Mobile Application Store Collection Service Design Document

Date: October 30, 2013

Author: Fanxing Meng

Reviewer(s): Jayaprakash Ganta

Introduction

This document defines the design for the Mobile Application Store Collection Service. Mobile application stores provide contents including applications, ringtones and wallpaper for mobile device users. Collections enable administrators to organize similar products into collections to facilitate end-users in searching for relevant products.

Overview

In this phase we have developed the collection service of the app store. The collection service allows administrator to create and modify collection-specific criteria and contents. Specifically, dynamic collection contains search criteria identical to product query and will execute through the query engine when its contents are needed. Static collection contains a static list of products belonging to it. When traversing inside the collection, one of its method retrieves its product information in the product catalog for most-up-to-date information. Both types of collections may contain other collections.

Requirements

The structural requirement is that both the collection class and the product class extend a collectable class, and that collection can contain any number of collectable objects. A collection contains its ID, name, description, and a list of child collections. Static collection contains an extra field of product IDs, while dynamic collections contains search criteria.

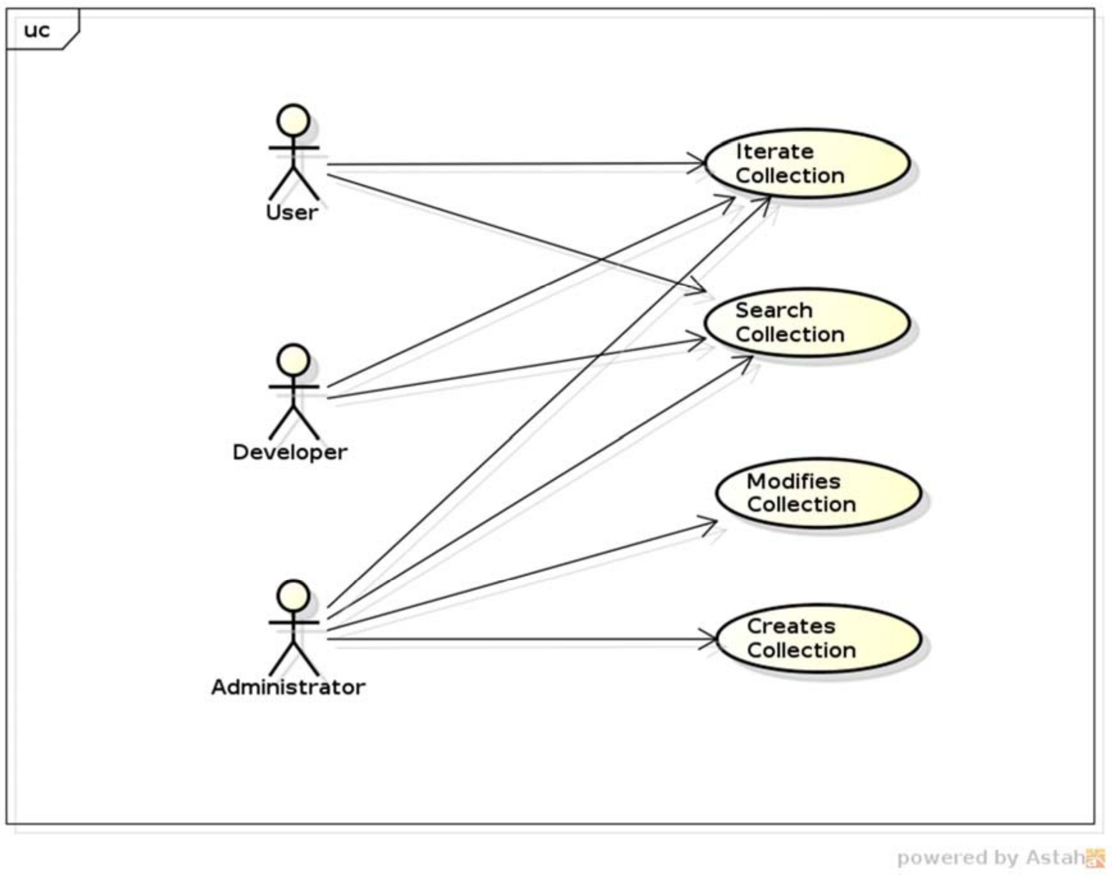
The functional requirement is to use factory pattern to create collections, add content IDs to static collections, and set search criteria to dynamic collections. These are administrative functions that needs authentication. General functions are searching collections and iterating over collections using iterator pattern. When iterating, indirectly referenced product information in collections will be searched, updated, and dereferenced for manipulating.

