Subject: Esc: maps nested in arrays and vice versa - underdevelopment or a bug?
Posted by fudadmin on Thu, 11 May 2006 23:57:52 GMT
View Forum Message <> Reply to Message

Esc: maps nested in arrays and vice versa - underdevelopment or a bug?

Subject: Re: Esc: maps nested in arrays and vice versa - underdevelopment or a bug?
Posted by fudadmin on Fri, 12 May 2006 03:55:35 GMT
View Forum Message <> Reply to Message

fudadmin wrote on Fri, 12 May 2006 00:57Esc: maps nested in arrays and vice versa - underdevelopment or a bug?

In fact, I should have named this topic: "Don't mess your your maps with unquoted keys!"
When porting one of my Dialect programs I encountered an unexpected behaviour with lot of "void"'s in my complex arrays-maps structure. Afterwards I realized that Esc requires map keys to be all quoted in such complex array cases. But in Ultimate's++ home page manual:

{ "alfa":10, "beta":20, 20:"40" }

20 is unquoted. When is the creation of new voids expected?

This is a piece of the structure:

```
menus=[[{
 menu:"MainMenu",
 submenus:[
   [{ menu:"View", name:"ViewLonG" , img:"Ctrl", key:"CtrL_V",
      submenus:[
      [{ menu:"View1" }],
      [{ menu:"View2",
            submenus:[
            [{ menu:"View21", name:"View21Long" }],
            [{ menu:"View22" }]
                  ]
       }]
             ]
   }],
   [{ menu:"File",
      submenus:[
      [{ menu:"New", name:"New File...", img:"Ctrl" }],
      [{ menu:"Open" }],
      [{ menu:"Save" }],
      [{ menu:"SaveAs" }],
```

```
    [{ menu:"Quit" }]
        ],
    end:"end"
  }],
....
```

Subject: Re: Esc: maps nested in arrays and vice versa - underdevelopment or a bug?
Posted by mirek on Fri, 12 May 2006 06:30:29 GMT
View Forum Message <> Reply to Message

fudadmin wrote on Thu, 11 May 2006 23:55fudadmin wrote on Fri, 12 May 2006 00:57Esc: maps nested in arrays and vice versa - underdevelopment or a bug?

In fact, I should have named this topic: "Don't mess your your maps with unquoted keys!"
When porting one of my Dialect programs I encountered an unexpected behaviour with lot of "void"'s in my complex arrays-maps structure. Afterwards I realized that Esc requires map keys to be all quoted in such complex array cases. But in Ultimate's++ home page manual:

{ "alfa":10, "beta":20, 20:"40" }

20 is unquoted. When is the creation of new voids expected?

Hm, strange. I guess it should work (it did...). Are you awere that above can be written as

```
{ [ 'a', 'l', 'f', 'a' ] : 10, ['b', 'e', 't', 'a'] : 20,
  20 : ['4', '0'] }
```

?

In other words, first two keys are arrays, last key is single number.

Could not that be the source of trouble?

Mirek

Subject: Re: Esc: maps nested in arrays and vice versa - underdevelopment or a bug?
Posted by fudadmin on Fri, 12 May 2006 13:58:06 GMT
View Forum Message <> Reply to Message

ok, here is a simple example:

```
menus1={  menu :"MainMenu",  submenus :"MainSubmenu1" };
menus2={ "menu":"MainMenu", "submenus":"MainSubmenu1" };

x<<to_string(menus1)<<"\n";
x<<to_string(menus1)<<"\n";
```

OUTPUT:

```
{ void:"MainSubmenu1" }
{ "menu":"MainMenu" }, "submenus":"MainSubmenu1" }
```

Is this normal?
And I can't understand the reason for one more extra curly bracket in the middle in the case with quoted keys (menu2)...

Subject: Re: Esc: maps nested in arrays and vice versa - underdevelopment or a bug?
Posted by mirek on Fri, 12 May 2006 15:22:38 GMT
View Forum Message <> Reply to Message

fudadmin wrote on Fri, 12 May 2006 09:58ok, here is a simple example:

```
menus1={  menu :"MainMenu",  submenus :"MainSubmenu1" };
menus2={ "menu":"MainMenu", "submenus":"MainSubmenu1" };

x<<to_string(menus1)<<"\n";
x<<to_string(menus1)<<"\n";
```

OUTPUT:

```
{ void:"MainSubmenu1" }
{ "menu":"MainMenu" }, "submenus":"MainSubmenu1" }
```

Is this normal?

