api::update
/Users/alanstorm/Sites2011/magento1point6point1.dev/app/code/core/Mage/Catalog/Model/Product/Attribute/Api.php
```

Doc Comment

```
/**
 * Update product attribute
 *
 * @param string|integer $attribute attribute code or ID
 * @param array $data
 * @return boolean
 */
```

Method Implementation Definition

```
    public function update($attribute, $data)
    {
        $model = $this->_getAttribute($attribute);

        if ($model->getEntityTypeId() != $this->_entityTypeId) {
            $this->_fault('can_not_edit');
        }
```

```
        $data['attribute_code'] = $model->getAttributeCode();
        $data['is_user_defined'] = $model->getIsUserDefined();
        $data['frontend_input'] = $model->getFrontendInput();

        $this->_prepareDataForSave($data);

        $model->addData($data);
        try {
            $model->save();
            // clear translation cache because attribute labels are stored in translation
            Mage::app()->cleanCache(array(Mage_Core_Model_Translate::CACHE_TAG));
            return true;
        } catch (Exception $e) {
            $this->_fault('unable_to_save', $e->getMessage());
        }
    }
```

Method: remove

```
$client->call($sessionId, 'catalog_product_attribute.remove', $attribute);
```

Implemented In

```
Mage_Catalog_Model_Product_Attribute_Api::remove
/Users/alanstorm/Sites2011/magento1point6point1.dev/app/code/core/Mage/Catalog/Model/Product/Attribute/Api.php
```

Doc Comment

```
/**
 * Remove attribute
 *
 * @param integer|string $attribute attribute ID or code
 * @return boolean
 */
```

Method Implementation Definition

```
    public function remove($attribute)
    {
        $model = $this->_getAttribute($attribute);

        if ($model->getEntityTypeId() != $this->_entityTypeId) {
            $this->_fault('can_not_delete');
        }

        try {
            $model->delete();
            return true;
        } catch (Exception $e) {
            $this->_fault('can_not_delete', $e->getMessage());
        }
    }
```

Method: info

```
$client->call($sessionId, 'catalog_product_attribute.info', $attribute);
```

Implemented In

```
Mage_Catalog_Model_Product_Attribute_Api::info
/Users/alanstorm/Sites2011/magento1point6point1.dev/app/code/core/Mage/Catalog/Model/Product/Attribute/Api.php
```

Doc Comment

```
/**
 * Get full information about attribute with list of options
```

```
 *
 * @param integer|string $attribute attribute ID or code
 * @return array
 */
```

Method Implementation Definition

```
    public function info($attribute)
    {
        $model = $this->_getAttribute($attribute);

        if ($model->isScopeGlobal()) {
            $scope = 'global';
        } elseif ($model->isScopeWebsite()) {
            $scope = 'website';
        } else {
            $scope = 'store';
        }

        $frontendLabels = array(
            array(
                'store_id' => 0,
                'label' => $model->getFrontendLabel()
            )
        );
        foreach ($model->getStoreLabels() as $store_id => $label) {
            $frontendLabels[] = array(
                'store_id' => $store_id,
                'label' => $label
            );
        }

        $result = array(
            'attribute_id' => $model->getId(),
            'attribute_code' => $model->getAttributeCode(),
            'frontend_input' => $model->getFrontendInput(),
            'default_value' => $model->getDefaultValue(),
            'is_unique' => $model->getIsUnique(),
            'is_required' => $model->getIsRequired(),
            'apply_to' => $model->getApplyTo(),
            'is_configurable' => $model->getIsConfigurable(),
            'is_searchable' => $model->getIsSearchable(),
            'is_visible_in_advanced_search' => $model->getIsVisibleInAdvancedSearch(),
            'is_comparable' => $model->getIsComparable(),
            'is_used_for_promo_rules' => $model->getIsUsedForPromoRules(),
            'is_visible_on_front' => $model->getIsVisibleOnFront(),
            'used_in_product_listing' => $model->getUsedInProductListing(),
            'frontend_label' => $frontendLabels
        );
        if ($model->getFrontendInput() != 'price') {
            $result['scope'] = $scope;
        }

        // set additional fields to different types
        switch ($model->getFrontendInput()) {
            case 'text':
                    $result['additional_fields'] = array(
                        'frontend_class' => $model->getFrontendClass(),
                        'is_html_allowed_on_front' => $model->getIsHtmlAllowedOnFront(),
                        'used_for_sort_by' => $model->getUsedForSortBy()
                    );
                    break;
            case 'textarea':
```

```
                $result['additional_fields'] = array(
                    'is_wysiwyg_enabled' => $model->getIsWysiwygEnabled(),
                    'is_html_allowed_on_front' => $model->getIsHtmlAllowedOnFront(),
                );
                break;
            case 'date':
            case 'boolean':
                $result['additional_fields'] = array(
                    'used_for_sort_by' => $model->getUsedForSortBy()
                );
                break;
            case 'multiselect':
                $result['additional_fields'] = array(
                    'is_filterable' => $model->getIsFilterable(),
                    'is_filterable_in_search' => $model->getIsFilterableInSearch(),
                    'position' => $model->getPosition()
                );
                break;
            case 'select':
            case 'price':
                $result['additional_fields'] = array(
                    'is_filterable' => $model->getIsFilterable(),
                    'is_filterable_in_search' => $model->getIsFilterableInSearch(),
                    'position' => $model->getPosition(),
                    'used_for_sort_by' => $model->getUsedForSortBy()
                );
                break;
            default:
                $result['additional_fields'] = array();
                break;
        }

        // set options
        $options = $this->options($model->getId());
        // remove empty first element
        if ($model->getFrontendInput() != 'boolean') {
            array_shift($options);
        }

        if (count($options) > 0) {
            $result['options'] = $options;
        }

        return $result;
    }
```

Method: addOption

```
$client->call($sessionId, 'catalog_product_attribute.addOption', $attribute, $data);
```

Implemented In

```
Mage_Catalog_Model_Product_Attribute_Api::addOption
/Users/alanstorm/Sites2011/magento1point6point1.dev/app/code/core/Mage/Catalog/Model/Product/Attribute/Api.php
```

Doc Comment

```
/**
 * Add option to select or multiselect attribute
 *
 * @param  integer|string $attribute attribute ID or code
 * @param  array $data
 * @return bool
```

```
 */
```

Method Implementation Definition

```
    public function addOption($attribute, $data)
    {
        $model = $this->_getAttribute($attribute);

        if (!$model->usesSource()) {
            $this->_fault('invalid_frontend_input');
        }

        /** @var $helperCatalog Mage_Catalog_Helper_Data */
        $helperCatalog = Mage::helper('catalog');

        $optionLabels = array();
        foreach ($data['label'] as $label) {
            $storeId = $label['store_id'];
            $labelText = $helperCatalog->stripTags($label['value']);
            if (is_array($storeId)) {
                foreach ($storeId as $multiStoreId) {
                    $optionLabels[$multiStoreId] = $labelText;
                }
            } else {
                $optionLabels[$storeId] = $labelText;
            }
        }
        // data in the following format is accepted by the model
        // it simulates parameters of the request made to
        // Mage_Adminhtml_Catalog_Product_AttributeController::saveAction()
        $modelData = array(
            'option' => array(
                'value' => array(
                    'option_1' => $optionLabels
                ),
                'order' => array(
                    'option_1' => (int) $data['order']
                )
            )
        );
        if ($data['is_default']) {
            $modelData['default'][] = 'option_1';
        }

        $model->addData($modelData);
        try {
            $model->save();
        } catch (Exception $e) {
            $this->_fault('unable_to_add_option', $e->getMessage());
        }

        return true;
    }
```

Method: removeOption

```
$client->call($sessionId, 'catalog_product_attribute.removeOption', $attribute, $optionId);
```

Implemented In

Mage_Catalog_Model_Product_Attribute_Api::removeOption
/Users/alanstorm/Sites2011/magento1point6point1.dev/app/code/core/Mage/Catalog/Model/Product/Attribute/Api.php

Doc Comment

```
/**
 * Remove option from select or multiselect attribute
 *
 * @param  integer|string $attribute attribute ID or code
 * @param  integer $optionId option to remove ID
 * @return bool
 */
```

Method Implementation Definition

```
    public function removeOption($attribute, $optionId)
    {
        $model = $this->_getAttribute($attribute);

        if (!$model->usesSource()) {
            $this->_fault('invalid_frontend_input');
        }

        // data in the following format is accepted by the model
        // it simulates parameters of the request made to
        // Mage_Adminhtml_Catalog_Product_AttributeController::saveAction()
        $modelData = array(
            'option' => array(
                'value' => array(
                    $optionId => array()
                ),
                'delete' => array(
                    $optionId => '1'
                )
            )
        );
        $model->addData($modelData);
        try {
            $model->save();
        } catch (Exception $e) {
            $this->_fault('unable_to_remove_option', $e->getMessage());
        }

        return true;
    }
```

Resource: catalog_product_attribute_set

Method: create

```
$client->call($sessionId, 'catalog_product_attribute_set.create', $attributeSetName, $skeletonSetId);
```

Implemented In

Mage_Catalog_Model_Product_Attribute_Set_Api::create
/Users/alanstorm/Sites2011/magento1point6point1.dev/app/code/core/Mage/Catalog/Model/Product/Attribute/Set/Api.php

Doc Comment

```
/**
 * Create new attribute set based on another set
 *
 * @param string $attributeSetName
 * @param string $skeletonSetId
 * @return integer
 */
```

Method Implementation Definition

```
    public function create($attributeSetName, $skeletonSetId)
```

```
    {
        // check if set with requested $skeletonSetId exists
        if (!Mage::getModel('eav/entity_attribute_set')->load($skeletonSetId)->getId()){
            $this->_fault('invalid_skeleton_set_id');
        }
        // get catalog product entity type id
        $entityTypeId = Mage::getModel('catalog/product')->getResource()->getTypeId();
        /** @var $attributeSet Mage_Eav_Model_Entity_Attribute_Set */
        $attributeSet = Mage::getModel('eav/entity_attribute_set')
                ->setEntityTypeId($entityTypeId)
                ->setAttributeSetName($attributeSetName);
        try {
            // check if name is valid
            $attributeSet->validate();
            // copy parameters to new set from skeleton set
            $attributeSet->save();
            $attributeSet->initFromSkeleton($skeletonSetId)->save();
        } catch (Mage_Eav_Exception $e) {
            $this->_fault('invalid_data', $e->getMessage());
        } catch (Exception $e) {
            $this->_fault('create_attribute_set_error', $e->getMessage());
        }
        return (int)$attributeSet->getId();
    }
```

Method: remove

```
$client->call($sessionId, 'catalog_product_attribute_set.remove', $attributeSetId, $forceProductsRemove);
```

Implemented In

Mage_Catalog_Model_Product_Attribute_Set_Api::remove
/Users/alanstorm/Sites2011/magento1point6point1.dev/app/code/core/Mage/Catalog/Model/Product/Attribute/Set/Api.php

Doc Comment

```
/**
 * Remove attribute set
 *
 * @param string $attributeSetId
 * @param bool $forceProductsRemove
 * @return bool
 */
```

Method Implementation Definition

```
    public function remove($attributeSetId, $forceProductsRemove = false)
    {
        // if attribute set has related goods and $forceProductsRemove is not set throw exception
        if (!$forceProductsRemove) {
            /** @var $catalogProductsCollection Mage_Catalog_Model_Resource_Eav_Mysql4_Product_Collection */
            $catalogProductsCollection = Mage::getModel('catalog/product')->getCollection()
                    ->addFieldToFilter('attribute_set_id', $attributeSetId);
            if (count($catalogProductsCollection)) {
                $this->_fault('attribute_set_has_related_products');
            }
        }
        $attributeSet = Mage::getModel('eav/entity_attribute_set')->load($attributeSetId);
        // check if set with requested id exists
        if (!$attributeSet->getId()){
            $this->_fault('invalid_attribute_set_id');
        }
        try {
            $attributeSet->delete();
```

```
        } catch (Exception $e) {
            $this->_fault('remove_attribute_set_error', $e->getMessage());
        }
        return true;
    }
```

Method: list

```
$client->call($sessionId, 'catalog_product_attribute_set.list');
```

Implemented In

```
Mage_Catalog_Model_Product_Attribute_Set_Api::items
/Users/alanstorm/Sites2011/magento1point6point1.dev/app/code/core/Mage/Catalog/Model/Product/Attribute/Set/Api.php
```

Doc Comment

```
/**
 * Retrieve attribute set list
 *
 * @return array
 */
```

Method Implementation Definition

```
    public function items()
    {
        $entityType = Mage::getModel('catalog/product')->getResource()->getEntityType();
        $collection = Mage::getResourceModel('eav/entity_attribute_set_collection')
            ->setEntityTypeFilter($entityType->getId());

        $result = array();
        foreach ($collection as $attributeSet) {
            $result[] = array(
                'set_id' => $attributeSet->getId(),
                'name'   => $attributeSet->getAttributeSetName()
            );

        }

        return $result;
    }
```

Method: attributeAdd

```
$client->call($sessionId, 'catalog_product_attribute_set.attributeAdd', $attributeId, $attributeSetId, $attributeGroupId, $sortOrder);
```

Implemented In

```
Mage_Catalog_Model_Product_Attribute_Set_Api::attributeAdd
/Users/alanstorm/Sites2011/magento1point6point1.dev/app/code/core/Mage/Catalog/Model/Product/Attribute/Set/Api.php
```

Doc Comment

```
/**
 * Add attribute to attribute set
 *
 * @param string $attributeId
 * @param string $attributeSetId
 * @param string|null $attributeGroupId
 * @param string $sortOrder
 * @return bool
 */
```

Method Implementation Definition

```
    public function attributeAdd($attributeId, $attributeSetId, $attributeGroupId = null, $sortOrder = '0')
    {
        // check if attribute with requested id exists
        /** @var $attribute Mage_Eav_Model_Entity_Attribute */
        $attribute = Mage::getModel('eav/entity_attribute')->load($attributeId);
        if (!$attribute->getId()) {
            $this->_fault('invalid_attribute_id');
        }
        // check if attribute set with requested id exists
        /** @var $attributeSet Mage_Eav_Model_Entity_Attribute_Set */
        $attributeSet = Mage::getModel('eav/entity_attribute_set')->load($attributeSetId);
        if (!$attributeSet->getId()) {
            $this->_fault('invalid_attribute_set_id');
        }
        if (!empty($attributeGroupId)) {
            // check if attribute group with requested id exists
            if (!Mage::getModel('eav/entity_attribute_group')->load($attributeGroupId)->getId()) {
                $this->_fault('invalid_attribute_group_id');
            }
        } else {
            // define default attribute group id for current attribute set
            $attributeGroupId = $attributeSet->getDefaultGroupId();
        }
        $attribute->setAttributeSetId($attributeSet->getId())->loadEntityAttributeIdBySet();
        if ($attribute->getEntityAttributeId()) {
            $this->_fault('attribute_is_already_in_set');
        }
        try {
           $attribute->setEntityTypeId($attributeSet->getEntityTypeId())
                    ->setAttributeSetId($attributeSetId)
                    ->setAttributeGroupId($attributeGroupId)
                    ->setSortOrder($sortOrder)
                    ->save();
        } catch (Exception $e) {
            $this->_fault('add_attribute_error', $e->getMessage());
        }
        return true;
    }
```

Method: attributeRemove

```
$client->call($sessionId, 'catalog_product_attribute_set.attributeRemove', $attributeId, $attributeSetId);
```

Implemented In

Mage_Catalog_Model_Product_Attribute_Set_Api::attributeRemove
/Users/alanstorm/Sites2011/magento1point6point1.dev/app/code/core/Mage/Catalog/Model/Product/Attribute/Set/Api.php

Doc Comment

```
/**
 * Remove attribute from attribute set
 *
 * @param string $attributeId
 * @param string $attributeSetId
 * @return bool
 */
```

Method Implementation Definition

```
    public function attributeRemove($attributeId, $attributeSetId)
    {
        // check if attribute with requested id exists
        /** @var $attribute Mage_Eav_Model_Entity_Attribute */
```

```
        $attribute = Mage::getModel('eav/entity_attribute')->load($attributeId);
        if (!$attribute->getId()) {
            $this->_fault('invalid_attribute_id');
        }
        // check if attribute set with requested id exists
        /** @var $attributeSet Mage_Eav_Model_Entity_Attribute_Set */
        $attributeSet = Mage::getModel('eav/entity_attribute_set')->load($attributeSetId);
        if (!$attributeSet->getId()) {
            $this->_fault('invalid_attribute_set_id');
        }
        // check if attribute is in set
        $attribute->setAttributeSetId($attributeSet->getId())->loadEntityAttributeIdBySet();
        if (!$attribute->getEntityAttributeId()) {
            $this->_fault('attribute_is_not_in_set');
        }
        try {
            // delete record from eav_entity_attribute
            // using entity_attribute_id loaded by loadEntityAttributeIdBySet()
            $attribute->deleteEntity();
        } catch (Exception $e) {
            $this->_fault('remove_attribute_error', $e->getMessage());
        }

        return true;
    }
```

Method: groupAdd

```
$client->call($sessionId, 'catalog_product_attribute_set.groupAdd', $attributeSetId, $groupName);
```

Implemented In

Mage_Catalog_Model_Product_Attribute_Set_Api::groupAdd
/Users/alanstorm/Sites2011/magento1point6point1.dev/app/code/core/Mage/Catalog/Model/Product/Attribute/Set/Api.php

Doc Comment

```
/**
 * Create group within existing attribute set
 *
 * @param  string|int $attributeSetId
 * @param  string $groupName
 * @return int
 */
```

Method Implementation Definition

```
    public function groupAdd($attributeSetId, $groupName)
    {
        /** @var $group Mage_Eav_Model_Entity_Attribute_Group */
        $group = Mage::getModel('eav/entity_attribute_group');
        $group->setAttributeSetId($attributeSetId)
                ->setAttributeGroupName(
                    Mage::helper('catalog')->stripTags($groupName)
                );
        if ($group->itemExists()) {
            $this->_fault('group_already_exists');
        }
        try {
            $group->save();
        } catch (Exception $e) {
            $this->_fault('group_add_error', $e->getMessage());
        }
        return (int)$group->getId();
```

