U++ - Feature #1028
Function-like templates

03/27/2015 10:16 PM - Jan Dolinár

Status:	Approved	**Start date:**	03/27/2015
Priority:	Low	**Due date:**	
Assignee:	Miroslav Fidler	**% Done:**	0%
Category:	Skylark	**Estimated time:**	0.00 hour
Target version:		**Spent time:**	0.00 hour

Description

Several times lately, I found myself in a situation when I needed to render similar code in witz template multiple times, but with slightly different visuals or content. To do it easily, I wrote myself this little witz function:

```
Value Render(const Vector<Value>& arg, const Renderer *r) {
  if (arg.GetCount() < 1)
    return "";
  // create new Renderer, as we can not modify the one currently used
  Renderer rr;
  // copy variables so they are avalable in the rendered template too
  for(int i = 1; i < r->Variables().GetCount(); i++) {
    rr(r->Variables().GetKey(i), r->Variables()[i]);
  }
  // add arguments as variables
  for(int i = 1; i < arg.GetCount(); i++) {
    rr("_"+IntStr(i), arg[i]);
  }
  // render the template passed in first argument
  return rr.Render(AsString(arg[0]));
}

INITBLOCK {
  Compiler::Register("render", Render);
};
```

The usage is simple, I just have a template which uses $_1, $_2, etc. for the variable arguments and in the main template I call it as a function, e.g. $render("MyApp/mytable.witz", "MyHeader", $CONF.TABLE1_BGCOLOR, $TABLE1_CONTENT).

I know this can probably be done using includes and defines in witz, but this way it feels much more natural and also the syntax is much simpler to write. Also, the function might be simpler and more efficient if implemented directly in Skylark, it might be possible to avoid creating brand new renderer or at least copy the variables more efficiently.

If you think this might be of interest to other people, you can add it to the Skylark/stdlib.icpp. If you think it's stupid idea, just close this issue :)

History

#1 - 05/04/2015 01:25 PM - Miroslav Fidler

- Status changed from Ready for CR to Approved