Subject: Some questions about witz
Posted by dolik.rce on Sun, 26 Aug 2012 05:46:54 GMT
View Forum Message <> Reply to Message

Hi Mirek,

Welcome back from the holiday, I hope you enjoyed it

I finally got time to play with skylark for a while and I found couple things around witz that seem a bit impractical to me...

1) If id is encountered for which there is corresponding #define, it is replaced by empty text (quoted from witz docs). Is this by design to make things simpler or is it just arbitrary decision that can be changed? I think there is many situations where you want to use the same fragment many times. For example:<html>
<head>
<title>#TITLE</title>
</head>
<body>
<h1>#TITLE</h1>
some text
</body>
</html>

#define TITLE Skylark testFor me this code doesn't even behave as described in tutorial, the h1 element is not empty, but it renders as literal string "#TITLE".

2) On similar note, replacing unknown defines by empty string is making it hard to work with inlined css. Also, the results are somewhat inconsistent. Example:<style type="text/css">
.style1 {background-color: #8888FF;}
.style2 {background-color: #FF8888;}
#content {position: absolute; left: 150px;}
</style>renders as<style type="text/css">
.style1 {background-color: #8888FF;}
.style2 {background-color: ;}
 {position: absolute; left: 150px;}
</style> As you can see, where the define is valid "C identifier", it is replaced by empty string, otherwise it stays. This can be worked around by using external styles, but that makes debugging harder, since browser tend to heavily cache those. Also, when you want to use witz to generate the styles, this might be problem. Handling colors can worked around by using rgb(255,128,128) syntax, resulting in more writing, but working CSS. Setting style to element with given id, e.g. #content, is probably not possible at all. What about just leaving any unknown substitution in the text as is?

3) Expresions like map[key] are said to work in witz tutorial, but as far as I can tell it is not implemented yet.

4) In dot notation "value.field", it is not possible to use fields that are not valid C identifiers. As a result, you can not directly access values in variable corresponding to e.g. VectorMap<int,String>, because "map.0" makes the parser fail. I know that in this syntax it looks weird, but since the map[0] doesn't work, it is all we got. I think that the CParser would need a new method for this, something like ReadWord(), which would be used instead of ReadId() and would read all continuous non-whitespace characters, including special symbols etc.

I hope this is not too much criticism at once It just accumulated in me over the last week as I waited till you get back

Best regards,
Honza
