Using QCache

Description

The QCache class is a template class that provides a cache. **QCache<Key, T>** defines a cache that stores objects of type **T** associated with keys of type **Key**. This class can be used to solve problems where the user needs to maintain a cache of some items. For example, if the user wants to cache the last 10 browser links visited, or the names of the last 10 music/video files played. The advantage of QCache is that it automatically takes ownership of the objects that are inserted into the cache and deletes them to make room for new objects, if necessary. The policy of removal/deletion is based on **Least Recently Used Algorithm**. Each object has a cost associated with it and the cache has a maximum cost (default is 100). When inserting objects, if the total cost of all objects exceeds the maximum cost then the object which was Least Recently Used is deleted to make room for the new object.

Solution

Code:

```
// define the cache, default max cost = 100
QCache<int, QString> cacheFileNames;

// insert an item with cost = 50
// insertion success because Maxcost =100 and total object cost = 50
int key=0;
QString *itemToInsert1 = new QString;
*itemToInsert1 = "File1";
cache.insert(key,itemToInsert1,50);

// insert an another item with cost = 50
// insertion success because Maxcost =100 and total object cost = 50+50
key=1;
QString *itemToInsert2 = new QString;
*itemToInsert2 = "File2";
cache.insert(key,itemToInsert2,50);

//Try to insert an another item with cost = 50
// Now Maxcost =100 and total object cost will be = 50+50+50, which is
// more than Maxcost, so remove & delete the LRU object i.e Object inserted
// in key = 0 and insert the object with key=2.
// Now cache will have 2 items (objects with key 1 and key 2) only
key=2;
QString *itemToInsert3 = new QString;
*itemToInsert2 = "File3";
cache.insert(key,itemToInsert3,50);
```


(C) Copyright Nokia 2013. All rights reserved.

(C) Copyright Nokia 2013. All rights reserved.