```
    }
```

Method: groupRename

```
$client->call($sessionId, 'catalog_product_attribute_set.groupRename', $groupId, $groupName);
```

Implemented In

```
Mage_Catalog_Model_Product_Attribute_Set_Api::groupRename
/Users/alanstorm/Sites2011/magento1point6point1.dev/app/code/core/Mage/Catalog/Model/Product/Attribute/Set/
Api.php
```

Doc Comment

```
/**
 * Rename existing group
 *
 * @param string|int $groupId
 * @param string $groupName
 * @return boolean
 */
```

Method Implementation Definition

```
    public function groupRename($groupId, $groupName)
    {
        $model = Mage::getModel('eav/entity_attribute_group')->load($groupId);

        if (!$model->getAttributeGroupName()) {
            $this->_fault('invalid_attribute_group_id');
        }

        $model->setAttributeGroupName(
                Mage::helper('catalog')->stripTags($groupName)
        );
        try {
            $model->save();
        } catch (Exception $e) {
            $this->_fault('group_rename_error', $e->getMessage());
        }
        return true;
    }
```

Method: groupRemove

```
$client->call($sessionId, 'catalog_product_attribute_set.groupRemove', $attributeGroupId);
```

Implemented In

```
Mage_Catalog_Model_Product_Attribute_Set_Api::groupRemove
/Users/alanstorm/Sites2011/magento1point6point1.dev/app/code/core/Mage/Catalog/Model/Product/Attribute/Set/
Api.php
```

Doc Comment

```
/**
    * Remove group from existing attribute set
    *
    * @param  string|int $attributeGroupId
    * @return bool
    */
```

Method Implementation Definition

```
       public function groupRemove($attributeGroupId)
       {
           /** @var $group Mage_Catalog_Model_Product_Attribute_Group */
           $group = Mage::getModel('catalog/product_attribute_group')->load($attributeGroupId);
           if (!$group->getId()) {
```

```
            $this->_fault('invalid_attribute_group_id');
        }
        if ($group->hasConfigurableAttributes()) {
            $this->_fault('group_has_configurable_attributes');
        }
        if ($group->hasSystemAttributes()) {
            $this->_fault('group_has_system_attributes');
        }
        try {
            $group->delete();
        } catch (Exception $e) {
            $this->_fault('group_remove_error', $e->getMessage());
        }
        return true;
    }
```

Resource: catalog_product_type

Method: list

```
$client->call($sessionId, 'catalog_product_type.list');
```

Implemented In

```
Mage_Catalog_Model_Product_Type_Api::items
/Users/alanstorm/Sites2011/magento1point6point1.dev/app/code/core/Mage/Catalog/Model/Product/Type/Api.php
```

Doc Comment

```
/**
 * Retrieve product type list
 *
 * @return array
 */
```

Method Implementation Definition

```
    public function items()
    {
        $result = array();

        foreach (Mage_Catalog_Model_Product_Type::getOptionArray() as $type=>$label) {
            $result[] = array(
                'type'  => $type,
                'label' => $label
            );
        }

        return $result;
    }
```

Resource: catalog_product_attribute_media

Method: currentStore

```
$client->call($sessionId, 'catalog_product_attribute_media.currentStore', $store);
```

Implemented In

```
Mage_Catalog_Model_Product_Attribute_Media_Api::currentStore
/Users/alanstorm/Sites2011/magento1point6point1.dev/app/code/core/Mage/Catalog/Model/Api/Resource.php
```

Doc Comment

```
/**
 * Set current store for catalog.
 *
```

```
 * @param string|int $store
 * @return int
 */
```

Method Implementation Definition

```
    public function currentStore($store=null)
    {
        if (!is_null($store)) {
            try {
                $storeId = Mage::app()->getStore($store)->getId();
            } catch (Mage_Core_Model_Store_Exception $e) {
                $this->_fault('store_not_exists');
            }

            $this->_getSession()->setData($this->_storeIdSessionField, $storeId);
        }

        return $this->_getStoreId();
    }
```

Method: list

```
$client->call($sessionId, 'catalog_product_attribute_media.list', $productId, $store, $identifierType);
```

Implemented In

Mage_Catalog_Model_Product_Attribute_Media_Api::items
/Users/alanstorm/Sites2011/magento1point6point1.dev/app/code/core/Mage/Catalog/Model/Product/Attribute/Media/Api.php

Doc Comment

```
/**
 * Retrieve images for product
 *
 * @param int|string $productId
 * @param string|int $store
 * @return array
 */
```

Method Implementation Definition

```
    public function items($productId, $store = null, $identifierType = null)
    {
        $product = $this->_initProduct($productId, $store, $identifierType);

        $gallery = $this->_getGalleryAttribute($product);

        $galleryData = $product->getData(self::ATTRIBUTE_CODE);

        if (!isset($galleryData['images']) || !is_array($galleryData['images'])) {
            return array();
        }

        $result = array();

        foreach ($galleryData['images'] as &$image) {
            $result[] = $this->_imageToArray($image, $product);
        }

        return $result;
    }
```

Method: info

```
$client->call($sessionId, 'catalog_product_attribute_media.info', $productId, $file, $store, $identifierType);
```

Implemented In

```
Mage_Catalog_Model_Product_Attribute_Media_Api::info
/Users/alanstorm/Sites2011/magento1point6point1.dev/app/code/core/Mage/Catalog/Model/Product/Attribute/Media/
Api.php
```

Doc Comment

```
/**
 * Retrieve image data
 *
 * @param int|string $productId
 * @param string $file
 * @param string|int $store
 * @return array
 */
```

Method Implementation Definition

```
    public function info($productId, $file, $store = null, $identifierType = null)
    {
        $product = $this->_initProduct($productId, $store, $identifierType);

        $gallery = $this->_getGalleryAttribute($product);

        if (!$image = $gallery->getBackend()->getImage($product, $file)) {
            $this->_fault('not_exists');
        }

        return $this->_imageToArray($image, $product);
    }
```

Method: types

```
$client->call($sessionId, 'catalog_product_attribute_media.types', $setId);
```

Implemented In

```
Mage_Catalog_Model_Product_Attribute_Media_Api::types
/Users/alanstorm/Sites2011/magento1point6point1.dev/app/code/core/Mage/Catalog/Model/Product/Attribute/Media/
Api.php
```

Doc Comment

```
/**
 * Retrieve image types (image, small_image, thumbnail, etc...)
 *
 * @param int $setId
 * @return array
 */
```

Method Implementation Definition

```
    public function types($setId)
    {
        $attributes = Mage::getModel('catalog/product')->getResource()
                ->loadAllAttributes()
                ->getSortedAttributes($setId);

        $result = array();

        foreach ($attributes as $attribute) {
            /* @var $attribute Mage_Catalog_Model_Resource_Eav_Attribute */
            if ($attribute->isInSet($setId)
                && $attribute->getFrontendInput() == 'media_image') {
                if ($attribute->isScopeGlobal()) {
                    $scope = 'global';
```

```
                } elseif ($attribute->isScopeWebsite()) {
                    $scope = 'website';
                } else {
                    $scope = 'store';
                }

                $result[] = array(
                    'code'         => $attribute->getAttributeCode(),
                    'scope'        => $scope
                );
            }
        }

        return $result;
    }
```

Method: create

```
$client->call($sessionId, 'catalog_product_attribute_media.create', $productId, $data, $store, $identifierType);
```

Implemented In

```
Mage_Catalog_Model_Product_Attribute_Media_Api::create
/Users/alanstorm/Sites2011/magento1point6point1.dev/app/code/core/Mage/Catalog/Model/Product/Attribute/Media/
Api.php
```

Doc Comment

```
/**
 * Create new image for product and return image filename
 *
 * @param int|string $productId
 * @param array $data
 * @param string|int $store
 * @return string
 */
```

Method Implementation Definition

```
    public function create($productId, $data, $store = null, $identifierType = null)
    {
        $data = $this->_prepareImageData($data);

        $product = $this->_initProduct($productId, $store, $identifierType);

        $gallery = $this->_getGalleryAttribute($product);

        if (!isset($data['file']) || !isset($data['file']['mime']) || !isset($data['file']['content'])) {
            $this->_fault('data_invalid', Mage::helper('catalog')->__('The image is not specified.'));
        }

        if (!isset($this->_mimeTypes[$data['file']['mime']])) {
            $this->_fault('data_invalid', Mage::helper('catalog')->__('Invalid image type.'));
        }

        $fileContent = @base64_decode($data['file']['content'], true);
        if (!$fileContent) {
            $this->_fault('data_invalid', Mage::helper('catalog')->__('The image contents is not valid base64
data.'));
        }

        unset($data['file']['content']);

        $tmpDirectory = Mage::getBaseDir('var') . DS . 'api' . DS . $this->_getSession()->getSessionId();
```

```
        if (isset($data['file']['name']) && $data['file']['name']) {
            $fileName  = $data['file']['name'];
        } else {
            $fileName  = 'image';
        }
        $fileName .= '.' . $this->_mimeTypes[$data['file']['mime']];

        $ioAdapter = new Varien_Io_File();
        try {
            // Create temporary directory for api
            $ioAdapter->checkAndCreateFolder($tmpDirectory);
            $ioAdapter->open(array('path'=>$tmpDirectory));
            // Write image file
            $ioAdapter->write($fileName, $fileContent, 0666);
            unset($fileContent);

            // Adding image to gallery
            $file = $gallery->getBackend()->addImage(
                $product,
                $tmpDirectory . DS . $fileName,
                null,
                true
            );

            // Remove temporary directory
            $ioAdapter->rmdir($tmpDirectory, true);

            $gallery->getBackend()->updateImage($product, $file, $data);

            if (isset($data['types'])) {
                $gallery->getBackend()->setMediaAttribute($product, $data['types'], $file);
            }

            $product->save();
        } catch (Mage_Core_Exception $e) {
            $this->_fault('not_created', $e->getMessage());
        } catch (Exception $e) {
            $this->_fault('not_created', Mage::helper('catalog')->__('Cannot create image.'));
        }

        return $gallery->getBackend()->getRenamedImage($file);
    }
```

Method: update

```
$client->call($sessionId, 'catalog_product_attribute_media.update', $productId, $file, $data, $store, $identifierType);
```

Implemented In

Mage_Catalog_Model_Product_Attribute_Media_Api::update
/Users/alanstorm/Sites2011/magento1point6point1.dev/app/code/core/Mage/Catalog/Model/Product/Attribute/Media/Api.php

Doc Comment

```
/**
 * Update image data
 *
 * @param int|string $productId
 * @param string $file
 * @param array $data
 * @param string|int $store
```

```
 * @return boolean
 */
```

Method Implementation Definition

```
    public function update($productId, $file, $data, $store = null, $identifierType = null)
    {
        $data = $this->_prepareImageData($data);

        $product = $this->_initProduct($productId, $store, $identifierType);

        $gallery = $this->_getGalleryAttribute($product);

        if (!$gallery->getBackend()->getImage($product, $file)) {
            $this->_fault('not_exists');
        }

        if (isset($data['file']['mime']) && isset($data['file']['content'])) {
            if (!isset($this->_mimeTypes[$data['file']['mime']])) {
                $this->_fault('data_invalid', Mage::helper('catalog')->__('Invalid image type.'));
            }

            $fileContent = @base64_decode($data['file']['content'], true);
            if (!$fileContent) {
                $this->_fault('data_invalid', Mage::helper('catalog')->__('Image content is not valid base64
data.'));
            }

            unset($data['file']['content']);

            $ioAdapter = new Varien_Io_File();
            try {
                $fileName = Mage::getBaseDir('media'). DS . 'catalog' . DS . 'product' . $file;
                $ioAdapter->open(array('path'=>dirname($fileName)));
                $ioAdapter->write(basename($fileName), $fileContent, 0666);

            } catch(Exception $e) {
                $this->_fault('not_created', Mage::helper('catalog')->__('Can\'t create image.'));
            }
        }

        $gallery->getBackend()->updateImage($product, $file, $data);

        if (isset($data['types']) && is_array($data['types'])) {
            $oldTypes = array();
            foreach ($product->getMediaAttributes() as $attribute) {
                if ($product->getData($attribute->getAttributeCode()) == $file) {
                     $oldTypes[] = $attribute->getAttributeCode();
                }
            }

            $clear = array_diff($oldTypes, $data['types']);

            if (count($clear) > 0) {
                $gallery->getBackend()->clearMediaAttribute($product, $clear);
            }

            $gallery->getBackend()->setMediaAttribute($product, $data['types'], $file);
        }

        try {
            $product->save();
        } catch (Mage_Core_Exception $e) {
```

```
            $this->_fault('not_updated', $e->getMessage());
        }

        return true;
    }
```

Method: remove

```
$client->call($sessionId, 'catalog_product_attribute_media.remove', $productId, $file, $identifierType);
```

Implemented In

```
Mage_Catalog_Model_Product_Attribute_Media_Api::remove
/Users/alanstorm/Sites2011/magento1point6point1.dev/app/code/core/Mage/Catalog/Model/Product/Attribute/Media/Api.php
```

Doc Comment

```
/**
 * Remove image from product
 *
 * @param int|string $productId
 * @param string $file
 * @return boolean
 */
```

Method Implementation Definition

```
    public function remove($productId, $file, $identifierType = null)
    {
        $product = $this->_initProduct($productId, null, $identifierType);

        $gallery = $this->_getGalleryAttribute($product);

        if (!$gallery->getBackend()->getImage($product, $file)) {
            $this->_fault('not_exists');
        }

        $gallery->getBackend()->removeImage($product, $file);

        try {
            $product->save();
        } catch (Mage_Core_Exception $e) {
            $this->_fault('not_removed', $e->getMessage());
        }

        return true;
    }
```

Resource: catalog_product_attribute_tier_price

Method: info

```
$client->call($sessionId, 'catalog_product_attribute_tier_price.info', $productId, $identifierType);
```

Implemented In

```
Mage_Catalog_Model_Product_Attribute_Tierprice_Api::info
/Users/alanstorm/Sites2011/magento1point6point1.dev/app/code/core/Mage/Catalog/Model/Product/Attribute/Tierprice/Api.php
```

Doc Comment

Method Implementation Definition

```
    public function info($productId, $identifierType = null)
```

```
    {
        $product = $this->_initProduct($productId, $identifierType);
        $tierPrices = $product->getData(self::ATTRIBUTE_CODE);

        if (!is_array($tierPrices)) {
            return array();
        }

        $result = array();

        foreach ($tierPrices as $tierPrice) {
            $row = array();
            $row['customer_group_id'] = (empty($tierPrice['all_groups']) ? $tierPrice['cust_group'] : 'all' );
            $row['website']           = ($tierPrice['website_id'] ?
                            Mage::app()->getWebsite($tierPrice['website_id'])->getCode() :
                            'all'
                    );
            $row['qty']               = $tierPrice['price_qty'];
            $row['price']             = $tierPrice['price'];

            $result[] = $row;
        }

        return $result;
    }
```

Method: update

```
$client->call($sessionId, 'catalog_product_attribute_tier_price.update', $productId, $tierPrices,
$identifierType);
```

Implemented In

```
Mage_Catalog_Model_Product_Attribute_Tierprice_Api::update
/Users/alanstorm/Sites2011/magento1point6point1.dev/app/code/core/Mage/Catalog/Model/Product/Attribute/
Tierprice/Api.php
```

Doc Comment

```
/**
 * Update tier prices of product
 *
 * @param int|string $productId
 * @param array $tierPrices
 * @return boolean
 */
```

Method Implementation Definition

```
    public function update($productId, $tierPrices, $identifierType = null)
    {
        $product = $this->_initProduct($productId, $identifierType);

        $updatedTierPrices = $this->prepareTierPrices($product, $tierPrices);
        if (is_null($updatedTierPrices)) {
            $this->_fault('data_invalid', Mage::helper('catalog')->__('Invalid Tier Prices'));
        }

        $product->setData(self::ATTRIBUTE_CODE, $updatedTierPrices);
        try {
            /**
             * @todo implement full validation process with errors returning which are ignoring now
             * @todo see Mage_Catalog_Model_Product::validate()
             */
            if (is_array($errors = $product->validate())) {
```

```
                $strErrors = array();
                foreach($errors as $code=>$error) {
                    $strErrors[] = ($error === true)? Mage::helper('catalog')->__('Value for "%s" is invalid.',
$code) : Mage::helper('catalog')->__('Value for "%s" is invalid: %s', $code, $error);
                }
                $this->_fault('data_invalid', implode("\n", $strErrors));
            }

            $product->save();
        } catch (Mage_Core_Exception $e) {
            $this->_fault('not_updated', $e->getMessage());
        }

        return true;
    }
```

Resource: catalog_product_link

Method: list

```
$client->call($sessionId, 'catalog_product_link.list', $type, $productId, $identifierType);
```

Implemented In

```
Mage_Catalog_Model_Product_Link_Api::items
/Users/alanstorm/Sites2011/magento1point6point1.dev/app/code/core/Mage/Catalog/Model/Product/Link/Api.php
```

Doc Comment

```
/**
 * Retrieve product link associations
 *
 * @param string $type
 * @param int|sku $productId
 * @param  string $identifierType
 * @return array
 */
```

Method Implementation Definition

```
    public function items($type, $productId, $identifierType = null)
    {
        $typeId = $this->_getTypeId($type);

        $product = $this->_initProduct($productId, $identifierType);

        $link = $product->getLinkInstance()
            ->setLinkTypeId($typeId);

        $collection = $this->_initCollection($link, $product);

        $result = array();

        foreach ($collection as $linkedProduct) {
            $row = array(
                'product_id' => $linkedProduct->getId(),
                'type'       => $linkedProduct->getTypeId(),
                'set'        => $linkedProduct->getAttributeSetId(),
                'sku'        => $linkedProduct->getSku()
            );

            foreach ($link->getAttributes() as $attribute) {
                $row[$attribute['code']] = $linkedProduct->getData($attribute['code']);
            }
```

```
            $result[] = $row;
        }

        return $result;
    }
```

Method: assign

```
$client->call($sessionId, 'catalog_product_link.assign', $type, $productId, $linkedProductId, $data, $identifierType);
```

Implemented In

```
Mage_Catalog_Model_Product_Link_Api::assign
/Users/alanstorm/Sites2011/magento1point6point1.dev/app/code/core/Mage/Catalog/Model/Product/Link/Api.php
```

Doc Comment