Yes. First 'menu' is interpreted as value of variable 'menu'. As long as you have not assigned

something to 'menu', it has void value.

Quote:
And I can't understand the reason for one more extra curly bracket in the middle in the case with quoted keys (menu2)...

Actually, this one is most likely bug in to_string routine....

...now fixed (just misplaced r << " }"). My apologies. And thanks for finding that bug.

Mirek

Subject: Re: Esc: maps nested in arrays and vice versa - underdevelopment or a bug?
Posted by fudadmin on Fri, 12 May 2006 18:36:52 GMT
View Forum Message <> Reply to Message

But logical output for unquoted map keys should be:

{ void :"MainMenu", void :"MainSubmenu1" };

Why all the rest of map members after one "void" are killed?

Subject: Re: Esc: maps nested in arrays and vice versa - underdevelopment or a bug?
Posted by fudadmin on Fri, 12 May 2006 19:02:00 GMT
View Forum Message <> Reply to Message

fudadmin wrote on Fri, 12 May 2006 19:36But logical output for unquoted map keys should be:

{ void :"MainMenu", void :"MainSubmenu1" };

Why all the rest of map members after one "void" are killed?

I've just checked once again with more members. Actually they are all killed "before" (starting from the first .... except the last one in my case).

Subject: Re: Esc: maps nested in arrays and vice versa - underdevelopment or a bug?
Posted by mirek on Fri, 12 May 2006 20:31:23 GMT
View Forum Message <> Reply to Message

fudadmin wrote on Fri, 12 May 2006 15:02fudadmin wrote on Fri, 12 May 2006 19:36But logical output for unquoted map keys should be:

{ void :"MainMenu", void :"MainSubmenu1" };

Why all the rest of map members after one "void" are killed?

I've just checked once again with more members. Actually they are all killed "before" (starting from the first .... except the last one in my case).

Of course they are. This is not multimap and they all have the same key value - void...

Mirek

Subject: Re: Esc: maps nested in arrays and vice versa - underdevelopment or a bug?
Posted by fudadmin on Fri, 12 May 2006 21:36:26 GMT
View Forum Message <> Reply to Message

Quote:
This is not multimap and they all have the same key value - void...

I had been using these kind of structures for 5 years with Dialect interpreter but I have never imagined that someone will try to invent square wheels... which also means you can't drive if one member in a big structure doesn't have value.

Nevertheless, are these structures both maps or none, or only the second one?

menus1={ menu :"MainMenu", submenus :"MainSubmenu1" };
menus2={ "menu":"MainMenu", "submenus":"MainSubmenu1" };

How many keys and values are in each of them?

And does it mean that you can have only one key in a map???
And otherwise you have to create a multimap(what's this?)?

Subject: Re: Esc: maps nested in arrays and vice versa - underdevelopment or a bug?
Posted by mirek on Fri, 12 May 2006 21:54:49 GMT
View Forum Message <> Reply to Message

fudadmin wrote on Fri, 12 May 2006 17:36Quote:
This is not multimap and they all have the same key value - void...

I had been using these kind of structures for 5 years with Dialect interpreter but I have never imagined that someone will try to invent square wheels...

Actually, this is the same as maps in any other scripting language. (Now I wonder how look Dialect maps like

Quote:
Nevertheless, are these structures both maps or none, or only the second one?

```
menus1={  menu :"MainMenu",  submenus :"MainSubmenu1" };
menus2={ "menu":"MainMenu", "submenus":"MainSubmenu1" };
```

How many keys and values are in each of them?

Sure, if they are initialize as { x:y, ... }, they are maps.

Quote:
And does it mean that you can have only one key in a map???

You can have one value per one key. In other words, each key has unique value.

In this example, for menus1, value of menu and submenus is void. Therefore, your initiazation could also be written as

```
menus1[void] = "MainMenu";
menus1[void] = "MainSubmenu1";

menus2["menu"] = "MainMenu";
menus2["submenus"] = "MainSubmenu1";
```

I hope this helps...

