

---

Subject: LoadFromFile, StoreToFile and Serialize  
Posted by [281264](#) on Wed, 29 Sep 2010 10:39:33 GMT  
[View Forum Message](#) <> [Reply to Message](#)

---

Hi,

I have a simple query regarding how the referred functions work. I am attaching an easy example with the objective to save and retrieve the last size and pos of the main window. But it does not work. Why?

Thank you.

Cheers,

Javier

---

#### File Attachments

1) [prueba\\_crear\\_proyecto.7z](#), downloaded 270 times

---

---

Subject: Re: LoadFromFile, StoreToFile and Serialize  
Posted by [dolik.rce](#) on Wed, 29 Sep 2010 11:42:07 GMT  
[View Forum Message](#) <> [Reply to Message](#)

---

Hi Javier,

You actually have it all correct, there is only mistake in your main. You create temporary instance of the window when calling each function, so it works on that, but it appears like doesn't work because Run() is called on another instance, which did not get the deserialized data.

To fix it, just create a variable for the window. It should look like this:

```
GUI_APP_MAIN
{
    prueba_crear_proyecto crear_proyecto;
    LoadFromFile(crear_proyecto, ConfigFile());
    crear_proyecto.Run();
    StoreToFile(crear_proyecto, ConfigFile());
}
```

Best regards,  
Honza

---

---

Subject: Re: LoadFromFile, StoreToFile and Serialize  
Posted by [281264](#) on Wed, 29 Sep 2010 11:59:18 GMT

---

Honza,

Thank you.

Permit me to ask some more questions:

1.- what is the actual difference between creating an object vs working with a temporary instance? Are there other more meaningful implications? What do you do in your applications?

best choice? Are there other options available?

Many thanks.

Cheers,

Javier

---

Subject: Re: LoadFromFile, StoreToFile and Serialize  
Posted by [dolik.rce](#) on Wed, 29 Sep 2010 15:33:30 GMT  
[View Forum Message](#) <> [Reply to Message](#)

---

281264 wrote on Wed, 29 September 2010 13:59:1.- what is the actual difference between creating an object vs working with a temporary instance? Are there other more meaningful implications? What do you do in your applications?

Well, in both cases you create an object, the only difference is the syntax. E.g.:  
`class something{  
 something(){...}  
 ...  
}`

```
main(){  
    something s; //this creates variable of type something (the call to the constructor is implicit)  
    something(); //this is like calling a function, which returns object of type something  
    //(or you can think of anonymous object - it exist, you just don't know how is it called)  
    something t=something(); //you can either store it in variable of appropriate type  
    something().Run(); //or call some of its methods (but in this case you didn't save  
    //the reference to the variable anywhere so you have no way to access it later)  
}I personally use whatever is shorter. In simple cases, where I just need to create the window and  
Run() it then "mywin().Run()" is totally sufficient. But when I need to access it more times, then I  
have to use the longer syntax: "mywin mw; mw.SetSomething();mw.Run();..."
```

data/config/etc..is the combination of mentioned functions the best choice? Are there other options available? There are many options. Serialization is IMHO the simplest, as it requires only to implement Serialize(), which is very simple thanks to overloaded operator %. What is actually the

best choice depends a lot on what you intend to do. For example to store window position or state of the application (e.g. what document is opened), the binary serialization is great. But if you want to save configuration, it is a good idea to make it human readable. For this you can either use Xmlize functions available in U++, or if you prefer plain text there are some classes to help you with this here on the forum. The final decision what format to use is always up to the designer of the application

Honza

---

---

Subject: Re: LoadFromFile, StoreToFile and Serialize  
Posted by [281264](#) on Wed, 29 Sep 2010 16:07:50 GMT  
[View Forum Message](#) <> [Reply to Message](#)

---

Fine, as always.

Many thanks.

Cheers,

Javier

---