```
/**
 * Add product link association
 *
 * @param string $type
 * @param int|string $productId
 * @param int|string $linkedProductId
 * @param array $data
 * @param  string $identifierType
 * @return boolean
 */
```

Method Implementation Definition

```
    public function assign($type, $productId, $linkedProductId, $data = array(), $identifierType = null)
    {
        $typeId = $this->_getTypeId($type);

        $product = $this->_initProduct($productId, $identifierType);

        $link = $product->getLinkInstance()
            ->setLinkTypeId($typeId);

        $collection = $this->_initCollection($link, $product);
        $idBySku = $product->getIdBySku($linkedProductId);
        if ($idBySku) {
            $linkedProductId = $idBySku;
        }

        $links = $this->_collectionToEditableArray($collection);

        $links[(int)$linkedProductId] = array();

        foreach ($collection->getLinkModel()->getAttributes() as $attribute) {
            if (isset($data[$attribute['code']])) {
                $links[(int)$linkedProductId][$attribute['code']] = $data[$attribute['code']];
            }
        }

        try {
            if ($type == 'grouped') {
                $link->getResource()->saveGroupedLinks($product, $links, $typeId);
            } else {
                $link->getResource()->saveProductLinks($product, $links, $typeId);
            }

            $_linkInstance = Mage::getSingleton('catalog/product_link');
            $_linkInstance->saveProductRelations($product);
```

```
            $indexerStock = Mage::getModel('cataloginventory/stock_status');
            $indexerStock->updateStatus($productId);

            $indexerPrice = Mage::getResourceModel('catalog/product_indexer_price');
            $indexerPrice->reindexProductIds($productId);
        } catch (Exception $e) {
            $this->_fault('data_invalid', Mage::helper('catalog')->__('Link product does not exist.'));
        }

        return true;
    }
```

Method: update

```
$client->call($sessionId, 'catalog_product_link.update', $type, $productId, $linkedProductId, $data,
$identifierType);
```

Implemented In

```
Mage_Catalog_Model_Product_Link_Api::update
/Users/alanstorm/Sites2011/magento1point6point1.dev/app/code/core/Mage/Catalog/Model/Product/Link/Api.php
```

Doc Comment

```
/**
 * Update product link association info
 *
 * @param string $type
 * @param int|string $productId
 * @param int|string $linkedProductId
 * @param array $data
 * @param  string $identifierType
 * @return boolean
 */
```

Method Implementation Definition

```
    public function update($type, $productId, $linkedProductId, $data = array(), $identifierType = null)
    {
        $typeId = $this->_getTypeId($type);

        $product = $this->_initProduct($productId, $identifierType);

        $link = $product->getLinkInstance()
            ->setLinkTypeId($typeId);

        $collection = $this->_initCollection($link, $product);

        $links = $this->_collectionToEditableArray($collection);

        $idBySku = $product->getIdBySku($linkedProductId);
        if ($idBySku) {
            $linkedProductId = $idBySku;
        }

        foreach ($collection->getLinkModel()->getAttributes() as $attribute) {
            if (isset($data[$attribute['code']])) {
                $links[(int)$linkedProductId][$attribute['code']] = $data[$attribute['code']];
            }
        }

        try {
            if ($type == 'grouped') {
                $link->getResource()->saveGroupedLinks($product, $links, $typeId);
```

```
            } else {
                $link->getResource()->saveProductLinks($product, $links, $typeId);
            }

            $_linkInstance = Mage::getSingleton('catalog/product_link');
            $_linkInstance->saveProductRelations($product);

            $indexerStock = Mage::getModel('cataloginventory/stock_status');
            $indexerStock->updateStatus($productId);

            $indexerPrice = Mage::getResourceModel('catalog/product_indexer_price');
            $indexerPrice->reindexProductIds($productId);
        } catch (Exception $e) {
            $this->_fault('data_invalid', Mage::helper('catalog')->__('Link product does not exist.'));
        }

        return true;
    }
```

Method: remove

```
$client->call($sessionId, 'catalog_product_link.remove', $type, $productId, $linkedProductId, $identifierType);
```

Implemented In

Mage_Catalog_Model_Product_Link_Api::remove
/Users/alanstorm/Sites2011/magento1point6point1.dev/app/code/core/Mage/Catalog/Model/Product/Link/Api.php

Doc Comment

```
/**
 * Remove product link association
 *
 * @param string $type
 * @param int|string $productId
 * @param int|string $linkedProductId
 * @param  string $identifierType
 * @return boolean
 */
```

Method Implementation Definition

```
    public function remove($type, $productId, $linkedProductId, $identifierType = null)
    {
        $typeId = $this->_getTypeId($type);

        $product = $this->_initProduct($productId, $identifierType);

        $link = $product->getLinkInstance()
            ->setLinkTypeId($typeId);

        $collection = $this->_initCollection($link, $product);

        $idBySku = $product->getIdBySku($linkedProductId);
        if ($idBySku) {
            $linkedProductId = $idBySku;
        }

        $links = $this->_collectionToEditableArray($collection);

        if (isset($links[$linkedProductId])) {
            unset($links[$linkedProductId]);
        }

        try {
```

```
            $link->getResource()->saveProductLinks($product, $links, $typeId);
        } catch (Exception $e) {
            $this->_fault('not_removed');
        }

        return true;
    }
```

Method: types

```
$client->call($sessionId, 'catalog_product_link.types');
```

Implemented In

```
Mage_Catalog_Model_Product_Link_Api::types
/Users/alanstorm/Sites2011/magento1point6point1.dev/app/code/core/Mage/Catalog/Model/Product/Link/Api.php
```

Doc Comment

```
/**
 * Retrieve link types
 *
 * @return array
 */
```

Method Implementation Definition

```
    public function types()
    {
        return array_keys($this->_typeMap);
    }
```

Method: attributes

```
$client->call($sessionId, 'catalog_product_link.attributes', $type);
```

Implemented In

```
Mage_Catalog_Model_Product_Link_Api::attributes
/Users/alanstorm/Sites2011/magento1point6point1.dev/app/code/core/Mage/Catalog/Model/Product/Link/Api.php
```

Doc Comment

```
/**
 * Retrieve attribute list for specified type
 *
 * @param string $type
 * @return array
 */
```

Method Implementation Definition

```
    public function attributes($type)
    {
        $typeId = $this->_getTypeId($type);

        $attributes = Mage::getModel('catalog/product_link')
            ->getAttributes($typeId);

        $result = array();

        foreach ($attributes as $attribute) {
            $result[] = array(
                'code'  => $attribute['code'],
                'type'  => $attribute['type']
            );
        }

        return $result;
```

```
    }
```

Resource: catalog_product_custom_option

Method: add

```
$client->call($sessionId, 'catalog_product_custom_option.add', $productId, $data, $store);
```

Implemented In

```
Mage_Catalog_Model_Product_Option_Api::add
/Users/alanstorm/Sites2011/magento1point6point1.dev/app/code/core/Mage/Catalog/Model/Product/Option/Api.php
```

Doc Comment

```
/**
 * Add custom option to product
 *
 * @param string $productId
 * @param array $data
 * @param int|string|null $store
 * @return bool $isAdded
 */
```

Method Implementation Definition

```
    public function add($productId, $data, $store = null)
    {
        $product = $this->_getProduct($productId, $store, null);
        if (!(is_array($data['additional_fields']) and count($data['additional_fields']))) {
            $this->_fault('invalid_data');
        }
        if (!$this->_isTypeAllowed($data['type'])) {
            $this->_fault('invalid_type');
        }
        $this->_prepareAdditionalFields(
            $data,
            $product->getOptionInstance()->getGroupByType($data['type'])
        );
        $this->_saveProductCustomOption($product, $data);
        return true;
    }
```

Method: update

```
$client->call($sessionId, 'catalog_product_custom_option.update', $optionId, $data, $store);
```

Implemented In

```
Mage_Catalog_Model_Product_Option_Api::update
/Users/alanstorm/Sites2011/magento1point6point1.dev/app/code/core/Mage/Catalog/Model/Product/Option/Api.php
```

Doc Comment

```
/**
 * Update product custom option data
 *
 * @param string $optionId
 * @param array $data
 * @param int|string|null $store
 * @return bool
 */
```

Method Implementation Definition

```
    public function update($optionId, $data, $store = null)
    {
        /** @var $option Mage_Catalog_Model_Product_Option */
```

```
        $option = Mage::getModel('catalog/product_option')->load($optionId);
        if (!$option->getId()) {
            $this->_fault('option_not_exists');
        }
        $product = $this->_getProduct($option->getProductId(), $store, null);
        $option = $product->getOptionById($optionId);
        if (isset($data['type']) and !$this->_isTypeAllowed($data['type'])) {
            $this->_fault('invalid_type');
        }
        if (isset($data['additional_fields'])) {
            $this->_prepareAdditionalFields(
                $data,
                $option->getGroupByType()
            );
        }
        foreach ($option->getValues() as $valueId => $value) {
            if(isset($data['values'][$valueId])) {
                $data['values'][$valueId] = array_merge($value->getData(), $data['values'][$valueId]);
            }
        }
        $data = array_merge($option->getData(), $data);
        $this->_saveProductCustomOption($product, $data);
        return true;
    }
```

Method: types

```
$client->call($sessionId, 'catalog_product_custom_option.types');
```

Implemented In

```
Mage_Catalog_Model_Product_Option_Api::types
/Users/alanstorm/Sites2011/magento1point6point1.dev/app/code/core/Mage/Catalog/Model/Product/Option/Api.php
```

Doc Comment

```
/**
 * Read list of possible custom option types from module config
 *
 * @return array
 */
```

Method Implementation Definition

```
    public function types()
    {
        $path = Mage_Adminhtml_Model_System_Config_Source_Product_Options_Type::PRODUCT_OPTIONS_GROUPS_PATH;
        $types = array();
        foreach (Mage::getConfig()->getNode($path)->children() as $group) {
            $groupTypes = Mage::getConfig()->getNode($path . '/' . $group->getName() . '/types')->children();
            /** @var $type Mage_Core_Model_Config_Element */
            foreach($groupTypes as $type){
                $labelPath = $path . '/' . $group->getName() . '/types/' . $type->getName() . '/label';
                $types[] = array(
                    'label' => (string) Mage::getConfig()->getNode($labelPath),
                    'value' => $type->getName()
                );
            }
        }
        return $types;
    }
```

Method: info

```
$client->call($sessionId, 'catalog_product_custom_option.info', $optionId, $store);
```

Implemented In

```
Mage_Catalog_Model_Product_Option_Api::info
/Users/alanstorm/Sites2011/magento1point6point1.dev/app/code/core/Mage/Catalog/Model/Product/Option/Api.php
```

Doc Comment

```
/**
 * Get full information about custom option in product
 *
 * @param int|string $optionId
 * @param  int|string|null $store
 * @return array
 */
```

Method Implementation Definition

```
    public function info($optionId, $store = null)
    {
        /** @var $option Mage_Catalog_Model_Product_Option */
        $option = Mage::getModel('catalog/product_option')->load($optionId);
        if (!$option->getId()) {
            $this->_fault('option_not_exists');
        }
        /** @var $product Mage_Catalog_Model_Product */
        $product = $this->_getProduct($option->getProductId(), $store, null);
        $option = $product->getOptionById($optionId);
        $result = array(
            'title' => $option->getTitle(),
            'type' => $option->getType(),
            'is_require' => $option->getIsRequire(),
            'sort_order' => $option->getSortOrder(),
            // additional_fields should be two-dimensional array for all option types
            'additional_fields' => array(
                array(
                    'price' => $option->getPrice(),
                    'price_type' => $option->getPriceType(),
                    'sku' => $option->getSku()
                )
            )
        );
        // Set additional fields to each type group
        switch ($option->getGroupByType()) {
            case Mage_Catalog_Model_Product_Option::OPTION_GROUP_TEXT:
                $result['additional_fields'][0]['max_characters'] = $option->getMaxCharacters();
                break;
            case Mage_Catalog_Model_Product_Option::OPTION_GROUP_FILE:
                $result['additional_fields'][0]['file_extension'] = $option->getFileExtension();
                $result['additional_fields'][0]['image_size_x'] = $option->getImageSizeX();
                $result['additional_fields'][0]['image_size_y'] = $option->getImageSizeY();
                break;
            case Mage_Catalog_Model_Product_Option::OPTION_GROUP_SELECT:
                $result['additional_fields'] = array();
                foreach ($option->getValuesCollection() as $value) {
                    $result['additional_fields'][] = array(
                        'value_id' => $value->getId(),
                        'title' => $value->getTitle(),
                        'price' => $value->getPrice(),
                        'price_type' => $value->getPriceType(),
                        'sku' => $value->getSku(),
                        'sort_order' => $value->getSortOrder()
                    );
                }
```

```
                break;
            default:
                break;
        }

        return $result;
    }
```

Method: list

```
$client->call($sessionId, 'catalog_product_custom_option.list', $productId, $store);
```

Implemented In

```
Mage_Catalog_Model_Product_Option_Api::items
/Users/alanstorm/Sites2011/magento1point6point1.dev/app/code/core/Mage/Catalog/Model/Product/Option/Api.php
```

Doc Comment

```
/**
 * Retrieve list of product custom options
 *
 * @param  string $productId
 * @param  int|string|null $store
 * @return array
 */
```

Method Implementation Definition

```
    public function items($productId, $store = null)
    {
        $result = array();
        $product = $this->_getProduct($productId, $store, null);
        /** @var $option Mage_Catalog_Model_Product_Option */
        foreach ($product->getProductOptionsCollection() as $option) {
            $result[] = array(
                'option_id' => $option->getId(),
                'title' => $option->getTitle(),
                'type' => $option->getType(),
                'is_require' => $option->getIsRequire(),
                'sort_order' => $option->getSortOrder()
            );
        }
        return $result;
    }
```

Method: remove

```
$client->call($sessionId, 'catalog_product_custom_option.remove', $optionId);
```

Implemented In

```
Mage_Catalog_Model_Product_Option_Api::remove
/Users/alanstorm/Sites2011/magento1point6point1.dev/app/code/core/Mage/Catalog/Model/Product/Option/Api.php
```

Doc Comment

```
/**
 * Remove product custom option
 *
 * @param string $optionId
 * @return boolean
 */
```

Method Implementation Definition

```
    public function remove($optionId)
    {
        /** @var $option Mage_Catalog_Model_Product_Option */
```

```
        $option = Mage::getModel('catalog/product_option')->load($optionId);
        if (!$option->getId()) {
            $this->_fault('option_not_exists');
        }
        try {
            $option->getValueInstance()->deleteValue($optionId);
            $option->deletePrices($optionId);
            $option->deleteTitles($optionId);
            $option->delete();
        } catch (Exception $e){
            $this->fault('delete_option_error');
        }
        return true;
    }
```

Resource: catalog_product_custom_option_value

Method: list

```
$client->call($sessionId, 'catalog_product_custom_option_value.list', $optionId, $store);
```

Implemented In

```
Mage_Catalog_Model_Product_Option_Value_Api::items
/Users/alanstorm/Sites2011/magento1point6point1.dev/app/code/core/Mage/Catalog/Model/Product/Option/Value/
Api.php
```

Doc Comment

```
/**
 * Retrieve values from specified option
 *
 * @param string $optionId
 * @param int|string|null $store
 * @return array
 */
```

Method Implementation Definition

```
    public function items($optionId, $store = null)
    {
        /** @var $option Mage_Catalog_Model_Product_Option */
        $option = $this->_prepareOption($optionId, $store);
        $productOptionValues = $option->getValuesCollection();
        $result = array();
        foreach($productOptionValues as $value){
            $result[] = array(
                'value_id' => $value->getId(),
                'title' => $value->getTitle(),
                'price' => $value->getPrice(),
                'price_type' => $value->getPriceType(),
                'sku' => $value->getSku(),
                'sort_order' => $value->getSortOrder()
            );
        }
        return $result;
    }
```

Method: info

```
$client->call($sessionId, 'catalog_product_custom_option_value.info', $valueId, $store);
```

Implemented In

```
Mage_Catalog_Model_Product_Option_Value_Api::info
```

/Users/alanstorm/Sites2011/magento1point6point1.dev/app/code/core/Mage/Catalog/Model/Product/Option/Value/Api.php

Doc Comment

```
/**
 * Retrieve specified option value info
 *
 * @param string $valueId
 * @param int|string|null $store
 * @return array
 */
```

Method Implementation Definition

```
    public function info($valueId, $store = null)
    {
        /** @var $productOptionValue Mage_Catalog_Model_Product_Option_Value */
        $productOptionValue = Mage::getModel('catalog/product_option_value')->load($valueId);
        if (!$productOptionValue->getId()) {
            $this->_fault('value_not_exists');
        }
        $storeId = $this->_getStoreId($store);
        $productOptionValues = $productOptionValue
                ->getValuesByOption(
                    array($valueId),
                    $productOptionValue->getOptionId(),
                    $storeId
                )
                ->addTitleToResult($storeId)
                ->addPriceToResult($storeId);

        $result = $productOptionValues->toArray();
        // reset can be used as the only item is expected
        $result = reset($result['items']);
        if (empty($result)) {
            $this->_fault('value_not_exists');
        }
        // map option_type_id to value_id
        $result['value_id'] = $result['option_type_id'];
        unset($result['option_type_id']);
        return $result;
    }
```

Method: add

```
$client->call($sessionId, 'catalog_product_custom_option_value.add', $optionId, $data, $store);
```

Implemented In

Mage_Catalog_Model_Product_Option_Value_Api::add
/Users/alanstorm/Sites2011/magento1point6point1.dev/app/code/core/Mage/Catalog/Model/Product/Option/Value/Api.php

Doc Comment

```
/**
 * Add new values to select option
 *
 * @param string $optionId
 * @param array $data
 * @param int|string|null $store
 * @return bool
 */
```

Method Implementation Definition

```
    public function add($optionId, $data, $store = null)
```

```
    {
        /** @var $option Mage_Catalog_Model_Product_Option */
        $option = $this->_prepareOption($optionId, $store);
        /** @var $optionValueModel Mage_Catalog_Model_Product_Option_Value */
        $optionValueModel = Mage::getModel('catalog/product_option_value');
        $optionValueModel->setOption($option);
        foreach ($data as &$optionValue) {
            foreach ($optionValue as &$value) {
                $value = Mage::helper('catalog')->stripTags($value);
            }
        }
        $optionValueModel->setValues($data);
        try {
            $optionValueModel->saveValues();
        } catch (Exception $e) {
            $this->_fault('add_option_value_error', $e->getMessage());
        }
        return true;
    }
```

Method: update