BTW, there is also other equivalent "structure" notation for menus2:

menus2.menu = "MainMenu";
menus2.submenu = "MainSubmenu1";

Quote:
And otherwise you have to create a multimap(what's this?)?

Well, I used STL termnilogy here. std::map is map like this (unique keys). There is also std::multimap - in that case, keys are not unique - you can have more keys with the same value in the map (but then single operator[] is obviously not enough for dealing with such ADT).

Mirek

Subject: Re: Esc: maps nested in arrays and vice versa - underdevelopment or a bug?
Posted by fudadmin on Fri, 12 May 2006 22:23:17 GMT
View Forum Message <> Reply to Message

luzr wrote on Fri, 12 May 2006 22:54fudadmin wrote on Fri, 12 May 2006 17:36Quote:
This is not multimap and they all have the same key value - void...

I had been using these kind of structures for 5 years with Dialect interpreter but I have never imagined that someone will try to invent square wheels...

Actually, this is the same as maps in any other scripting language. (Now I wonder how look Dialect maps like

...
Quote:
Frames

Frames store data by equating a string, which becomes the frame’s localized key, to a value. Thus a frame is essentially a storage system of key-value pairs, often called "hash tables" or "dictionaries". A frame key must be a string (ex. x or myVariable), while the value can be of any data type. Frames are created using curly braces, {}. To set the variable x to an empty frame: x = {}. To create a frame with two initial key-value pairs, also known as slots, separate the values by commas (ex. myFrame = {age:27, eyeColor:"Brown"}, or, myFrame = {"age":27, "eyeColor":"Brown"}). Values are retrieved from a frame using either the dot or bracket operator. Thus both

x = myFrame.age and x = myFrame["age"] yield the value 27.

If a key is referenced that doesn't contain a value, the result will be nil. Once a frame has been created, new slots can be added using the same operators: myFrame.hairColor = "Blonde", or, myFrame["hairColor"] = "Blonde". To completely remove a slot fro m a frame, use the remove function. For example: remove(myFrame, "hairColor")

Two frames can be combined using the concatenation operator. If a key exists in both frames, then the value contained in the right operand is used in the result. For example: {a:1, b:2} ~ {b:3, c:4} results in the frame {a:1, b:3, c:4}.

When evaluated as a Boolean value, all frames return true.

Subject: Re: Esc: maps nested in arrays and vice versa - underdevelopment or a bug?
Posted by fudadmin on Fri, 12 May 2006 22:38:45 GMT
View Forum Message <> Reply to Message

Anyway, the difference in Ultimate++ is:
"Don't mess up your maps (and structures) with unquoted keys"...
When you accept that as a fact you can use them a little...

Subject: Re: Esc: maps nested in arrays and vice versa - underdevelopment or a bug?
Posted by mirek on Sat, 13 May 2006 05:37:43 GMT
View Forum Message <> Reply to Message

fudadmin wrote on Fri, 12 May 2006 18:38Anyway, the difference in Ultimate++ is:
"Don't mess up your maps (and structures) with unquoted keys"...
When you accept that as a fact you can use them a little...

Well, actually, the difference is that in Dialect, keys must be strings and identifiers in initialization list seems to be converted to strings. In Esc, keys can be anything, and in initialization list, both key and value are evaluated as expressions.

Sorry for being ortoghonal

Mirek

Subject: Re: Esc: maps nested in arrays and vice versa - underdevelopment or a bug?
Posted by fudadmin on Sat, 13 May 2006 12:03:52 GMT
View Forum Message <> Reply to Message

luzr wrote on Sat, 13 May 2006 06:37fudadmin wrote on Fri, 12 May 2006 18:38Anyway, the difference in Ultimate++ is:
"Don't mess up your maps (and structures) with unquoted keys"...
When you accept that as a fact you can use them a little...

Well, actually, the difference is that in Dialect, keys must be strings and identifiers in initialization list seems to be converted to strings. In Esc, keys can be anything, and in initialization list, both key and value are evaluated as expressions.

Sorry for being ortoghonal

Mirek

Well, in Dialect you can use symbol types as well ('symbol).

But - no need to be sorry - in Esc the instant maps evaluation gives more power to a programmer - the feature I've been missing for years in Dialect...

Just need to know how to switch from "square wheels" to "turbo"...