Subject: Skylark - serving CSS style sheets
Posted by mdelfede on Mon, 15 Jul 2013 06:20:44 GMT
View Forum Message <> Reply to Message

Hi,

I'm working with SCGI, so I'd prefere to serve ALL of my files directly from skylark and not let the webserver do it.
How can this be done ?
Is it possible to "insert" somehow the file inside compiled app and serve it directly, without saving to disk before as is done for witz templates ?

Subject: Re: Skylark - serving CSS style sheets
Posted by dolik.rce on Mon, 15 Jul 2013 12:17:14 GMT
View Forum Message <> Reply to Message

mdelfede wrote on Mon, 15 July 2013 08:20Hi,

I'm working with SCGI, so I'd prefere to serve ALL of my files directly from skylark and not let the webserver do it.
How can this be done ?
Hi,

Have a look at Skylark/static.icpp. There is a handler which can be used to serve styles, javascripts and images. Also, it contains some support for adding links to witz templates that allow for aggressive (configurable) usage of browser caches.
mdelfede wrote on Mon, 15 July 2013 08:20Is it possible to "insert" somehow the file inside compiled app and serve it directly, without saving to disk before as is done for witz templates ?
Any file can be included into executable using *.brc file. Then you can serving simply by passing the data to Http::Content().

Best regards,
Honza

Subject: Re: Skylark - serving CSS style sheets
Posted by mdelfede on Mon, 15 Jul 2013 12:22:31 GMT
View Forum Message <> Reply to Message

Hi, Honza,

thank you very much for answeriny.
I already use somewhere the .brc files, but I think that it would be better to have some more 'smart' way to handle styles, javascript and image files, without having to do manually all the .brc mechanics.

Ciao

Max

Subject: Re: Skylark - serving CSS style sheets
Posted by dolik.rce on Mon, 15 Jul 2013 19:58:26 GMT
View Forum Message <> Reply to Message

mdelfede wrote on Mon, 15 July 2013 14:22Hi, Honza,

thank you very much for answeriny.
I already use somewhere the .brc files, but I think that it would be better to have some more 'smart' way to handle styles, javascript and image files, without having to do manually all the .brc mechanics.

Ciao

Max

Manually? How can it be any easier than this:
BINARY_MASK(scripts, "js/*.js")
BINARY_MASK(styles, "css/*.css")
BINARY_MASK(images, "img/*.jpg")

Honza

Subject: Re: Skylark - serving CSS style sheets
Posted by mdelfede on Mon, 15 Jul 2013 20:48:55 GMT
View Forum Message <> Reply to Message

Ah! Never used BINARY_MASK.... I'll try it !

Thank you Honza!

Max

Subject: Re: Skylark - serving CSS style sheets
Posted by Zbych on Tue, 16 Jul 2013 06:31:35 GMT
View Forum Message <> Reply to Message

dolik.rce wrote on Mon, 15 July 2013 21:58

Manually? How can it be any easier than this:
BINARY_MASK(scripts, "js/*.js")
BINARY_MASK(styles, "css/*.css")
BINARY_MASK(images, "img/*.jpg")

It would be nice to extend BINARY macro the same way that IMAGE_ID works (to save also name of binary data).
With images you can use GetImlImage(name) to find image by it's name.

Subject: Re: Skylark - serving CSS style sheets
Posted by mdelfede on Tue, 16 Jul 2013 06:33:57 GMT
View Forum Message <> Reply to Message

It does already.... you've an array of names too.

Max

Subject: Re: Skylark - serving CSS style sheets
Posted by Zbych on Tue, 16 Jul 2013 07:04:05 GMT
View Forum Message <> Reply to Message

Yes, you are right.

Subject: Re: Skylark - serving CSS style sheets
Posted by mdelfede on Tue, 16 Jul 2013 07:06:47 GMT
View Forum Message <> Reply to Message

This is my code to serve .css coming from .brc :

```
SKYLARK(Styles, "styles/*")
{
 static VectorMap<String, String> cssMap;
 static bool init = false;

 if(!init)
 {
  for(int i = 0; i < styles_count; i++)
   cssMap.Add(styles_files[i], String(styles[i], styles_length[i]));
  init = true;
 }
```

```
 int i = cssMap.Find(http[0]);
 if(i >= 0)
  http.Content("text/css", cssMap[i]);
}
```

Subject: Re: Skylark - serving CSS style sheets
Posted by Zbych on Tue, 16 Jul 2013 07:29:13 GMT
View Forum Message <> Reply to Message

mdelfede wrote on Tue, 16 July 2013 09:06This is my code to serve .css coming from .brc :

```
SKYLARK(Styles, "styles/*")
{
 static VectorMap<String, String> cssMap;
 static bool init = false;

 if(!init)
 {
  for(int i = 0; i < styles_count; i++)
   cssMap.Add(styles_files[i], String(styles[i], styles_length[i]));
  init = true;
 }

 int i = cssMap.Find(http[0]);
 if(i >= 0)
  http.Content("text/css", cssMap[i]);
}
```

Since Skylark is multi thread, your init section is risky. You should use ONCELOCK.

Subject: Re: Skylark - serving CSS style sheets
Posted by mdelfede on Tue, 16 Jul 2013 07:31:10 GMT
View Forum Message <> Reply to Message

Right

Subject: Re: Skylark - serving CSS style sheets

Posted by Zbych on Tue, 16 Jul 2013 07:33:26 GMT
View Forum Message <> Reply to Message

And VectorMap is also not thread safe, so you should add INTERLOCK when you are accessing it.

Subject: Re: Skylark - serving CSS style sheets
Posted by mdelfede on Tue, 16 Jul 2013 07:35:54 GMT
View Forum Message <> Reply to Message

Zbych wrote on Tue, 16 July 2013 09:33And VectorMap is also not thread safe, so you should add INTERLOCK when you are accessing it.

Are you sure ???? I'm using VectorMap everywhere, and I think it should be safe reading it, of course NOT writing it, without INTERLOCK. Otherwise, I should spread interlocks in tons of code lines....

Subject: Re: Skylark - serving CSS style sheets
Posted by Zbych on Tue, 16 Jul 2013 07:45:21 GMT
View Forum Message <> Reply to Message

I am not sure, we have to ask someone with greater experience. But to be save I usually use a template that adds locking to a container and I don't have to do it manually.

Subject: Re: Skylark - serving CSS style sheets
Posted by mirek on Mon, 22 Jul 2013 20:31:49 GMT
View Forum Message <> Reply to Message

Zbych wrote on Tue, 16 July 2013 03:45I am not sure, we have to ask someone with greater experience. But to be save I usually use a template that adds locking to a container and I don't have to do it manually.

Most objects in U++ allow many readers and single writer, which is sort of sanity standard with multithreading anyway (here refering to other frameworks...)

Mirek