Use Cases

Current design supports the following use cases:

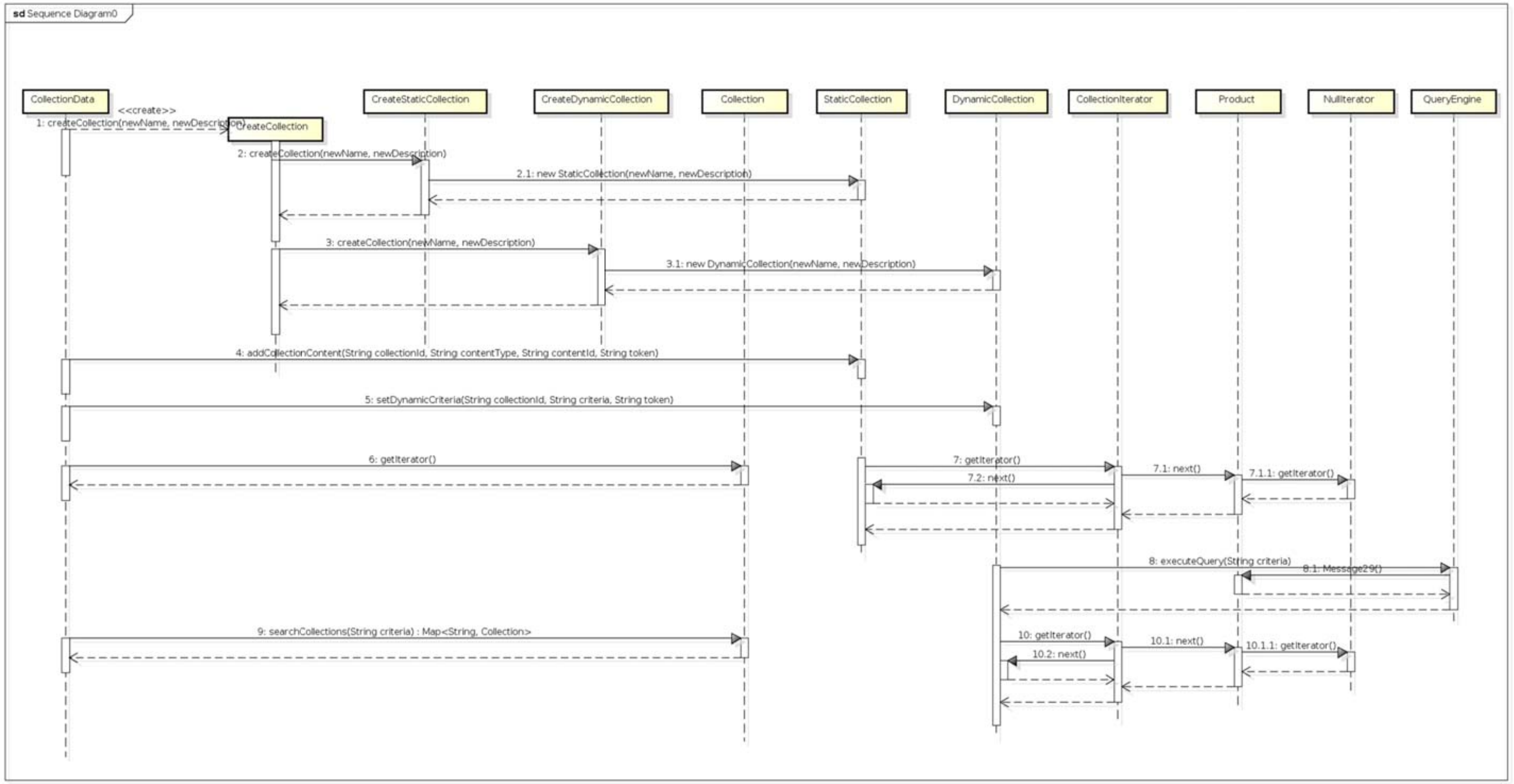
Customers and developers use collection search to find collections with name or description containing the keyword they specify. They can also use iterator to traverse over all collections, or traverse into one specific collection with all its contained child collections and products.