```
$client->call($sessionId, 'catalog_product_custom_option_value.update', $valueId, $data, $store);
```

Implemented In

Mage_Catalog_Model_Product_Option_Value_Api::update
/Users/alanstorm/Sites2011/magento1point6point1.dev/app/code/core/Mage/Catalog/Model/Product/Option/Value/Api.php

Doc Comment

```
/**
 * Update value to select option
 *
 * @param string $valueId
 * @param array $data
 * @param int|string|null $store
 * @return bool
 */
```

Method Implementation Definition

```
    public function update($valueId, $data, $store = null)
    {
        /** @var $productOptionValue Mage_Catalog_Model_Product_Option_Value */
        $productOptionValue = Mage::getModel('catalog/product_option_value')->load($valueId);
        if (!$productOptionValue->getId()) {
            $this->_fault('value_not_exists');
        }

        /** @var $option Mage_Catalog_Model_Product_Option */
        $option = $this->_prepareOption($productOptionValue->getOptionId(), $store);
        if (!$option->getId()) {
            $this->_fault('option_not_exists');
        }
        $productOptionValue->setOption($option);
        // Sanitize data
        foreach ($data as $key => $value) {
            $data[$key] = Mage::helper('catalog')->stripTags($value);
        }
        if (!isset($data['title']) OR empty($data['title'])) {
            $this->_fault('option_value_title_required');
        }
        $data['option_type_id'] = $valueId;
```

```
        $data['store_id'] = $this->_getStoreId($store);
        $productOptionValue->addValue($data);
        $productOptionValue->setData($data);

        try {
            $productOptionValue->save()->saveValues();
        } catch (Exception $e) {
            $this->_fault('update_option_value_error', $e->getMessage());
        }

        return true;
    }
```

Method: remove

```
$client->call($sessionId, 'catalog_product_custom_option_value.remove', $valueId);
```

Implemented In

```
Mage_Catalog_Model_Product_Option_Value_Api::remove
/Users/alanstorm/Sites2011/magento1point6point1.dev/app/code/core/Mage/Catalog/Model/Product/Option/Value/
Api.php
```

Doc Comment

```
/**
 * Delete value from select option
 *
 * @param int $valueId
 * @return boolean
 */
```

Method Implementation Definition

```
    public function remove($valueId)
    {
        /** @var $optionValue Mage_Catalog_Model_Product_Option_Value */
        $optionValue = Mage::getModel('catalog/product_option_value')->load($valueId);
        if (!$optionValue->getId()) {
            $this->_fault('value_not_exists');
        }

        // check values count
        if(count($this->items($optionValue->getOptionId())) <= 1){
            $this->_fault('cant_delete_last_value');
        }

        try {
            $optionValue->deleteValues($valueId);
        } catch (Mage_Core_Exception $e) {
            $this->_fault('not_deleted', $e->getMessage());
        }

        return true;
    }
```

Resource: sales_order

Method: list

```
$client->call($sessionId, 'sales_order.list', $filters);
```

Implemented In

```
Mage_Sales_Model_Order_Api::items
```

/Users/alanstorm/Sites2011/magento1point6point1.dev/app/code/core/Mage/Sales/Model/Order/Api.php

Doc Comment

```
/**
 * Retrieve list of orders by filters
 *
 * @param array $filters
 * @return array
 */
```

Method Implementation Definition

```
    public function items($filters = null)
    {
        //TODO: add full name logic
        $billingAliasName = 'billing_o_a';
        $shippingAliasName = 'shipping_o_a';

        $collection = Mage::getModel("sales/order")->getCollection()
            ->addAttributeToSelect('*')
            ->addAddressFields()
            ->addExpressionFieldToSelect(
                'billing_firstname', "{{billing_firstname}}", array
('billing_firstname'=>"$billingAliasName.firstname")
            )
            ->addExpressionFieldToSelect(
                'billing_lastname', "{{billing_lastname}}", array
('billing_lastname'=>"$billingAliasName.lastname")
            )
            ->addExpressionFieldToSelect(
                'shipping_firstname', "{{shipping_firstname}}", array
('shipping_firstname'=>"$shippingAliasName.firstname")
            )
            ->addExpressionFieldToSelect(
                'shipping_lastname', "{{shipping_lastname}}", array
('shipping_lastname'=>"$shippingAliasName.lastname")
            )
            ->addExpressionFieldToSelect(
                    'billing_name',
                    "CONCAT({{billing_firstname}}, ' ', {{billing_lastname}})",
                    array('billing_firstname'=>"$billingAliasName.firstname",
'billing_lastname'=>"$billingAliasName.lastname")
            )
            ->addExpressionFieldToSelect(
                    'shipping_name',
                    'CONCAT({{shipping_firstname}}, " ", {{shipping_lastname}})',
                    array('shipping_firstname'=>"$shippingAliasName.firstname",
'shipping_lastname'=>"$shippingAliasName.lastname")
            );

        if (is_array($filters)) {
            try {
                foreach ($filters as $field => $value) {
                    if (isset($this->_attributesMap['order'][$field])) {
                        $field = $this->_attributesMap['order'][$field];
                    }

                    $collection->addFieldToFilter($field, $value);
                }
            } catch (Mage_Core_Exception $e) {
                $this->_fault('filters_invalid', $e->getMessage());
            }
        }
```

```
        $result = array();

        foreach ($collection as $order) {
            $result[] = $this->_getAttributes($order, 'order');
        }

        return $result;
    }
```

Method: info

```
$client->call($sessionId, 'sales_order.info', $orderIncrementId);
```

Implemented In

```
Mage_Sales_Model_Order_Api::info
/Users/alanstorm/Sites2011/magento1point6point1.dev/app/code/core/Mage/Sales/Model/Order/Api.php
```

Doc Comment

```
/**
 * Retrieve full order information
 *
 * @param string $orderIncrementId
 * @return array
 */
```

Method Implementation Definition

```
    public function info($orderIncrementId)
    {
        $order = $this->_initOrder($orderIncrementId);

        if ($order->getGiftMessageId() > 0) {
            $order->setGiftMessage(
                Mage::getSingleton('giftmessage/message')->load($order->getGiftMessageId())->getMessage()
            );
        }

        $result = $this->_getAttributes($order, 'order');

        $result['shipping_address'] = $this->_getAttributes($order->getShippingAddress(), 'order_address');
        $result['billing_address']  = $this->_getAttributes($order->getBillingAddress(), 'order_address');
        $result['items'] = array();

        foreach ($order->getAllItems() as $item) {
            if ($item->getGiftMessageId() > 0) {
                $item->setGiftMessage(
                    Mage::getSingleton('giftmessage/message')->load($item->getGiftMessageId())->getMessage()
                );
            }

            $result['items'][] = $this->_getAttributes($item, 'order_item');
        }

        $result['payment'] = $this->_getAttributes($order->getPayment(), 'order_payment');

        $result['status_history'] = array();

        foreach ($order->getAllStatusHistory() as $history) {
            $result['status_history'][] = $this->_getAttributes($history, 'order_status_history');
        }

        return $result;
```

```
    }
```

Method: addComment

```
$client->call($sessionId, 'sales_order.addComment', $orderIncrementId, $status, $comment, $notify);
```

Implemented In

```
Mage_Sales_Model_Order_Api::addComment
/Users/alanstorm/Sites2011/magento1point6point1.dev/app/code/core/Mage/Sales/Model/Order/Api.php
```

Doc Comment

```
/**
 * Add comment to order
 *
 * @param string $orderIncrementId
 * @param string $status
 * @param string $comment
 * @param boolean $notify
 * @return boolean
 */
```

Method Implementation Definition

```
    public function addComment($orderIncrementId, $status, $comment = null, $notify = false)
    {
        $order = $this->_initOrder($orderIncrementId);

        $order->addStatusToHistory($status, $comment, $notify);


        try {
            if ($notify && $comment) {
                $oldStore = Mage::getDesign()->getStore();
                $oldArea = Mage::getDesign()->getArea();
                Mage::getDesign()->setStore($order->getStoreId());
                Mage::getDesign()->setArea('frontend');
            }

            $order->save();
            $order->sendOrderUpdateEmail($notify, $comment);
            if ($notify && $comment) {
                Mage::getDesign()->setStore($oldStore);
                Mage::getDesign()->setArea($oldArea);
            }

        } catch (Mage_Core_Exception $e) {
            $this->_fault('status_not_changed', $e->getMessage());
        }

        return true;
    }
```

Method: hold

```
$client->call($sessionId, 'sales_order.hold', $orderIncrementId);
```

Implemented In

```
Mage_Sales_Model_Order_Api::hold
/Users/alanstorm/Sites2011/magento1point6point1.dev/app/code/core/Mage/Sales/Model/Order/Api.php
```

Doc Comment

```
/**
 * Hold order
```

```
 *
 * @param string $orderIncrementId
 * @return boolean
 */
```

Method Implementation Definition

```
    public function hold($orderIncrementId)
    {
        $order = $this->_initOrder($orderIncrementId);

        try {
            $order->hold();
            $order->save();
        } catch (Mage_Core_Exception $e) {
            $this->_fault('status_not_changed', $e->getMessage());
        }

        return true;
    }
```

Method: unhold

```
$client->call($sessionId, 'sales_order.unhold', $orderIncrementId);
```

Implemented In

```
Mage_Sales_Model_Order_Api::unhold
/Users/alanstorm/Sites2011/magento1point6point1.dev/app/code/core/Mage/Sales/Model/Order/Api.php
```

Doc Comment

```
/**
 * Unhold order
 *
 * @param string $orderIncrementId
 * @return boolean
 */
```

Method Implementation Definition

```
    public function unhold($orderIncrementId)
    {
        $order = $this->_initOrder($orderIncrementId);

        try {
            $order->unhold();
            $order->save();
        } catch (Mage_Core_Exception $e) {
            $this->_fault('status_not_changed', $e->getMessage());
        }

        return true;
    }
```

Method: cancel

```
$client->call($sessionId, 'sales_order.cancel', $orderIncrementId);
```

Implemented In

```
Mage_Sales_Model_Order_Api::cancel
/Users/alanstorm/Sites2011/magento1point6point1.dev/app/code/core/Mage/Sales/Model/Order/Api.php
```

Doc Comment

```
/**
 * Cancel order
 *
```

```
 * @param string $orderIncrementId
 * @return boolean
 */
```

Method Implementation Definition

```
    public function cancel($orderIncrementId)
    {
        $order = $this->_initOrder($orderIncrementId);

        try {
            $order->cancel();
            $order->save();
        } catch (Mage_Core_Exception $e) {
            $this->_fault('status_not_changed', $e->getMessage());
        }

        return true;
    }
```

Resource: sales_order_shipment

Method: list

```
$client->call($sessionId, 'sales_order_shipment.list', $filters);
```

Implemented In

```
Mage_Sales_Model_Order_Shipment_Api::items
/Users/alanstorm/Sites2011/magento1point6point1.dev/app/code/core/Mage/Sales/Model/Order/Shipment/Api.php
```

Doc Comment

```
/**
 * Retrive shipments by filters
 *
 * @param array $filters
 * @return array
 */
```

Method Implementation Definition

```
    public function items($filters = null)
    {
        //TODO: add full name logic
        $collection = Mage::getResourceModel('sales/order_shipment_collection')
            ->addAttributeToSelect('increment_id')
            ->addAttributeToSelect('created_at')
            ->addAttributeToSelect('total_qty')
            ->joinAttribute('shipping_firstname', 'order_address/firstname', 'shipping_address_id', null,
'left')
            ->joinAttribute('shipping_lastname', 'order_address/lastname', 'shipping_address_id', null, 'left')
            ->joinAttribute('order_increment_id', 'order/increment_id', 'order_id', null, 'left')
            ->joinAttribute('order_created_at', 'order/created_at', 'order_id', null, 'left');

        if (is_array($filters)) {
            try {
                foreach ($filters as $field => $value) {
                    if (isset($this->_attributesMap['shipment'][$field])) {
                        $field = $this->_attributesMap['shipment'][$field];
                    }

                    $collection->addFieldToFilter($field, $value);
                }
            } catch (Mage_Core_Exception $e) {
                $this->_fault('filters_invalid', $e->getMessage());
```

```
            }
        }

        $result = array();

        foreach ($collection as $shipment) {
            $result[] = $this->_getAttributes($shipment, 'shipment');
        }

        return $result;
    }
```

Method: info

```
$client->call($sessionId, 'sales_order_shipment.info', $shipmentIncrementId);
```

Implemented In

```
Mage_Sales_Model_Order_Shipment_Api::info
/Users/alanstorm/Sites2011/magento1point6point1.dev/app/code/core/Mage/Sales/Model/Order/Shipment/Api.php
```

Doc Comment

```
/**
 * Retrieve shipment information
 *
 * @param string $shipmentIncrementId
 * @return array
 */
```

Method Implementation Definition

```
    public function info($shipmentIncrementId)
    {
        $shipment = Mage::getModel('sales/order_shipment')->loadByIncrementId($shipmentIncrementId);

        /* @var $shipment Mage_Sales_Model_Order_Shipment */

        if (!$shipment->getId()) {
            $this->_fault('not_exists');
        }

        $result = $this->_getAttributes($shipment, 'shipment');

        $result['items'] = array();
        foreach ($shipment->getAllItems() as $item) {
            $result['items'][] = $this->_getAttributes($item, 'shipment_item');
        }

        $result['tracks'] = array();
        foreach ($shipment->getAllTracks() as $track) {
            $result['tracks'][] = $this->_getAttributes($track, 'shipment_track');
        }

        $result['comments'] = array();
        foreach ($shipment->getCommentsCollection() as $comment) {
            $result['comments'][] = $this->_getAttributes($comment, 'shipment_comment');
        }

        return $result;
    }
```

Method: create

```
$client->call($sessionId, 'sales_order_shipment.create', $orderIncrementId, $itemsQty, $comment, $email, $includeComment);
```

Implemented In

```
Mage_Sales_Model_Order_Shipment_Api::create
/Users/alanstorm/Sites2011/magento1point6point1.dev/app/code/core/Mage/Sales/Model/Order/Shipment/Api.php
```

Doc Comment

```
/**
 * Create new shipment for order
 *
 * @param string $orderIncrementId
 * @param array $itemsQty
 * @param string $comment
 * @param booleam $email
 * @param boolean $includeComment
 * @return string
 */
```

Method Implementation Definition

```
    public function create($orderIncrementId, $itemsQty = array(), $comment = null, $email = false, $includeComment = false)
    {
        $order = Mage::getModel('sales/order')->loadByIncrementId($orderIncrementId);

        /**
          * Check order existing
          */
        if (!$order->getId()) {
             $this->_fault('order_not_exists');
        }

        /**
         * Check shipment create availability
         */
        if (!$order->canShip()) {
             $this->_fault('data_invalid', Mage::helper('sales')->__('Cannot do shipment for order.'));
        }

         /* @var $shipment Mage_Sales_Model_Order_Shipment */
        $shipment = $order->prepareShipment($itemsQty);
        if ($shipment) {
            $shipment->register();
            $shipment->addComment($comment, $email && $includeComment);
            if ($email) {
                $shipment->setEmailSent(true);
            }
            $shipment->getOrder()->setIsInProcess(true);
            try {
                $transactionSave = Mage::getModel('core/resource_transaction')
                    ->addObject($shipment)
                    ->addObject($shipment->getOrder())
                    ->save();
                $shipment->sendEmail($email, ($includeComment ? $comment : ''));
            } catch (Mage_Core_Exception $e) {
                $this->_fault('data_invalid', $e->getMessage());
            }
            return $shipment->getIncrementId();
        }
        return null;
    }
```

Method: addComment

```
$client->call($sessionId, 'sales_order_shipment.addComment', $shipmentIncrementId, $comment, $email, $includeInEmail);
```

Implemented In

```
Mage_Sales_Model_Order_Shipment_Api::addComment
/Users/alanstorm/Sites2011/magento1point6point1.dev/app/code/core/Mage/Sales/Model/Order/Shipment/Api.php
```

Doc Comment

```
/**
 * Add comment to shipment
 *
 * @param string $shipmentIncrementId
 * @param string $comment
 * @param boolean $email
 * @param boolean $includeInEmail
 * @return boolean
 */
```

Method Implementation Definition

```
    public function addComment($shipmentIncrementId, $comment, $email = false, $includeInEmail = false)
    {
        $shipment = Mage::getModel('sales/order_shipment')->loadByIncrementId($shipmentIncrementId);

        /* @var $shipment Mage_Sales_Model_Order_Shipment */

        if (!$shipment->getId()) {
            $this->_fault('not_exists');
        }


        try {
            $shipment->addComment($comment, $email);
            $shipment->sendUpdateEmail($email, ($includeInEmail ? $comment : ''));
            $shipment->save();
        } catch (Mage_Core_Exception $e) {
            $this->_fault('data_invalid', $e->getMessage());
        }

        return true;
    }
```

Method: addTrack

```
$client->call($sessionId, 'sales_order_shipment.addTrack', $shipmentIncrementId, $carrier, $title, $trackNumber);
```

Implemented In

```
Mage_Sales_Model_Order_Shipment_Api::addTrack
/Users/alanstorm/Sites2011/magento1point6point1.dev/app/code/core/Mage/Sales/Model/Order/Shipment/Api.php
```

Doc Comment

```
/**
 * Add tracking number to order
 *
 * @param string $shipmentIncrementId
 * @param string $carrier
 * @param string $title
 * @param string $trackNumber
 * @return int
 */
```

Method Implementation Definition

```
    public function addTrack($shipmentIncrementId, $carrier, $title, $trackNumber)
    {
        $shipment = Mage::getModel('sales/order_shipment')->loadByIncrementId($shipmentIncrementId);

        /* @var $shipment Mage_Sales_Model_Order_Shipment */

        if (!$shipment->getId()) {
            $this->_fault('not_exists');
        }

        $carriers = $this->_getCarriers($shipment);

        if (!isset($carriers[$carrier])) {
            $this->_fault('data_invalid', Mage::helper('sales')->__('Invalid carrier specified.'));
        }

