

---

Subject: Re: Jsonize problems with maps  
Posted by [mirek](#) on Thu, 19 Apr 2012 07:20:20 GMT  
[View Forum Message](#) <> [Reply to Message](#)

---

Mindtraveller wrote on Wed, 18 April 2012 19:03: First, I had to implement two member functions of ValueMap class, which were declared in Documentation but were actually absent: const Value& GetKey(int i) const { return data->key[i]; }  
const Value& GetValue(int i) const { return data->value[i]; }

Just for explanation:

They are not present in "legacy" Value implementation. Documentation is about new "SVO Value".

I still did not had balls to finally switch SVO Value as default (it was delayed as I have found a subtle bug a month ago)... but it is definitely coming before the end of May.

So, here is the patch: template <class T, class K, class V>

```
void JsonizeStringMap(JsonIO& io, T& map)
{
    if(io.IsLoading()) {
        map.Clear();
        const ValueMap& va = io.Get();
        map.Reserve(va.GetCount());
        for(int i = 0; i < va.GetCount(); i++) {
            Value vv = va[i];
            K key;
            V value;
            LoadFromJsonValue(key, va.GetKey(i));
            LoadFromJsonValue(value, va.GetValue(i));
            map.Add(key, value);
        }
    }
    else {
        Index<Value> index;
        Vector<Value> values;
        index.Reserve(map.GetCount());
        values.Reserve(map.GetCount());
        for (int i=0; i<map.GetCount(); ++i)
        {
            index.Add(StoreAsJsonValue(map.GetKey(i)));
            values.Add(StoreAsJsonValue(map[i]));
        }
        ValueMap vm(index, values);
        io.Set(vm);
    }
}
```

```

template <class K, class V, class H>
void Jsonize(JsonIO& io, VectorMap<K, V, H>& map, bool)
{
    JsonizeStringMap<VectorMap<K, V, H>, K, V>(io, map);
}

```

```

template <class K, class V, class H>
void Jsonize(JsonIO& io, ArrayMap<K, V, H>& map, bool)
{
    JsonizeStringMap<ArrayMap<K, V, H>, K, V>(io, map);
}

```

Here is simple demo:#include <Core/Core.h>  
using namespace Upp;

```

struct TestStruct
{
    struct TestV : Moveable<TestV>
    {
        int a;
        int b;
        String ToString() const
        {
            return Format("a=%d, b=%d", a,b);
        }
    }

    void Jsonize(JsonIO &json)
    {
        json
            ("a", a)
            ("b", b)
        ;
    }
};

void Add()
{
    TestV v;
    v.a = Random(100);
    v.b = Random(100);
    map.AddPick(FormatIntHex(Random() ^ (int) GetTickCount()), v);
}

void Jsonize(JsonIO &json)
{
    //Upp::Jsonize(json,map); // <- default
}

```

```
::Jsonize(json,map,true); // <- string map
}
VectorMap<String,TestV> map;
};
```

CONSOLE\_APP\_MAIN

```
{
    TestStruct test, test2;
    test.Add();
    test.Add();
```

```
    LoadFromJson(test2, StoreAsJson(test));
```

```
    Cout() << StoreAsJson(test) << "\n=====\\n" << StoreAsJson(test2)
    << "\\n\\n";
}
[/quote]
```

Applied, only I do not really see the point of Jsonize with another bool parameter, that is why I have renamed it to StringMap. Thanks!

Mirek