Administrators can manage collections besides searching and iterating. Managing includes creating new static or dynamic collections, add child collections and products to static collections, and add child collections and update search criteria to dynamic collections.



Sequence Diagram

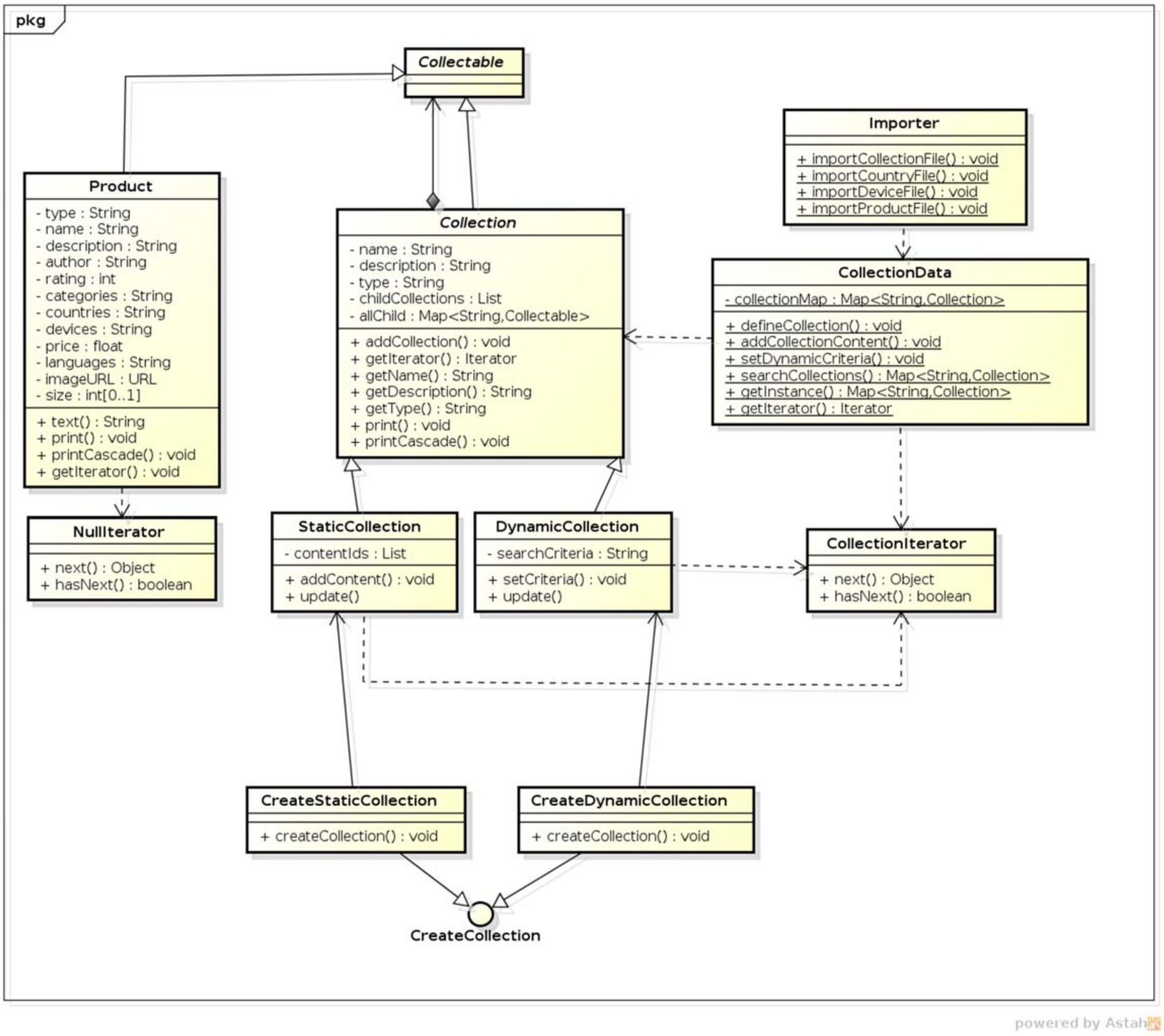
Each class will cooperate with other classes as shown below:


sd Sequence Diagram0
CollectionData
CreateCollection
CreateStaticCollection
CreateDynamicCollection
Collection
StaticCollection
DynamicCollection
CollectionIterator
Product
NullIterator
QueryEngine
<<create>>
1: createCollection(newName, newDescription)
2: createCollection(newName, newDescription)
2.1: new StaticCollection(newName, newDescription)
3: createCollection(newName, newDescription)
3.1: new DynamicCollection(newName, newDescription)
4: addCollectionContent(String collectionId, String contentType, String contentId, String token)
5: setDynamicCriteria(String collectionId, String criteria, String token)
6: getIterator()
7: getIterator()
7.1: next()
7.1.1: getIterator()
7.2: next()
8: executeQuery(String criteria)
8.1: Message29()
9: searchCollections(String criteria) : Map<String, Collection>
10: getIterator()
10.1: next()
10.1.1: getIterator()
10.2: next()
powered by Astah

Implementation

Class Diagram

The following class diagram defines the classes defined in this design.



Class Dictionary

This section specifies the class dictionary for the collection service. The classes are defined within the package “cscie97.asn3.ecommerce.collection”.

Collectable

Collectable class defines the methods available to collections and products.

Methods

Method Name	Signature	Description
addCollection	(newId:String):void	Add a child collection to current collection.
addContent	(filename:String, pass:String):void	Add a product ID to current static collection.
setCriteria	(newCriteria:String):void	Set the search criteria for current dynamic collection.
getIterator	(void):Iterator	Get iterator of collections and products.
print	(void):void	Print formatted string of product or collection information.
printCascade	(void):void	Print formatted string of collection information with all its children.
update	(void):void	Updates static collection by content ID list or dynamic collection by criteria.
getName	(void):String	Returns the product or collection name.
getType	(void):String	Returns the type of product or collection.
getDescription	(void):String	Returns the description.
getAuthor	(void):String	Returns the author.
getRating	(void):int	Returns the rating.
getCatagories	(void):String	Returns the categories.
getCountries	(void):String	Returns the countries downloadable at.
getDevices	(void):String	Returns the devices supported.
getPrice	(void):float	Returns the price.
getLanguages	(void):String	Returns the languages.

getImageURL	(void):URL	Returns the image URL.
getSize	(void):int	Returns the size. If not an application, returned value is -1.
text	(void):String	Returns the name and description.

Collection

The Collection class provides the fields that are common to both static and dynamic collections and overrides some methods in the collectable class that are common to all collections.

Methods

Method Name	Signature	Description
addCollection		Add a child collection to current collection.
print	(void):void	Print formatted string of product or collection information.
printCascade	(void):void	Print formatted string of collection information with all its children.
getName	(void):String	Returns the collection name.
getType	(void):String	Returns the type of collection.
getDescription	(void):String	Returns the description.
getIterator	Iterator	Get a recursive collection iterator of collections.

Associations

Association Name	Type	Description
allChild	Map<String, Collectable>	All child elements including collections and static content / dynamic searched results.
childCollections	List<String>	The IDs of child collections.

Properties

Property Name	Type	Description

name	String	Protected name field for the collection.
type	String	Protected type field for the collection.
description	String	Protected description field for the collection.

DynamicCollection

This class DynamicCollection defines unique variables for dynamic collections and special methods unique to it.

Methods

Method Name	Signature	Description
setCriteria	(newCriteria:String):void	Set the search criteria for current dynamic collection.
update	(void):void	Updates dynamic collection by criteria.

Properties

Property Name	Type	Description
searchCriteria	String	The search criteria for a dynamic collection.