        $track = Mage::getModel('sales/order_shipment_track')
                    ->setNumber($trackNumber)
                    ->setCarrierCode($carrier)
                    ->setTitle($title);

        $shipment->addTrack($track);

        try {
            $shipment->save();
            $track->save();
        } catch (Mage_Core_Exception $e) {
            $this->_fault('data_invalid', $e->getMessage());
        }

        return $track->getId();
    }
```

Method: removeTrack

```
$client->call($sessionId, 'sales_order_shipment.removeTrack', $shipmentIncrementId, $trackId);
```

Implemented In

```
Mage_Sales_Model_Order_Shipment_Api::removeTrack
/Users/alanstorm/Sites2011/magento1point6point1.dev/app/code/core/Mage/Sales/Model/Order/Shipment/Api.php
```

Doc Comment

```
/**
 * Remove tracking number
 *
 * @param string $shipmentIncrementId
 * @param int $trackId
 * @return boolean
 */
```

Method Implementation Definition

```
    public function removeTrack($shipmentIncrementId, $trackId)
    {
        $shipment = Mage::getModel('sales/order_shipment')->loadByIncrementId($shipmentIncrementId);

        /* @var $shipment Mage_Sales_Model_Order_Shipment */

        if (!$shipment->getId()) {
            $this->_fault('not_exists');
        }
```

```
        if(!$track = $shipment->getTrackById($trackId)) {
            $this->_fault('track_not_exists');
        }

        try {
            $track->delete();
        } catch (Mage_Core_Exception $e) {
            $this->_fault('track_not_deleted', $e->getMessage());
        }

        return true;
    }
```

Method: getCarriers

```
$client->call($sessionId, 'sales_order_shipment.getCarriers', $orderIncrementId);
```

Implemented In

```
Mage_Sales_Model_Order_Shipment_Api::getCarriers
/Users/alanstorm/Sites2011/magento1point6point1.dev/app/code/core/Mage/Sales/Model/Order/Shipment/Api.php
```

Doc Comment

```
/**
 * Retrieve allowed shipping carriers for specified order
 *
 * @param string $orderIncrementId
 * @return array
 */
```

Method Implementation Definition

```
    public function getCarriers($orderIncrementId)
    {
        $order = Mage::getModel('sales/order')->loadByIncrementId($orderIncrementId);

        /**
          * Check order existing
          */
        if (!$order->getId()) {
            $this->_fault('order_not_exists');
        }

        return $this->_getCarriers($order);
    }
```

Resource: sales_order_invoice

Method: list

```
$client->call($sessionId, 'sales_order_invoice.list', $filters);
```

Implemented In

```
Mage_Sales_Model_Order_Invoice_Api::items
/Users/alanstorm/Sites2011/magento1point6point1.dev/app/code/core/Mage/Sales/Model/Order/Invoice/Api.php
```

Doc Comment

```
/**
 * Retrive invoices by filters
 *
 * @param array $filters
 * @return array
 */
```

Method Implementation Definition

```
    public function items($filters = null)
    {
        //TODO: add full name logic
        $collection = Mage::getResourceModel('sales/order_invoice_collection')
            ->addAttributeToSelect('order_id')
            ->addAttributeToSelect('increment_id')
            ->addAttributeToSelect('created_at')
            ->addAttributeToSelect('state')
            ->addAttributeToSelect('grand_total')
            ->addAttributeToSelect('order_currency_code')
            ->joinAttribute('billing_firstname', 'order_address/firstname', 'billing_address_id', null, 'left')
            ->joinAttribute('billing_lastname', 'order_address/lastname', 'billing_address_id', null, 'left')
            ->joinAttribute('order_increment_id', 'order/increment_id', 'order_id', null, 'left')
            ->joinAttribute('order_created_at', 'order/created_at', 'order_id', null, 'left');

        if (is_array($filters)) {
            try {
                foreach ($filters as $field => $value) {
                    if (isset($this->_attributesMap['invoice'][$field])) {
                        $field = $this->_attributesMap['invoice'][$field];
                    }

                    $collection->addFieldToFilter($field, $value);
                }
            } catch (Mage_Core_Exception $e) {
                $this->_fault('filters_invalid', $e->getMessage());
            }
        }

        $result = array();

        foreach ($collection as $invoice) {
            $result[] = $this->_getAttributes($invoice, 'invoice');
        }

        return $result;
    }
```

Method: info

```
$client->call($sessionId, 'sales_order_invoice.info', $invoiceIncrementId);
```

Implemented In

```
Mage_Sales_Model_Order_Invoice_Api::info
/Users/alanstorm/Sites2011/magento1point6point1.dev/app/code/core/Mage/Sales/Model/Order/Invoice/Api.php
```

Doc Comment

```
/**
 * Retrieve invoice information
 *
 * @param string $invoiceIncrementId
 * @return array
 */
```

Method Implementation Definition

```
    public function info($invoiceIncrementId)
    {
        $invoice = Mage::getModel('sales/order_invoice')->loadByIncrementId($invoiceIncrementId);

        /* @var Mage_Sales_Model_Order_Invoice $invoice */
```

```
        if (!$invoice->getId()) {
            $this->_fault('not_exists');
        }

        $result = $this->_getAttributes($invoice, 'invoice');
        $result['order_increment_id'] = $invoice->getOrderIncrementId();

        $result['items'] = array();
        foreach ($invoice->getAllItems() as $item) {
            $result['items'][] = $this->_getAttributes($item, 'invoice_item');
        }

        $result['comments'] = array();
        foreach ($invoice->getCommentsCollection() as $comment) {
            $result['comments'][] = $this->_getAttributes($comment, 'invoice_comment');
        }

        return $result;
    }
```

Method: create

```
$client->call($sessionId, 'sales_order_invoice.create', $orderIncrementId, $itemsQty, $comment, $email,
$includeComment);
```

Implemented In

```
Mage_Sales_Model_Order_Invoice_Api::create
/Users/alanstorm/Sites2011/magento1point6point1.dev/app/code/core/Mage/Sales/Model/Order/Invoice/Api.php
```

Doc Comment

```
/**
 * Create new invoice for order
 *
 * @param string $orderIncrementId
 * @param array $itemsQty
 * @param string $comment
 * @param booleam $email
 * @param boolean $includeComment
 * @return string
 */
```

Method Implementation Definition

```
    public function create($orderIncrementId, $itemsQty, $comment = null, $email = false, $includeComment =
false)
    {
        $order = Mage::getModel('sales/order')->loadByIncrementId($orderIncrementId);

        /* @var $order Mage_Sales_Model_Order */
        /**
          * Check order existing
          */
        if (!$order->getId()) {
             $this->_fault('order_not_exists');
        }

        /**
         * Check invoice create availability
         */
        if (!$order->canInvoice()) {
             $this->_fault('data_invalid', Mage::helper('sales')->__('Cannot do invoice for order.'));
        }
```

```
        $invoice = $order->prepareInvoice($itemsQty);

        $invoice->register();

        if ($comment !== null) {
            $invoice->addComment($comment, $email);
        }

        if ($email) {
            $invoice->setEmailSent(true);
        }

        $invoice->getOrder()->setIsInProcess(true);

        try {
            $transactionSave = Mage::getModel('core/resource_transaction')
                ->addObject($invoice)
                ->addObject($invoice->getOrder())
                ->save();

            $invoice->sendEmail($email, ($includeComment ? $comment : ''));
        } catch (Mage_Core_Exception $e) {
            $this->_fault('data_invalid', $e->getMessage());
        }

        return $invoice->getIncrementId();
    }
```

Method: addComment

```
$client->call($sessionId, 'sales_order_invoice.addComment', $invoiceIncrementId, $comment, $email,
$includeComment);
```

Implemented In

```
Mage_Sales_Model_Order_Invoice_Api::addComment
/Users/alanstorm/Sites2011/magento1point6point1.dev/app/code/core/Mage/Sales/Model/Order/Invoice/Api.php
```

Doc Comment

```
/**
 * Add comment to invoice
 *
 * @param string $invoiceIncrementId
 * @param string $comment
 * @param boolean $email
 * @param boolean $includeComment
 * @return boolean
 */
```

Method Implementation Definition

```
    public function addComment($invoiceIncrementId, $comment, $email = false, $includeComment = false)
    {
        $invoice = Mage::getModel('sales/order_invoice')->loadByIncrementId($invoiceIncrementId);

        /* @var $invoice Mage_Sales_Model_Order_Invoice */

        if (!$invoice->getId()) {
            $this->_fault('not_exists');
        }


        try {
```

```
            $invoice->addComment($comment, $email);
            $invoice->sendUpdateEmail($email, ($includeComment ? $comment : ''));
            $invoice->save();
        } catch (Mage_Core_Exception $e) {
            $this->_fault('data_invalid', $e->getMessage());
        }

        return true;
    }
```

Method: capture

```
$client->call($sessionId, 'sales_order_invoice.capture', $invoiceIncrementId);
```

Implemented In

```
Mage_Sales_Model_Order_Invoice_Api::capture
/Users/alanstorm/Sites2011/magento1point6point1.dev/app/code/core/Mage/Sales/Model/Order/Invoice/Api.php
```

Doc Comment

```
/**
 * Capture invoice
 *
 * @param string $invoiceIncrementId
 * @return boolean
 */
```

Method Implementation Definition

```
    public function capture($invoiceIncrementId)
    {
        $invoice = Mage::getModel('sales/order_invoice')->loadByIncrementId($invoiceIncrementId);

        /* @var $invoice Mage_Sales_Model_Order_Invoice */

        if (!$invoice->getId()) {
            $this->_fault('not_exists');
        }

        if (!$invoice->canCapture()) {
            $this->_fault('status_not_changed', Mage::helper('sales')->__('Invoice cannot be captured.'));
        }

        try {
            $invoice->capture();
            $invoice->getOrder()->setIsInProcess(true);
            $transactionSave = Mage::getModel('core/resource_transaction')
                ->addObject($invoice)
                ->addObject($invoice->getOrder())
                ->save();
        } catch (Mage_Core_Exception $e) {
            $this->_fault('status_not_changed', $e->getMessage());
        } catch (Exception $e) {
            $this->_fault('status_not_changed', Mage::helper('sales')->__('Invoice capturing problem.'));
        }

        return true;
    }
```

Method: void

```
$client->call($sessionId, 'sales_order_invoice.void', $invoiceIncrementId);
```

Implemented In

Mage_Sales_Model_Order_Invoice_Api::void
/Users/alanstorm/Sites2011/magento1point6point1.dev/app/code/core/Mage/Sales/Model/Order/Invoice/Api.php

Doc Comment

```
/**
 * Void invoice
 *
 * @param unknown_type $invoiceIncrementId
 * @return unknown
 */
```

Method Implementation Definition

```
    public function void($invoiceIncrementId)
    {
        $invoice = Mage::getModel('sales/order_invoice')->loadByIncrementId($invoiceIncrementId);

        /* @var $invoice Mage_Sales_Model_Order_Invoice */

        if (!$invoice->getId()) {
            $this->_fault('not_exists');
        }

        if (!$invoice->canVoid()) {
            $this->_fault('status_not_changed', Mage::helper('sales')->__('Invoice cannot be voided.'));
        }

        try {
            $invoice->void();
            $invoice->getOrder()->setIsInProcess(true);
            $transactionSave = Mage::getModel('core/resource_transaction')
                ->addObject($invoice)
                ->addObject($invoice->getOrder())
                ->save();
        } catch (Mage_Core_Exception $e) {
            $this->_fault('status_not_changed', $e->getMessage());
        } catch (Exception $e) {
            $this->_fault('status_not_changed', Mage::helper('sales')->__('Invoice void problem'));
        }

        return true;
    }
```

Method: cancel

```
$client->call($sessionId, 'sales_order_invoice.cancel', $invoiceIncrementId);
```

Implemented In

Mage_Sales_Model_Order_Invoice_Api::cancel
/Users/alanstorm/Sites2011/magento1point6point1.dev/app/code/core/Mage/Sales/Model/Order/Invoice/Api.php

Doc Comment

```
/**
 * Cancel invoice
 *
 * @param string $invoiceIncrementId
 * @return boolean
 */
```

Method Implementation Definition

```
    public function cancel($invoiceIncrementId)
    {
        $invoice = Mage::getModel('sales/order_invoice')->loadByIncrementId($invoiceIncrementId);
```

```
        /* @var $invoice Mage_Sales_Model_Order_Invoice */

        if (!$invoice->getId()) {
            $this->_fault('not_exists');
        }

        if (!$invoice->canCancel()) {
            $this->_fault('status_not_changed', Mage::helper('sales')->__('Invoice cannot be canceled.'));
        }

        try {
            $invoice->cancel();
            $invoice->getOrder()->setIsInProcess(true);
            $transactionSave = Mage::getModel('core/resource_transaction')
                ->addObject($invoice)
                ->addObject($invoice->getOrder())
                ->save();
        } catch (Mage_Core_Exception $e) {
            $this->_fault('status_not_changed', $e->getMessage());
        } catch (Exception $e) {
            $this->_fault('status_not_changed', Mage::helper('sales')->__('Invoice canceling problem.'));
        }

        return true;
    }
```

Resource: sales_order_creditmemo

Method: list

```
$client->call($sessionId, 'sales_order_creditmemo.list', $filter);
```

Implemented In

```
Mage_Sales_Model_Order_Creditmemo_Api::items
/Users/alanstorm/Sites2011/magento1point6point1.dev/app/code/core/Mage/Sales/Model/Order/Creditmemo/Api.php
```

Doc Comment

```
/**
 * Retrieve credit memos by filters
 *
 * @param array|null $filter
 * @return array
 */
```

Method Implementation Definition

```
    public function items($filter = null)
    {
        $filter = $this->_prepareListFilter($filter);
        try {
            $result = array();
            /** @var $creditmemoModel Mage_Sales_Model_Order_Creditmemo */
            $creditmemoModel = Mage::getModel('sales/order_creditmemo');
            // map field name entity_id to creditmemo_id
            foreach ($creditmemoModel->getFilteredCollectionItems($filter) as $creditmemo) {
                $result[] = $this->_getAttributes($creditmemo, 'creditmemo');
            }
        } catch (Exception $e) {
            $this->_fault('invalid_filter', $e->getMessage());
        }
        return $result;
    }
```

Method: info

```
$client->call($sessionId, 'sales_order_creditmemo.info', $creditmemoIncrementId);
```

Implemented In

```
Mage_Sales_Model_Order_Creditmemo_Api::info
/Users/alanstorm/Sites2011/magento1point6point1.dev/app/code/core/Mage/Sales/Model/Order/Creditmemo/Api.php
```

Doc Comment

```
/**
 * Retrieve credit memo information
 *
 * @param string $creditmemoIncrementId
 * @return array
 */
```

Method Implementation Definition

```
    public function info($creditmemoIncrementId)
    {
        $creditmemo = $this->_getCreditmemo($creditmemoIncrementId);
        // get credit memo attributes with entity_id' => 'creditmemo_id' mapping
        $result = $this->_getAttributes($creditmemo, 'creditmemo');
        $result['order_increment_id'] = $creditmemo->getOrder()->load($creditmemo->getOrderId())->getIncrementId
();
        // items refunded
        $result['items'] = array();
        foreach ($creditmemo->getAllItems() as $item) {
            $result['items'][] = $this->_getAttributes($item, 'creditmemo_item');
        }
        // credit memo comments
        $result['comments'] = array();
        foreach ($creditmemo->getCommentsCollection() as $comment) {
            $result['comments'][] = $this->_getAttributes($comment, 'creditmemo_comment');
        }
        return $result;
    }
```

Method: create

```
$client->call($sessionId, 'sales_order_creditmemo.create', $orderIncrementId, $data, $comment, $notifyCustomer,
$includeComment, $refundToStoreCreditAmount);
```

Implemented In

```
Mage_Sales_Model_Order_Creditmemo_Api::create
/Users/alanstorm/Sites2011/magento1point6point1.dev/app/code/core/Mage/Sales/Model/Order/Creditmemo/Api.php
```

Doc Comment

```
/**
 * Create new credit memo for order
 *
 * @param string $orderIncrementId
 * @param array $data array('qtys' => array('sku1' => qty1, ... , 'skuN' => qtyN),
 *      'shipping_amount' => value, 'adjustment_positive' => value, 'adjustment_negative' => value)
 * @param string|null $comment
 * @param bool $notifyCustomer
 * @param bool $includeComment
 * @param string $refundToStoreCreditAmount
 * @return string $creditmemoIncrementId
 */
```

Method Implementation Definition

```
    public function create($orderIncrementId, $data = null, $comment = null, $notifyCustomer = false,
        $includeComment = false, $refundToStoreCreditAmount = null)
```

```
    {
        /** @var $order Mage_Sales_Model_Order */
        $order = Mage::getModel('sales/order')->load($orderIncrementId, 'increment_id');
        if (!$order->getId()) {
            $this->_fault('order_not_exists');
        }
        if (!$order->canCreditmemo()) {
            $this->_fault('cannot_create_creditmemo');
        }
        $data = $this->_prepareCreateData($data);

        /** @var $service Mage_Sales_Model_Service_Order */
        $service = Mage::getModel('sales/service_order', $order);
        /** @var $creditmemo Mage_Sales_Model_Order_Creditmemo */
        $creditmemo = $service->prepareCreditmemo($data);

        // refund to Store Credit
        if ($refundToStoreCreditAmount) {
            // check if refund to Store Credit is available
            if ($order->getCustomerIsGuest()) {
                $this->_fault('cannot_refund_to_storecredit');
            }
            $refundToStoreCreditAmount = max(
                0,
                min($creditmemo->getBaseCustomerBalanceReturnMax(), $refundToStoreCreditAmount)
            );
            if ($refundToStoreCreditAmount) {
                $refundToStoreCreditAmount = $creditmemo->getStore()->roundPrice($refundToStoreCreditAmount);
                $creditmemo->setBaseCustomerBalanceTotalRefunded($refundToStoreCreditAmount);
                $refundToStoreCreditAmount = $creditmemo->getStore()->roundPrice(
                    $refundToStoreCreditAmount*$order->getStoreToOrderRate()
                );
                // this field can be used by customer balance observer
                $creditmemo->setBsCustomerBalTotalRefunded($refundToStoreCreditAmount);
                // setting flag to make actual refund to customer balance after credit memo save
                $creditmemo->setCustomerBalanceRefundFlag(true);
            }
        }
        $creditmemo->setPaymentRefundDisallowed(true)->register();
        // add comment to creditmemo
        if (!empty($comment)) {
            $creditmemo->addComment($comment, $notifyCustomer);
        }
        try {
            Mage::getModel('core/resource_transaction')
                ->addObject($creditmemo)
                ->addObject($order)
                ->save();
            // send email notification
            $creditmemo->sendEmail($notifyCustomer, ($includeComment ? $comment : ''));
        } catch (Mage_Core_Exception $e) {
            $this->_fault('data_invalid', $e->getMessage());
        }
        return $creditmemo->getIncrementId();
    }
```

Method: addComment

