Subject: Re: Should RGBA have got 4 arguments constructor?
Posted by Klugier on Thu, 24 Jul 2014 20:32:08 GMT
View Forum Message <> Reply to Message

Hello Mirek,

We can easily implement RGBA constructor in C++11 branch... This standard changes a little bit POD rules. Now we can define constructor, but firstly we need to tell compiler that default constructor exists. I enclose demonstrative code:

```
#include <Core/Core.h>

#include <type_traits>

using namespace Upp;

namespace UppCpp11 {
struct RGBA {
 RGBA() = default;
 RGBA(byte r, byte g, byte b, byte a) {
  this->r = r;
  this->g = g;
  this->b = b;
  this->a = a;
 }

 byte b, g, r, a;
};

struct NotPodRGBA {
 NotPodRGBA(byte r, byte g, byte b, byte a) {
  this->r = r;
  this->g = g;
  this->b = b;
  this->a = a;
 }

 byte b, g, r, a;
};
}

CONSOLE_APP_MAIN
{
 UppCpp11::RGBA color(255, 255, 255, 255);
 Cout() << "Is RGBA POD? " << (std::is_pod<UppCpp11::RGBA>::value ? "True" : "False") <<
"!\n";
 Cout() << "Is NotPodRGBA POD? " << (std::is_pod<UppCpp11::NotPodRGBA>::value ? "True" :
```

"False") << "!\n";
}

Result:

Is RGBA POD? True!
Is NotPodRGBA POD? False!

More information you can find on:
http://stackoverflow.com/questions/4178175/what-are-aggregates-and-pods-and-how-why-are-they-special/7189821#7189821 - second answer.

P.S.
Tested with g++ 4.8 using std=c++11 flag.

Sincerely,
Klugier
