Subject: Re: Issues using 2015.2 version
Posted by mr_ped on Wed, 27 Jan 2016 21:42:21 GMT
View Forum Message <> Reply to Message

Do you really want to pass full copy?

I think for example for "Save" a const reference is much better, i.e.:
int SaveMasterRecord(const Vector<String> & t);

About the sort... you probably want to keep "pairs" intact?
Then you can again use const reference as input for the function.

But maybe you should check for the built-in Sort, like this:

```
Vector<int> pairs{1, 4, 2, 3};
Vector<int> sortedPairs = clone(pairs);
Sort(sortedPairs, [](const int & a, const int & b){return a < b;} );
// lambda expression used for simple "a is less than b" predicate here
// to give you idea, how you can code your own comparator for "Pair" type
//(the "less than" is default predicate of Upp::Sort(T & container); )
```

So if you would define operator < in Pair class, you would be able to write:

```
Vector<Pair> sortedPairs = clone(pairs);
Sort(sortedPairs);
```

Anyway, passing full copy of Vector to function does make little sense to me, only when you actually store the parameter somewhere... but in most cases you want either just to read the data (const Vector & v), or you want to
manipulate them, but the owner of the Vector is still the caller, so you want to operate on his instance: (Vector & v);