```
$client->call($sessionId, 'sales_order_creditmemo.addComment', $creditmemoIncrementId, $comment,
$notifyCustomer, $includeComment);
```

Implemented In

Mage_Sales_Model_Order_Creditmemo_Api::addComment
/Users/alanstorm/Sites2011/magento1point6point1.dev/app/code/core/Mage/Sales/Model/Order/Creditmemo/Api.php

Doc Comment

```
/**
 * Add comment to credit memo
 *
 * @param string $creditmemoIncrementId
 * @param string $comment
 * @param boolean $notifyCustomer
 * @param boolean $includeComment
 * @return boolean
 */
```

Method Implementation Definition

```
    public function addComment($creditmemoIncrementId, $comment, $notifyCustomer = false, $includeComment =
false)
    {
        $creditmemo = $this->_getCreditmemo($creditmemoIncrementId);
        try {
            $creditmemo->addComment($comment, $notifyCustomer)->save();
            $creditmemo->sendUpdateEmail($notifyCustomer, ($includeComment ? $comment : ''));
        } catch (Mage_Core_Exception $e) {
            $this->_fault('data_invalid', $e->getMessage());
        }

        return true;
    }
```

Method: cancel

$client->call($sessionId, 'sales_order_creditmemo.cancel', $creditmemoIncrementId);

Implemented In

Mage_Sales_Model_Order_Creditmemo_Api::cancel
/Users/alanstorm/Sites2011/magento1point6point1.dev/app/code/core/Mage/Sales/Model/Order/Creditmemo/Api.php

Doc Comment

```
/**
 * Cancel credit memo
 *
 * @param string $creditmemoIncrementId
 * @return boolean
 */
```

Method Implementation Definition

```
    public function cancel($creditmemoIncrementId)
    {
        $creditmemo = $this->_getCreditmemo($creditmemoIncrementId);

        if (!$creditmemo->canCancel()) {
            $this->_fault('status_not_changed', Mage::helper('sales')->__('Credit memo cannot be canceled.'));
        }
        try {
            $creditmemo->cancel()->save();
        } catch (Exception $e) {
            $this->_fault('status_not_changed', Mage::helper('sales')->__('Credit memo canceling problem.'));
        }

        return true;
    }
```

Resource: cataloginventory_stock_item

Method: list

```
$client->call($sessionId, 'cataloginventory_stock_item.list', $productIds);
```

Implemented In

```
Mage_CatalogInventory_Model_Stock_Item_Api::items
/Users/alanstorm/Sites2011/magento1point6point1.dev/app/code/core/Mage/CatalogInventory/Model/Stock/Item/Api.php
```

Doc Comment

Method Implementation Definition

```
    public function items($productIds)
    {
        if (!is_array($productIds)) {
            $productIds = array($productIds);
        }

        $product = Mage::getModel('catalog/product');

        foreach ($productIds as &$productId) {
            if ($newId = $product->getIdBySku($productId)) {
                $productId = $newId;
            }
        }

        $collection = Mage::getModel('catalog/product')
            ->getCollection()
            ->setFlag('require_stock_items', true)
            ->addFieldToFilter('entity_id', array('in'=>$productIds));

        $result = array();

        foreach ($collection as $product) {
            if ($product->getStockItem()) {
                $result[] = array(
                    'product_id'    => $product->getId(),
                    'sku'           => $product->getSku(),
                    'qty'           => $product->getStockItem()->getQty(),
                    'is_in_stock'   => $product->getStockItem()->getIsInStock()
                );
            }
        }

        return $result;
    }
```

Method: update

```
$client->call($sessionId, 'cataloginventory_stock_item.update', $productId, $data);
```

Implemented In

```
Mage_CatalogInventory_Model_Stock_Item_Api::update
/Users/alanstorm/Sites2011/magento1point6point1.dev/app/code/core/Mage/CatalogInventory/Model/Stock/Item/Api.php
```

Doc Comment

Method Implementation Definition

```
public function update($productId, $data)
{
    $product = Mage::getModel('catalog/product');

    if ($newId = $product->getIdBySku($productId)) {
        $productId = $newId;
    }

    $product->setStoreId($this->_getStoreId())
        ->load($productId);

    if (!$product->getId()) {
        $this->_fault('not_exists');
    }

    if (!$stockData = $product->getStockData()) {
        $stockData = array();
    }

    if (isset($data['qty'])) {
        $stockData['qty'] = $data['qty'];
    }

    if (isset($data['is_in_stock'])) {
        $stockData['is_in_stock'] = $data['is_in_stock'];
    }

    if (isset($data['manage_stock'])) {
        $stockData['manage_stock'] = $data['manage_stock'];
    }

    if (isset($data['use_config_manage_stock'])) {
        $stockData['use_config_manage_stock'] = $data['use_config_manage_stock'];
    }

    $product->setStockData($stockData);

    try {
        $product->save();
    } catch (Mage_Core_Exception $e) {
        $this->_fault('not_updated', $e->getMessage());
    }

    return true;
}
```

Resource: cart

Method: create

```
$client->call($sessionId, 'cart.create', $store);
```

Implemented In

```
Mage_Checkout_Model_Cart_Api::create
/Users/alanstorm/Sites2011/magento1point6point1.dev/app/code/core/Mage/Checkout/Model/Cart/Api.php
```

Doc Comment

```
/**
 * Create new quote for shopping cart
 *
 * @param int|string $store
```

```
 * @return int
 */
```

Method Implementation Definition

```
    public function create($store = null)
    {
        $storeId = $this->_getStoreId($store);

        try {
            /*@var $quote Mage_Sales_Model_Quote*/
            $quote = Mage::getModel('sales/quote');
            $quote->setStoreId($storeId)
                    ->setIsActive(false)
                    ->setIsMultiShipping(false)
                    ->save();
        } catch (Mage_Core_Exception $e) {
            $this->_fault('create_quote_fault', $e->getMessage());
        }
        return (int) $quote->getId();
    }
```

Method: order

```
$client->call($sessionId, 'cart.order', $quoteId, $store, $agreements);
```

Implemented In

```
Mage_Checkout_Model_Cart_Api::createOrder
/Users/alanstorm/Sites2011/magento1point6point1.dev/app/code/core/Mage/Checkout/Model/Cart/Api.php
```

Doc Comment

```
/**
 * Create an order from the shopping cart (quote)
 *
 * @param  $quoteId
 * @param  $store
 * @param  $agreements array
 * @return string
 */
```

Method Implementation Definition

```
    public function createOrder($quoteId, $store = null, $agreements = null)
    {
        $requiredAgreements = Mage::helper('checkout')->getRequiredAgreementIds();
        if (!empty($requiredAgreements)) {
            $diff = array_diff($agreements, $requiredAgreements);
            if (!empty($diff)) {
                $this->_fault('required_agreements_are_not_all');
            }
        }

        $quote = $this->_getQuote($quoteId, $store);
        if ($quote->getIsMultiShipping()) {
            $this->_fault('invalid_checkout_type');
        }
        if ($quote->getCheckoutMethod() == Mage_Checkout_Model_Api_Resource_Customer::MODE_GUEST
                && !Mage::helper('checkout')->isAllowedGuestCheckout($quote, $quote->getStoreId())) {
            $this->_fault('guest_checkout_is_not_enabled');
        }

        /** @var $customerResource Mage_Checkout_Model_Api_Resource_Customer */
        $customerResource = Mage::getModel("checkout/api_resource_customer");
        $isNewCustomer = $customerResource->prepareCustomerForQuote($quote);
```

```
        try {
            $quote->collectTotals();
            /** @var $service Mage_Sales_Model_Service_Quote */
            $service = Mage::getModel('sales/service_quote', $quote);
            $service->submitAll();

            if ($isNewCustomer) {
                try {
                    $customerResource->involveNewCustomer($quote);
                } catch (Exception $e) {
                    Mage::logException($e);
                }
            }

            $order = $service->getOrder();
            if ($order) {
                Mage::dispatchEvent('checkout_type_onepage_save_order_after',
                    array('order' => $order, 'quote' => $quote));

                try {
                    $order->sendNewOrderEmail();
                } catch (Exception $e) {
                    Mage::logException($e);
                }
            }

            Mage::dispatchEvent(
                'checkout_submit_all_after',
                array('order' => $order, 'quote' => $quote)
            );
        } catch (Mage_Core_Exception $e) {
            $this->_fault('create_order_fault', $e->getMessage());
        }

        return $order->getIncrementId();
    }
```

Method: info

```
$client->call($sessionId, 'cart.info', $quoteId, $store);
```

Implemented In

```
Mage_Checkout_Model_Cart_Api::info
/Users/alanstorm/Sites2011/magento1point6point1.dev/app/code/core/Mage/Checkout/Model/Cart/Api.php
```

Doc Comment

```
/**
 * Retrieve full information about quote
 *
 * @param  $quoteId
 * @param  $store
 * @return array
 */
```

Method Implementation Definition

```
    public function info($quoteId, $store = null)
    {
        $quote = $this->_getQuote($quoteId, $store);

        if ($quote->getGiftMessageId() > 0) {
            $quote->setGiftMessage(
```

```
                Mage::getSingleton('giftmessage/message')->load($quote->getGiftMessageId())->getMessage()
            );
        }

        $result = $this->_getAttributes($quote, 'quote');
        $result['shipping_address'] = $this->_getAttributes($quote->getShippingAddress(), 'quote_address');
        $result['billing_address'] = $this->_getAttributes($quote->getBillingAddress(), 'quote_address');
        $result['items'] = array();

        foreach ($quote->getAllItems() as $item) {
            if ($item->getGiftMessageId() > 0) {
                $item->setGiftMessage(
                    Mage::getSingleton('giftmessage/message')->load($item->getGiftMessageId())->getMessage()
                );
            }

            $result['items'][] = $this->_getAttributes($item, 'quote_item');
        }

        $result['payment'] = $this->_getAttributes($quote->getPayment(), 'quote_payment');

        return $result;
    }
```

Method: totals

```
$client->call($sessionId, 'cart.totals', $quoteId, $store);
```

Implemented In

```
Mage_Checkout_Model_Cart_Api::totals
/Users/alanstorm/Sites2011/magento1point6point1.dev/app/code/core/Mage/Checkout/Model/Cart/Api.php
```

Doc Comment

```
/**
 * @param  $quoteId
 * @param  $store
 * @return void
 */
```

Method Implementation Definition

```
    public function totals($quoteId, $store = null)
    {
        $quote = $this->_getQuote($quoteId, $store);

        $totals = $quote->getTotals();

        $totalsResult = array();
        foreach ($totals as $total) {
            $totalsResult[] = array(
                "title" => $total->getTitle(),
                "amount" => $total->getValue()
            );
        }
        return $totalsResult;
    }
```

Method: license

```
$client->call($sessionId, 'cart.license', $quoteId, $store);
```

Implemented In

```
Mage_Checkout_Model_Cart_Api::licenseAgreement
```

/Users/alanstorm/Sites2011/magento1point6point1.dev/app/code/core/Mage/Checkout/Model/Cart/Api.php

Doc Comment

```
/**
 * @param  $quoteId
 * @param  $store
 * @return array
 */
```

Method Implementation Definition

```
    public function licenseAgreement($quoteId, $store = null)
    {
        $quote = $this->_getQuote($quoteId, $store);
        $storeId = $quote->getStoreId();

        $agreements = array();
        if (Mage::getStoreConfigFlag('checkout/options/enable_agreements')) {
            $agreementsCollection = Mage::getModel('checkout/agreement')->getCollection()
                    ->addStoreFilter($storeId)
                    ->addFieldToFilter('is_active', 1);

            foreach ($agreementsCollection as $_a) {
                /** @var $_a  Mage_Checkout_Model_Agreement */
                $agreements[] = $this->_getAttributes($_a, "quote_agreement");
            }
        }

        return $agreements;
    }
```

Resource: cart_product

Method: add

```
$client->call($sessionId, 'cart_product.add', $quoteId, $productsData, $store);
```

Implemented In

Mage_Checkout_Model_Cart_Product_Api::add
/Users/alanstorm/Sites2011/magento1point6point1.dev/app/code/core/Mage/Checkout/Model/Cart/Product/Api.php

Doc Comment

```
/**
 * @param  $quoteId
 * @param  $productsData
 * @param  $store
 * @return bool
 */
```

Method Implementation Definition

```
    public function add($quoteId, $productsData, $store=null)
    {
        $quote = $this->_getQuote($quoteId, $store);
        if (empty($store)) {
            $store = $quote->getStoreId();
        }

        $productsData = $this->_prepareProductsData($productsData);
        if (empty($productsData)) {
            $this->_fault('invalid_product_data');
        }

        $errors = array();
        foreach ($productsData as $productItem) {
```

```
            if (isset($productItem['product_id'])) {
                $productByItem = $this->_getProduct($productItem['product_id'], $store, "id");
            } else if (isset($productItem['sku'])) {
                $productByItem = $this->_getProduct($productItem['sku'], $store, "sku");
            } else {
                $errors[] = Mage::helper('checkout')->__("One item of products do not have identifier or sku");
                continue;
            }

            $productRequest = $this->_getProductRequest($productItem);
            try {
                $result = $quote->addProduct($productByItem, $productRequest);
                if (is_string($result)) {
                    Mage::throwException($result);
                }
            } catch (Mage_Core_Exception $e) {
                $errors[] = $e->getMessage();
            }
        }

        if (!empty($errors)) {
            $this->_fault("add_product_fault", implode(PHP_EOL, $errors));
        }

        try {
            $quote->collectTotals()->save();
        } catch(Exception $e) {
            $this->_fault("add_product_quote_save_fault", $e->getMessage());
        }

        return true;
    }
```

Method: update

```
$client->call($sessionId, 'cart_product.update', $quoteId, $productsData, $store);
```

Implemented In

```
Mage_Checkout_Model_Cart_Product_Api::update
/Users/alanstorm/Sites2011/magento1point6point1.dev/app/code/core/Mage/Checkout/Model/Cart/Product/Api.php
```

Doc Comment

```
/**
 * @param  $quoteId
 * @param  $productsData
 * @param  $store
 * @return bool
 */
```

Method Implementation Definition

```
    public function update($quoteId, $productsData, $store=null)
    {
        $quote = $this->_getQuote($quoteId, $store);
        if (empty($store)) {
            $store = $quote->getStoreId();
        }

        $productsData = $this->_prepareProductsData($productsData);
        if (empty($productsData)) {
            $this->_fault('invalid_product_data');
        }
```

```
        $errors = array();
        foreach ($productsData as $productItem) {
            if (isset($productItem['product_id'])) {
                $productByItem = $this->_getProduct($productItem['product_id'], $store, "id");
            } else if (isset($productItem['sku'])) {
                $productByItem = $this->_getProduct($productItem['sku'], $store, "sku");
            } else {
                $errors[] = Mage::helper('checkout')->__("One item of products do not have identifier or sku");
                continue;
            }

            /** @var $quoteItem Mage_Sales_Model_Quote_Item */
            $quoteItem = $this->_getQuoteItemByProduct($quote, $productByItem, $this->_getProductRequest
($productItem));
            if (is_null($quoteItem->getId())) {
                $errors[] = Mage::helper('checkout')->__("One item of products is not belong any of quote
item");
                continue;
            }

            if ($productItem['qty'] > 0) {
                $quoteItem->setQty($productItem['qty']);
            }
        }

        if (!empty($errors)) {
            $this->_fault("update_product_fault", implode(PHP_EOL, $errors));
        }

        try {
            $quote->save();
        } catch(Exception $e) {
            $this->_fault("update_product_quote_save_fault", $e->getMessage());
        }

        return true;
    }
```

Method: remove

```
$client->call($sessionId, 'cart_product.remove', $quoteId, $productsData, $store);
```

Implemented In

```
Mage_Checkout_Model_Cart_Product_Api::remove
/Users/alanstorm/Sites2011/magento1point6point1.dev/app/code/core/Mage/Checkout/Model/Cart/Product/Api.php
```

Doc Comment

```
/**
 * @param  $quoteId
 * @param  $productsData
 * @param  $store
 * @return bool
 */
```

Method Implementation Definition

```
    public function remove($quoteId, $productsData, $store=null)
    {
        $quote = $this->_getQuote($quoteId, $store);
        if (empty($store)) {
            $store = $quote->getStoreId();
        }
```

```
        $productsData = $this->_prepareProductsData($productsData);
        if (empty($productsData)) {
            $this->_fault('invalid_product_data');
        }

        $errors = array();
        foreach ($productsData as $productItem) {
            if (isset($productItem['product_id'])) {
                $productByItem = $this->_getProduct($productItem['product_id'], $store, "id");
            } else if (isset($productItem['sku'])) {
                $productByItem = $this->_getProduct($productItem['sku'], $store, "sku");
            } else {
                $errors[] = Mage::helper('checkout')->__("One item of products do not have identifier or sku");
                continue;
            }

            try {
                /** @var $quoteItem Mage_Sales_Model_Quote_Item */
                $quoteItem = $this->_getQuoteItemByProduct($quote, $productByItem, $this->_getProductRequest
($productItem));
                if (is_null($quoteItem->getId())) {
                    $errors[] = Mage::helper('checkout')->__("One item of products is not belong any of quote
item");
                    continue;
                }
                $quote->removeItem($quoteItem->getId());
            } catch (Mage_Core_Exception $e) {
                $errors[] = $e->getMessage();
            }
        }

        if (!empty($errors)) {
            $this->_fault("remove_product_fault", implode(PHP_EOL, $errors));
        }

        try {
            $quote->save();
        } catch(Exception $e) {
            $this->_fault("remove_product_quote_save_fault", $e->getMessage());
        }

        return true;
    }
```

Method: list

```
$client->call($sessionId, 'cart_product.list', $quoteId, $store);
```

Implemented In