StaticCollection

This class StaticCollection defines unique variables for static collections and special methods unique to it.

Methods

Method Name	Signature	Description
addContent	(newId:String):void	Add a product ID to current static collection.
update	(void):void	Updates static collection by content ID list or dynamic collection by criteria.

Associations

Association Name	Type	Description

contentIds	List<String>	The content IDs of a static collection.

CollectionData

This class CollectionData manages a singleton collectionMap and controls all modifications and creations of collections

Methods

Method Name	Signature	Description
addCollectionContent	(collectionId:String, contentType:String, contentId:String, token:String):void	Adds content ID to static collection.
defineCollection	(newType:String, newId:String, newName:String, newDescription:String, token:String):void	Define a collection.
getInstance	(void): Map<String, Collection>	Gets the single instance of CollectionData.
getIterator	(void):Iterator	Gets the default iterator over all existing collections.
searchCollections	(criteria:String): Map<String, Collection>	Search for text fields in all collections.
setDynamicCriteria	(collectionId:String, newCriteria:String, token:String):void	Sets the search criteria for dynamic collection.

Associations

Association Name	Type	Description
collectionMap	Map<String, Collection>	The singleton collection map.

CreateCollection

This interface CreateCollection sets the method of how to create a collection.

Methods

Method Name	Signature	Description
createCollection	(newName:String, newDescription:String):Collection	The Abstract method createCollection that uses name and description to create a collection

CreateDynamicCollection

The Class CreateDynamicCollection that creates a dynamic collection

Methods

Method Name	Signature	Description
createCollection	(newName:String, newDescription:String):Collection	The method createCollection that uses name and description to create a dynamic collection

CreateStaticCollection

The Class CreateStaticCollection that creates a static collection

Methods

Method Name	Signature	Description
createCollection	(newName:String, newDescription:String):Collection	The method createCollection that uses name and description to create a static collection

NullIterator

The Class NullIterator that implements java's iterator but returns null and false for products as they are the end of the collectable tree.

Methods

Method Name	Signature	Description
hasNext	(void):boolean	Whether there is a next element at the end of collectable tree, which is false.
next	(void):Object	The next element at the end of collectable tree, which has none.
remove()	(void):void	remove()

CollectionIterator

The Class CollectionIterator that implements java's iterator but use an auxiliary stack to traverse the collectable tree.

Methods

Method Name	Signature	Description
hasNext	(void):boolean	Whether there is a next element at a node of the collectable tree.
next	(void):Object	The next element at a node of the collectable tree.
remove()	(void):void	remove()

Properties

Property Name	Type	Description
stack	Stack	The stack of iterators.

Implementation Details

There are several details that diverged from the requirement document.

On structure, all collection classes have been added a map of <string, collectable> called allChild. This is mainly to serve as a temporary storage to be used in iteration of the current level of collectables.

For the print method, there are two types of printing: although both work exactly the same on products, print() just prints a collection's name and description, while printCascade takes advantage of the recursive iterator to print all child collections and products within it.

To simplify the overall logic in iteration, we use two types of iterators, one for collections which may contain other collectables, and a null iterator for products which cannot contain any other collectables. In this way we achieve simple traversal by just using a stack and eliminates the need to detect object types at every point.

Testing

Implement a test driver class called TestDriver that implements a static main() method. The main() method should accept 5 parameters: an input countries data file, a devices data file, a products data file, a query file and a collection file. The main method will call the Importer. importCountryFile(),Importer.importDeviceFile(), Importer.importProductFile() and Importer.ImportCollectionFile() method, passing in the name of the file and a password.

After loading the input files, the main() method will invoke 4 different test functions: searchCollections that search on collection texts, iterateAll() that invokes the default iterator, iterateCollection() that iterates within a specific collection, and finally a printCollection() which achieves the same goal as the third function but takes advantage of the packed-in iterator.

The TestDriver class should be defined within the package “cscie97.asn3.test”.

Bugs

When putting child collections and products into a temporary map of <String, Collectable>, the sequence is not enforced, so when traversing at a specific level, interleaving between collections and products can occur.