```
Mage_Checkout_Model_Cart_Product_Api::items
/Users/alanstorm/Sites2011/magento1point6point1.dev/app/code/core/Mage/Checkout/Model/Cart/Product/Api.php
```

Doc Comment

```
/**
 * @param  $quoteId
 * @param  $store
 * @return array
 */
```

Method Implementation Definition

```
    public function items($quoteId, $store = null)
    {
```

```
        $quote = $this->_getQuote($quoteId, $store);
        if (empty($store)) {
            $store = $quote->getStoreId();
        }

        if (!$quote->getItemsCount()) {
            return array();
        }

        $productsResult = array();
        foreach ($quote->getAllItems() as $item) {
            /** @var $item Mage_Sales_Model_Quote_Item */
            $product = $item->getProduct();
            $productsResult[] = array( // Basic product data
                'product_id' => $product->getId(),
                'sku'        => $product->getSku(),
                'set'        => $product->getAttributeSetId(),
                'type'       => $product->getTypeId(),
                'categories' => $product->getCategoryIds(),
                'websites'   => $product->getWebsiteIds()
            );
        }

        return $productsResult;
    }
```

Method: moveToCustomerQuote

```
$client->call($sessionId, 'cart_product.moveToCustomerQuote', $quoteId, $productsData, $store);
```

Implemented In

```
Mage_Checkout_Model_Cart_Product_Api::moveToCustomerQuote
/Users/alanstorm/Sites2011/magento1point6point1.dev/app/code/core/Mage/Checkout/Model/Cart/Product/Api.php
```

Doc Comment

```
/**
 * @param  $quoteId
 * @param  $productsData
 * @param  $store
 * @return bool
 */
```

Method Implementation Definition

```
    public function moveToCustomerQuote($quoteId, $productsData, $store=null)
    {
        $quote = $this->_getQuote($quoteId, $store);

        if (empty($store)) {
            $store = $quote->getStoreId();
        }

        $customer = $quote->getCustomer();
        if (is_null($customer->getId())) {
            $this->_fault('customer_not_set_for_quote');
        }

        /** @var $customerQuote Mage_Sales_Model_Quote */
        $customerQuote = Mage::getModel('sales/quote')
            ->setStoreId($store)
            ->loadByCustomer($customer);

        if (is_null($customerQuote->getId())) {
```

```
            $this->_fault('customer_quote_not_exist');
        }

        if ($customerQuote->getId() == $quote->getId()) {
            $this->_fault('quotes_are_similar');
        }

        $productsData = $this->_prepareProductsData($productsData);
        if (empty($productsData)) {
            $this->_fault('invalid_product_data');
        }

        $errors = array();
        foreach($productsData as $key => $productItem){
            if (isset($productItem['product_id'])) {
                $productByItem = $this->_getProduct($productItem['product_id'], $store, "id");
            } else if (isset($productItem['sku'])) {
                $productByItem = $this->_getProduct($productItem['sku'], $store, "sku");
            } else {
                $errors[] = Mage::helper('checkout')->__("One item of products do not have identifier or sku");
                continue;
            }

            try {
                /** @var $quoteItem Mage_Sales_Model_Quote_Item */
                $quoteItem = $this->_getQuoteItemByProduct($quote, $productByItem, $this->_getProductRequest
($productItem));
                if($quoteItem->getId()){
                    $customerQuote->addItem($quoteItem);
                    $quote->removeItem($quoteItem->getId());
                    unset($productsData[$key]);
                } else {
                     $errors[] = Mage::helper('checkout')->__("One item of products is not belong any of quote
item");
                }
            } catch (Mage_Core_Exception $e) {
                $errors[] = $e->getMessage();
            }
        }

        if (count($productsData) || !empty($errors)) {
            $this->_fault('unable_to_move_all_products', implode(PHP_EOL, $errors));
        }

        try {
            $customerQuote
                ->collectTotals()
                ->save();

            $quote
                ->collectTotals()
                ->save();
        } catch (Exception $e) {
             $this->_fault("product_move_quote_save_fault", $e->getMessage());
        }

        return true;
    }
```

Resource: cart_customer

Method: set

```
$client->call($sessionId, 'cart_customer.set', $quoteId, $customerData, $store);
```

Implemented In

```
Mage_Checkout_Model_Cart_Customer_Api::set
/Users/alanstorm/Sites2011/magento1point6point1.dev/app/code/core/Mage/Checkout/Model/Cart/Customer/Api.php
```

Doc Comment

```
/**
 * Set customer for shopping cart
 *
 * @param int $quoteId
 * @param array|object $customerData
 * @param int | string $store
 * @return int
 */
```

Method Implementation Definition

```
    public function set($quoteId, $customerData, $store = null)
    {
        $quote = $this->_getQuote($quoteId, $store);

        $customerData = $this->_prepareCustomerData($customerData);
        if (!isset($customerData['mode'])) {
            $this->_fault('customer_mode_is_unknown');
        }

        switch($customerData['mode']) {
        case self::MODE_CUSTOMER:
            /** @var $customer Mage_Customer_Model_Customer */
            $customer = $this->_getCustomer($customerData['entity_id']);
            $customer->setMode(self::MODE_CUSTOMER);
            break;

        case self::MODE_REGISTER:
        case self::MODE_GUEST:
            /** @var $customer Mage_Customer_Model_Customer */
            $customer = Mage::getModel('customer/customer')
                ->setData($customerData);

            if ($customer->getMode() == self::MODE_GUEST) {
                $password = $customer->generatePassword();

                $customer
                    ->setPassword($password)
                    ->setConfirmation($password);
            }

            $isCustomerValid = $customer->validate();
            Mage::Log($isCustomerValid);
            if ($isCustomerValid !== true && is_array($isCustomerValid)) {
                $this->_fault('customer_data_invalid', implode(PHP_EOL, $isCustomerValid));
            }
            break;
        }

        try {
            $quote
                ->setCustomer($customer)
                ->setCheckoutMethod($customer->getMode())
```

```
                ->setPasswordHash($customer->encryptPassword($customer->getPassword()))
                ->save();
        } catch (Mage_Core_Exception $e) {
            $this->_fault('customer_not_set', $e->getMessage());
        }

        return true;
    }
```

Method: addresses

```
$client->call($sessionId, 'cart_customer.addresses', $quoteId, $customerAddressData, $store);
```

Implemented In

```
Mage_Checkout_Model_Cart_Customer_Api::setAddresses
/Users/alanstorm/Sites2011/magento1point6point1.dev/app/code/core/Mage/Checkout/Model/Cart/Customer/Api.php
```

Doc Comment

```
/**
 * @param  int $quoteId
 * @param  array of array|object $customerAddressData
 * @param  int|string $store
 * @return int
 */
```

Method Implementation Definition

```
    public function setAddresses($quoteId, $customerAddressData, $store = null)
    {
        $quote = $this->_getQuote($quoteId, $store);

        $customerAddressData = $this->_prepareCustomerAddressData($customerAddressData);
        if (is_null($customerAddressData)) {
            $this->_fault('customer_address_data_empty');
        }

        foreach ($customerAddressData as $addressItem) {
//            switch($addressItem['mode']) {
//            case self::ADDRESS_BILLING:
                /** @var $address Mage_Sales_Model_Quote_Address */
                $address = Mage::getModel("sales/quote_address");
//                break;
//            case self::ADDRESS_SHIPPING:
//                /** @var $address Mage_Sales_Model_Quote_Address */
//                $address = Mage::getModel("sales/quote_address");
//                break;
//            }
            $addressMode = $addressItem['mode'];
            unset($addressItem['mode']);

            if (!empty($addressItem['entity_id'])) {
                $customerAddress = $this->_getCustomerAddress($addressItem['entity_id']);
                if ($customerAddress->getCustomerId() != $quote->getCustomerId()) {
                    $this->_fault('address_not_belong_customer');
                }
                $address->importCustomerAddress($customerAddress);

            } else {
                $address->setData($addressItem);
            }

            $address->implodeStreetAddress();
```

```
            if (($validateRes = $address->validate())!==true) {
                $this->_fault('customer_address_invalid', implode(PHP_EOL, $validateRes));
            }

            switch($addressMode) {
            case self::ADDRESS_BILLING:
                $address->setEmail($quote->getCustomer()->getEmail());

                if (!$quote->isVirtual()) {
                    $usingCase = isset($addressItem['use_for_shipping']) ? (int)$addressItem
['use_for_shipping'] : 0;
                    switch($usingCase) {
                    case 0:
                        $shippingAddress = $quote->getShippingAddress();
                        $shippingAddress->setSameAsBilling(0);
                        break;
                    case 1:
                        $billingAddress = clone $address;
                        $billingAddress->unsAddressId()->unsAddressType();

                        $shippingAddress = $quote->getShippingAddress();
                        $shippingMethod = $shippingAddress->getShippingMethod();
                        $shippingAddress->addData($billingAddress->getData())
                            ->setSameAsBilling(1)
                            ->setShippingMethod($shippingMethod)
                            ->setCollectShippingRates(true);
                        break;
                    }
                }
                $quote->setBillingAddress($address);
                break;

            case self::ADDRESS_SHIPPING:
                $address->setCollectShippingRates(true)
                        ->setSameAsBilling(0);
                $quote->setShippingAddress($address);
                break;
            }

        }

        try {
            $quote
                ->collectTotals()
                ->save();
        } catch (Exception $e) {
            $this->_fault('address_is_not_set', $e->getMessage());
        }

        return true;
    }
```

Resource: cart_shipping

Method: method

```
$client->call($sessionId, 'cart_shipping.method', $quoteId, $shippingMethod, $store);
```

Implemented In

```
Mage_Checkout_Model_Cart_Shipping_Api::setShippingMethod
```

/Users/alanstorm/Sites2011/magento1point6point1.dev/app/code/core/Mage/Checkout/Model/Cart/Shipping/Api.php

Doc Comment

```
/**
 * Set an Shipping Method for Shopping Cart
 *
 * @param  $quoteId
 * @param  $shippingMethod
 * @param  $store
 * @return bool
 */
```

Method Implementation Definition

```
    public function setShippingMethod($quoteId, $shippingMethod, $store = null)
    {
        $quote = $this->_getQuote($quoteId, $store);

        $quoteShippingAddress = $quote->getShippingAddress();
        if(is_null($quoteShippingAddress->getId()) ) {
            $this->_fault("shipping_address_is_not_set");
        }

        $rate = $quote->getShippingAddress()->collectShippingRates()->getShippingRateByCode($shippingMethod);
        if (!$rate) {
            $this->_fault('shipping_method_is_not_available');
        }

        try {
            $quote->getShippingAddress()->setShippingMethod($shippingMethod);
            $quote->collectTotals()->save();
        } catch(Mage_Core_Exception $e) {
            $this->_fault('shipping_method_is_not_set', $e->getMessage());
        }

        return true;
    }
```

Method: list

```
$client->call($sessionId, 'cart_shipping.list', $quoteId, $store);
```

Implemented In

Mage_Checkout_Model_Cart_Shipping_Api::getShippingMethodsList
/Users/alanstorm/Sites2011/magento1point6point1.dev/app/code/core/Mage/Checkout/Model/Cart/Shipping/Api.php

Doc Comment

```
/**
 * Get list of available shipping methods
 *
 * @param  $quoteId
 * @param  $store
 * @return array
 */
```

Method Implementation Definition

```
    public function getShippingMethodsList($quoteId, $store=null)
    {
        $quote = $this->_getQuote($quoteId, $store);

        $quoteShippingAddress = $quote->getShippingAddress();
        if (is_null($quoteShippingAddress->getId())) {
            $this->_fault("shipping_address_is_not_set");
        }
```

```
        try {
            $quoteShippingAddress->collectShippingRates()->save();
            $groupedRates = $quoteShippingAddress->getGroupedAllShippingRates();

            $ratesResult = array();
            foreach ($groupedRates as $carrierCode => $rates ) {
                $carrierName = $carrierCode;
                if (!is_null(Mage::getStoreConfig('carriers/'.$carrierCode.'/title'))) {
                    $carrierName = Mage::getStoreConfig('carriers/'.$carrierCode.'/title');
                }

                foreach ($rates as $rate) {
                    $rateItem = $this->_getAttributes($rate, "quote_shipping_rate");
                    $rateItem['carrierName'] = $carrierName;
                    $ratesResult[] = $rateItem;
                    unset($rateItem);
                }
            }
        } catch (Mage_Core_Exception $e) {
            $this->_fault('shipping_methods_list_could_not_be_retrived', $e->getMessage());
        }

        return $ratesResult;
    }
```

Resource: cart_payment

Method: method

```
$client->call($sessionId, 'cart_payment.method', $quoteId, $paymentData, $store);
```

Implemented In

```
Mage_Checkout_Model_Cart_Payment_Api::setPaymentMethod
/Users/alanstorm/Sites2011/magento1point6point1.dev/app/code/core/Mage/Checkout/Model/Cart/Payment/Api.php
```

Doc Comment

```
/**
 * @param  $quoteId
 * @param  $paymentData
 * @param  $store
 * @return bool
 */
```

Method Implementation Definition

```
    public function setPaymentMethod($quoteId, $paymentData, $store=null)
    {
        $quote = $this->_getQuote($quoteId, $store);
        $store = $quote->getStoreId();
        $paymentData = $this->_preparePaymentData($paymentData);

        if (empty($paymentData)) {
            $this->_fault("payment_method_empty");
        }

        if ($quote->isVirtual()) {
            // check if billing address is set
            if (is_null($quote->getBillingAddress()->getId()) ) {
                $this->_fault('billing_address_is_not_set');
            }
            $quote->getBillingAddress()->setPaymentMethod(isset($paymentData['method']) ? $paymentData
['method'] : null);
```

```
        } else {
            // check if shipping address is set
            if (is_null($quote->getShippingAddress()->getId()) ) {
                $this->_fault('shipping_address_is_not_set');
            }
            $quote->getShippingAddress()->setPaymentMethod(isset($paymentData['method']) ? $paymentData
['method'] : null);
        }

        if (!$quote->isVirtual() && $quote->getShippingAddress()) {
            $quote->getShippingAddress()->setCollectShippingRates(true);
        }

        $total = $quote->getBaseSubtotal();
        $methods = Mage::helper('payment')->getStoreMethods($store, $quote);
        foreach ($methods as $key=>$method) {
            if ($method->getCode() == $paymentData['method']) {
                /** @var $method Mage_Payment_Model_Method_Abstract */
                if (!($this->_canUsePaymentMethod($method, $quote)
                        && ($total != 0
                            || $method->getCode() == 'free'
                            || ($quote->hasRecurringItems() && $method->canManageRecurringProfiles()))) {
                    $this->_fault("method_not_allowed");
                }
            }
        }

        try {
            $payment = $quote->getPayment();
            $payment->importData($paymentData);


            $quote->setTotalsCollectedFlag(false)
                    ->collectTotals()
                    ->save();
        } catch (Mage_Core_Exception $e) {
            $this->_fault('payment_method_is_not_set', $e->getMessage());
        }
        return true;
    }
```

Method: list

```
$client->call($sessionId, 'cart_payment.list', $quoteId, $store);
```

Implemented In

```
Mage_Checkout_Model_Cart_Payment_Api::getPaymentMethodsList
/Users/alanstorm/Sites2011/magento1point6point1.dev/app/code/core/Mage/Checkout/Model/Cart/Payment/Api.php
```

Doc Comment

```
/**
 * @param  $quoteId
 * @param  $store
 * @return array
 */
```

Method Implementation Definition

```
    public function getPaymentMethodsList($quoteId, $store=null)
    {
        $quote = $this->_getQuote($quoteId, $store);
        $store = $quote->getStoreId();
```

```
        $total = $quote->getBaseSubtotal();

        $methodsResult = array();
        $methods = Mage::helper('payment')->getStoreMethods($store, $quote);
        foreach ($methods as $key=>$method) {
            /** @var $method Mage_Payment_Model_Method_Abstract */
            if ($this->_canUsePaymentMethod($method, $quote)
                    && ($total != 0
                        || $method->getCode() == 'free'
                        || ($quote->hasRecurringItems() && $method->canManageRecurringProfiles()))) {
                $methodsResult[] =
                        array(
                            "code" => $method->getCode(),
                            "title" => $method->getTitle(),
                            "ccTypes" => $this->_getPaymentMethodAvailableCcTypes($method)
                        );
            }
        }

        return $methodsResult;
    }
```

Resource: cart_coupon

Method: add

```
$client->call($sessionId, 'cart_coupon.add', $quoteId, $couponCode, $store);
```

Implemented In

```
Mage_Checkout_Model_Cart_Coupon_Api::add
/Users/alanstorm/Sites2011/magento1point6point1.dev/app/code/core/Mage/Checkout/Model/Cart/Coupon/Api.php
```

Doc Comment

```
/**
 * @param  $quoteId
 * @param  $couponCode
 * @param  $storeId
 * @return bool
 */
```

Method Implementation Definition

```
    public function add($quoteId, $couponCode, $store = null)
    {
        return $this->_applyCoupon($quoteId, $couponCode, $store = null);
    }
```

Method: remove

```
$client->call($sessionId, 'cart_coupon.remove', $quoteId, $store);
```

Implemented In

```
Mage_Checkout_Model_Cart_Coupon_Api::remove
/Users/alanstorm/Sites2011/magento1point6point1.dev/app/code/core/Mage/Checkout/Model/Cart/Coupon/Api.php
```

Doc Comment

```
/**
 * @param  $quoteId
 * @param  $storeId
 * @return void
 */
```

Method Implementation Definition

```
    public function remove($quoteId, $store = null)
    {
        $couponCode = '';
        return $this->_applyCoupon($quoteId, $couponCode, $store);
    }
```

Resource: catalog_product_tag

Method: list

```
$client->call($sessionId, 'catalog_product_tag.list', $productId, $store);
```

Implemented In

```
Mage_Tag_Model_Api::items
/Users/alanstorm/Sites2011/magento1point6point1.dev/app/code/core/Mage/Tag/Model/Api.php
```

Doc Comment

```
/**
 * Retrieve list of tags for specified product
 *
 * @param int $productId
 * @param string|int $store
 * @return array
 */
```

Method Implementation Definition

```
    public function items($productId, $store = null)
    {
        $result = array();
        // fields list to return
        $fieldsForResult = array('tag_id', 'name');

        /** @var $product Mage_Catalog_Model_Product */
        $product = Mage::getModel('catalog/product')->load($productId);
        if (!$product->getId()) {
            $this->_fault('product_not_exists');
        }

        /** @var $tags Mage_Tag_Model_Resource_Tag_Collection */
        $tags = Mage::getModel('tag/tag')->getCollection()->joinRel()->addProductFilter($productId);
        if ($store) {
            $tags->addStoreFilter($this->_getStoreId($store));
        }

        /** @var $tag Mage_Tag_Model_Tag */
        foreach ($tags as $tag) {
            $result[$tag->getId()] = $tag->toArray($fieldsForResult);
        }

        return $result;
    }
```

Method: info

```
$client->call($sessionId, 'catalog_product_tag.info', $tagId, $store);
```

Implemented In

```
Mage_Tag_Model_Api::info
/Users/alanstorm/Sites2011/magento1point6point1.dev/app/code/core/Mage/Tag/Model/Api.php
```

Doc Comment

```
/**
```

```
 * Retrieve tag info as array('name'-> .., 'status' => ..,
 * 'base_popularity' => .., 'products' => array($productId => $popularity, ...))
 *
 * @param int $tagId
 * @param string|int $store
 * @return array
 */
```

Method Implementation Definition

```
    public function info($tagId, $store)
    {
        $result = array();
        $storeId = $this->_getStoreId($store);
        /** @var $tag Mage_Tag_Model_Tag */
        $tag = Mage::getModel('tag/tag')->setStoreId($storeId)->setAddBasePopularity()->load($tagId);
        if (!$tag->getId()) {
            $this->_fault('tag_not_exists');
        }
        $result['status'] = $tag->getStatus();
        $result['name'] = $tag->getName();
        $result['base_popularity'] = (is_numeric($tag->getBasePopularity())) ? $tag->getBasePopularity() : 0;
        // retrieve array($productId => $popularity, ...)
        $result['products'] = array();
        $relatedProductsCollection = $tag->getEntityCollection()->addTagFilter($tagId)
            ->addStoreFilter($storeId)->addPopularity($tagId);
        foreach ($relatedProductsCollection as $product) {
            $result['products'][$product->getId()] = $product->getPopularity();
        }

        return $result;
    }
```

Method: add

```
$client->call($sessionId, 'catalog_product_tag.add', $data);
```

Implemented In

```
Mage_Tag_Model_Api::add
/Users/alanstorm/Sites2011/magento1point6point1.dev/app/code/core/Mage/Tag/Model/Api.php
```

Doc Comment

```
/**
 * Add tag(s) to product.
 * Return array of added/updated tags as array($tagName => $tagId, ...)
 *
 * @param array $data
 * @return array
 */
```

Method Implementation Definition

```
    public function add($data)
    {
        $data = $this->_prepareDataForAdd($data);
        /** @var $product Mage_Catalog_Model_Product */
        $product = Mage::getModel('catalog/product')->load($data['product_id']);
        if (!$product->getId()) {
            $this->_fault('product_not_exists');
        }
        /** @var $customer Mage_Customer_Model_Customer */
        $customer = Mage::getModel('customer/customer')->load($data['customer_id']);
        if (!$customer->getId()) {
            $this->_fault('customer_not_exists');
```

```
        }
        $storeId = $this->_getStoreId($data['store']);

        try {
            /** @var $tag Mage_Tag_Model_Tag */
            $tag = Mage::getModel('tag/tag');
            $tagNamesArr = Mage::helper('tag')->cleanTags(Mage::helper('tag')->extractTags($data['tag']));
            foreach ($tagNamesArr as $tagName) {
                // unset previously added tag data
                $tag->unsetData();
                $tag->loadByName($tagName);
                if (!$tag->getId()) {
                    $tag->setName($tagName)
                        ->setFirstCustomerId($customer->getId())
                        ->setFirstStoreId($storeId)
                        ->setStatus($tag->getPendingStatus())
                        ->save();
                }
                $tag->saveRelation($product->getId(), $customer->getId(), $storeId);
                $result[$tagName] = $tag->getId();
            }
        } catch (Mage_Core_Exception $e) {
            $this->_fault('save_error', $e->getMessage());
        }

        return $result;
    }
```

Method: update

```
$client->call($sessionId, 'catalog_product_tag.update', $tagId, $data, $store);
```

Implemented In

Mage_Tag_Model_Api::update
/Users/alanstorm/Sites2011/magento1point6point1.dev/app/code/core/Mage/Tag/Model/Api.php

Doc Comment

```
/**
 * Change existing tag information
 *
 * @param int $tagId
 * @param array $data
 * @param string|int $store
 * @return bool
 */
```

Method Implementation Definition

```
    public function update($tagId, $data, $store)
    {
        $data = $this->_prepareDataForUpdate($data);
        $storeId = $this->_getStoreId($store);
        /** @var $tag Mage_Tag_Model_Tag */
        $tag = Mage::getModel('tag/tag')->setStoreId($storeId)->setAddBasePopularity()->load($tagId);
        if (!$tag->getId()) {
            $this->_fault('tag_not_exists');
        }

        // store should be set for 'base_popularity' to be saved in Mage_Tag_Model_Resource_Tag::_afterSave()
        $tag->setStore($storeId);
        if (isset($data['base_popularity'])) {
            $tag->setBasePopularity($data['base_popularity']);
        }
```

```
        if (isset($data['name'])) {
            $tag->setName(trim($data['name']));
        }
        if (isset($data['status'])) {
            // validate tag status
            if (!in_array($data['status'], array(
                $tag->getApprovedStatus(), $tag->getPendingStatus(), $tag->getDisabledStatus()))) {
                $this->_fault('invalid_data');
            }
            $tag->setStatus($data['status']);
        }

        try {
            $tag->save();
        } catch (Mage_Core_Exception $e) {
            $this->_fault('save_error', $e->getMessage());
        }

        return true;
    }
```

Method: remove

```
$client->call($sessionId, 'catalog_product_tag.remove', $tagId);
```

Implemented In

```
Mage_Tag_Model_Api::remove
/Users/alanstorm/Sites2011/magento1point6point1.dev/app/code/core/Mage/Tag/Model/Api.php
```

Doc Comment

```
/**
 * Remove existing tag
 *
 * @param int $tagId
 * @return bool
 */
```

Method Implementation Definition

```
    public function remove($tagId)
    {
        /** @var $tag Mage_Tag_Model_Tag */
        $tag = Mage::getModel('tag/tag')->load($tagId);
        if (!$tag->getId()) {
            $this->_fault('tag_not_exists');
        }
        try {
            $tag->delete();
        } catch (Mage_Core_Exception $e) {
            $this->_fault('remove_error', $e->getMessage());
        }

        return true;
    }
```

Resource: giftmessage

Method: setForQuote

```
$client->call($sessionId, 'giftmessage.setForQuote', $quoteId, $giftMessage, $store);
```

Implemented In

Mage_GiftMessage_Model_Api::setForQuote
/Users/alanstorm/Sites2011/magento1point6point1.dev/app/code/core/Mage/GiftMessage/Model/Api.php

Doc Comment

```
/**
 * Set GiftMessage for a Quote.
 *
 * @param String $quoteId
 * @param AssociativeArray $giftMessage
 * @param String $store
 * @return AssociativeArray
 */
```

Method Implementation Definition

```
    public function setForQuote($quoteId, $giftMessage, $store = null)
    {
        /** @var $quote Mage_Sales_Model_Quote */
        $quote = $this->_getQuote($quoteId, $store);

        $giftMessage = $this->_prepareData($giftMessage);
        if (empty($giftMessage)) {
            $this->_fault('giftmessage_invalid_data');
        }

        $giftMessage['type'] = 'quote';
        $giftMessages = array($quoteId => $giftMessage);
        $request = new Mage_Core_Controller_Request_Http();
        $request->setParam("giftmessage", $giftMessages);

        return $this->_setGiftMessage($quote->getId(), $request, $quote);
    }
```

Method: setForQuoteItem

```
$client->call($sessionId, 'giftmessage.setForQuoteItem', $quoteItemId, $giftMessage, $store);
```

Implemented In

Mage_GiftMessage_Model_Api::setForQuoteItem
/Users/alanstorm/Sites2011/magento1point6point1.dev/app/code/core/Mage/GiftMessage/Model/Api.php

Doc Comment

```
/**
 * Set GiftMessage for a QuoteItem by its Id.
 *
 * @param String $quoteItemId
 * @param AssociativeArray $giftMessage
 * @param String $store
 * @return AssociativeArray
 */
```

Method Implementation Definition

```
    public function setForQuoteItem($quoteItemId, $giftMessage, $store = null)
    {
        /** @var $quote Mage_Sales_Model_Quote_Item */
        $quoteItem = Mage::getModel('sales/quote_item')->load($quoteItemId);
        if (is_null($quoteItem->getId())) {
            $this->_fault("quote_item_not_exists");
        }

        /** @var $quote Mage_Sales_Model_Quote */
        $quote = $this->_getQuote($quoteItem->getQuoteId(), $store);

        $giftMessage = $this->_prepareData($giftMessage);
```

```
        $giftMessage['type'] = 'quote_item';

        $giftMessages = array($quoteItem->getId() => $giftMessage);

        $request = new Mage_Core_Controller_Request_Http();
        $request->setParam("giftmessage", $giftMessages);

        return $this->_setGiftMessage($quoteItemId, $request, $quote);
    }
```

Method: setForQuoteProduct

```
$client->call($sessionId, 'giftmessage.setForQuoteProduct', $quoteId, $productsAndMessages, $store);
```

Implemented In

Mage_GiftMessage_Model_Api::setForQuoteProduct
/Users/alanstorm/Sites2011/magento1point6point1.dev/app/code/core/Mage/GiftMessage/Model/Api.php

Doc Comment

```
/**
 * Set a GiftMessage to QuoteItem by product
 *
 * @param String $quoteId
 * @param Array $productsAndMessages
 * @param String $store
 * @return array
 */
```

Method Implementation Definition

```
    public function setForQuoteProduct($quoteId, $productsAndMessages, $store = null)
    {
        /** @var $quote Mage_Sales_Model_Quote */
        $quote = $this->_getQuote($quoteId, $store);

        $productsAndMessages = $this->_prepareData($productsAndMessages);
        if (empty($productsAndMessages)) {
            $this->_fault('invalid_data');
        }

        if (count($productsAndMessages) == 2
                && isset($productsAndMessages['product'])
                && isset($productsAndMessages['message'])) {
            $productsAndMessages = array($productsAndMessages);
        }

        $results = array();
        foreach ($productsAndMessages as $productAndMessage) {
            if (isset($productAndMessage['product']) && isset($productAndMessage['message'])) {
                $product = $this->_prepareData($productAndMessage['product']);
                if (empty($product)) {
                    $this->_fault('product_invalid_data');
                }
                $message = $this->_prepareData($productAndMessage['message']);
                if (empty($message)) {
                    $this->_fault('giftmessage_invalid_data');
                }

                if (isset($product['product_id'])) {
                    $productByItem = $this->_getProduct($product['product_id'], $store, "id");
                } elseif (isset($product['sku'])) {
                    $productByItem = $this->_getProduct($product['sku'], $store, "sku");
                } else {
```

```
                continue;
            }

            $productObj = $this->_getProductRequest($product);
            $quoteItem = $this->_getQuoteItemByProduct($quote, $productByItem, $productObj);
            $results[] = $this->setForQuoteItem($quoteItem->getId(), $message, $store);
        }
    }

    return $results;
}
```

Resource: catalog_product_downloadable_link

Method: add

```
$client->call($sessionId, 'catalog_product_downloadable_link.add', $productId, $resource, $resourceType, $store, $identifierType);
```

Implemented In

```
Mage_Downloadable_Model_Link_Api::add
/Users/alanstorm/Sites2011/magento1point6point1.dev/app/code/core/Mage/Downloadable/Model/Link/Api.php
```

Doc Comment

```
/**
 * Add downloadable content to product
 *
 * @param int|string $productId
 * @param array $resource
 * @param string $resourceType
 * @param string|int|null $store
 * @param string|null $identifierType ('sku'|'id')
 * @return boolean
 */
```

Method Implementation Definition

```
public function add($productId, $resource, $resourceType, $store = null, $identifierType = null)
{
    try {
        $this->_getValidator()->validateType($resourceType);
        $this->_getValidator()->validateAttributes($resource, $resourceType);
    } catch (Exception $e) {
        $this->_fault('validation_error', $e->getMessage());
    }

    $resource['is_delete'] = 0;
    if ($resourceType == 'link') {
        $resource['link_id'] = 0;
    } elseif ($resourceType == 'sample') {
        $resource['sample_id'] = 0;
    }

    if ($resource['type'] == 'file') {
        if (isset($resource['file'])) {
            $resource['file'] = $this->_uploadFile($resource['file'], $resourceType);
        }
        unset($resource[$resourceType.'_url']);
    } elseif ($resource['type'] == 'url') {
        unset($resource['file']);
    }

    if ($resourceType == 'link' && $resource['sample']['type'] == 'file') {
```

```
        if (isset($resource['sample']['file'])) {
            $resource['sample']['file'] = $this->_uploadFile($resource['sample']['file'], 'link_samples');
        }
        unset($resource['sample']['url']);
    } elseif ($resourceType == 'link' && $resource['sample']['type'] == 'url') {
        $resource['sample']['file'] = null;
    }

    $product = $this->_getProduct($productId, $store, $identifierType);
    try {
        $downloadable = array($resourceType => array($resource));
        $product->setDownloadableData($downloadable);
        $product->save();
    } catch (Exception $e) {
        $this->_fault('save_error', $e->getMessage());
    }

    return true;
}
```

Method: list

```
$client->call($sessionId, 'catalog_product_downloadable_link.list', $productId, $store, $identifierType);
```

Implemented In

Mage_Downloadable_Model_Link_Api::items
/Users/alanstorm/Sites2011/magento1point6point1.dev/app/code/core/Mage/Downloadable/Model/Link/Api.php

Doc Comment

```
/**
 * Retrieve downloadable product links
 *
 * @param int|string $productId
 * @param string|int $store
 * @param string $identifierType ('sku'|'id')
 * @return array
 */
```

Method Implementation Definition

```
    public function items($productId, $store = null, $identifierType = null)
    {
        $product = $this->_getProduct($productId, $store, $identifierType);

        $linkArr = array();
        $links = $product->getTypeInstance(true)->getLinks($product);
        foreach ($links as $item) {
            $tmpLinkItem = array(
                'link_id' => $item->getId(),
                'title' => $item->getTitle(),
                'price' => $item->getPrice(),
                'number_of_downloads' => $item->getNumberOfDownloads(),
                'is_shareable' => $item->getIsShareable(),
                'link_url' => $item->getLinkUrl(),
                'link_type' => $item->getLinkType(),
                'sample_file' => $item->getSampleFile(),
                'sample_url' => $item->getSampleUrl(),
                'sample_type' => $item->getSampleType(),
                'sort_order' => $item->getSortOrder()
            );
            $file = Mage::helper('downloadable/file')->getFilePath(
                Mage_Downloadable_Model_Link::getBasePath(), $item->getLinkFile()
            );
```

```
            if ($item->getLinkFile() && !is_file($file)) {
                Mage::helper('core/file_storage_database')->saveFileToFilesystem($file);
            }

            if ($item->getLinkFile() && is_file($file)) {
                $name = Mage::helper('downloadable/file')->getFileFromPathFile($item->getLinkFile());
                $tmpLinkItem['file_save'] = array(
                    array(
                        'file' => $item->getLinkFile(),
                        'name' => $name,
                        'size' => filesize($file),
                        'status' => 'old'
                    ));
            }
            $sampleFile = Mage::helper('downloadable/file')->getFilePath(
                Mage_Downloadable_Model_Link::getBaseSamplePath(), $item->getSampleFile()
            );
            if ($item->getSampleFile() && is_file($sampleFile)) {
                $tmpLinkItem['sample_file_save'] = array(
                    array(
                        'file' => $item->getSampleFile(),
                        'name' => Mage::helper('downloadable/file')->getFileFromPathFile($item->getSampleFile
()),
                        'size' => filesize($sampleFile),
                        'status' => 'old'
                    ));
            }
            if ($item->getNumberOfDownloads() == '0') {
                $tmpLinkItem['is_unlimited'] = 1;
            }
            if ($product->getStoreId() && $item->getStoreTitle()) {
                $tmpLinkItem['store_title'] = $item->getStoreTitle();
            }
            if ($product->getStoreId() && Mage::helper('downloadable')->getIsPriceWebsiteScope()) {
                $tmpLinkItem['website_price'] = $item->getWebsitePrice();
            }
            $linkArr[] = $tmpLinkItem;
        }
        unset($item);
        unset($tmpLinkItem);
        unset($links);

        $samples = $product->getTypeInstance(true)->getSamples($product)->getData();
        return array('links' => $linkArr, 'samples' => $samples);
    }
```

Method: remove

```
$client->call($sessionId, 'catalog_product_downloadable_link.remove', $linkId, $resourceType);
```

Implemented In

```
Mage_Downloadable_Model_Link_Api::remove
/Users/alanstorm/Sites2011/magento1point6point1.dev/app/code/core/Mage/Downloadable/Model/Link/Api.php
```

Doc Comment

```
/**
 * Remove downloadable product link
 * @param string $linkId
 * @param string $resourceType
 * @return bool
```

```
 */
```

Method Implementation Definition

```
    public function remove($linkId, $resourceType)
    {
        try {
            $this->_getValidator()->validateType($resourceType);
        } catch (Exception $e) {
            $this->_fault('validation_error', $e->getMessage());
        }

        switch($resourceType) {
            case 'link':
                $downloadableModel = Mage::getSingleton('downloadable/link');
                break;
            case 'sample':
                $downloadableModel = Mage::getSingleton('downloadable/sample');
                break;
        }

        $downloadableModel->load($linkId);
        if (is_null($downloadableModel->getId())) {
            $this->_fault('link_was_not_found');
        }

        try {
            $downloadableModel->delete();
        } catch (Exception $e) {
            $this->_fault('remove_error', $e->getMessage());
        }

        return true;
    }
```