µWeb - Feature #574

model to json output

2011-12-29 15:10 - Jan Klopper

Status:	Closed	**Start date:**	2011-12-29
Priority:	Normal	**Due date:**	
Assignee:	Elmer de Looff	**% Done:**	100%
Category:	Model	**Estimated time:**	0.00 hour
Target version:		**Spent time:**	3.00 hours

Description

The current model doesn't allow the user to call the cjson encode function as there's to many object types that the encode funcion cannot handle.
Writing a separate toJson function seems weird, as we're probably going to be outputting obejcts from the model to the browser in json loads of times.

At the moment, the datatime, and Record class don't play nice, wrapping them in dict and str calls helps, but this needs to be done recursively.

History

#1 - 2011-12-30 15:29 - Elmer de Looff

- Category set to Model

- Status changed from New to Resolved

- Assignee changed from Elmer de Looff to Jan Klopper

- % Done changed from 0 to 70

Did a bit of JSON performance testing last night, turns out that both cjson and simplejson are mad fast, json performance being an order of magnitude behind.

cjson lacks capability to convert unknown types to JSON format, whereas simplejson does support this, so choice has been made to go with simplejson for reasons that we can convert date and time ad-hoc using a Default function.

The model has gained two module-level functions:

RecordToDict

Which takes three arguments:

- record The record to convert to a dictionary
- complete (default: False), flags whether all the foreign relations of the record should be resolved prior to converting it to a dictionary
- recursive (default: False), flags whether foreign relations should have **their** foreign relations resolved. This only functions if complete is set to True.

MakeJson

Which takes a single dictionary as input and returns a JSON representation of it. datetime.datetime, datetime.time, and datetime.date are converted to string representations using strftime.

#2 - 2012-01-18 14:46 - Jan Klopper

- Status changed from Resolved to Feedback

- Assignee changed from Jan Klopper to Elmer de Looff

Using it feels touchy as I keep getting errors which tell me nothing at all (and does not show the pagemakers debugging output but a mod_python error).

Traceback (most recent call last):

```
File "/usr/lib/python2.6/dist-packages/mod_python/importer.py", line 1537, in HandlerDispatch
    default=default_handler, arg=req, silent=hlist.silent)


File "/usr/lib/python2.6/dist-packages/mod_python/importer.py", line 1229, in _process_target
    result = _execute_target(config, req, object, arg)


File "/usr/lib/python2.6/dist-packages/mod_python/importer.py", line 1128, in _execute_target
    result = object(arg)


File "/home/underdark/underdark/libs/uweb/__init__.py", line 117, in RequestHandler
    response = pages.InternalServerError()


File "/home/underdark/underdark/libs/uweb/pagemaker/__init__.py", line 346, in InternalServerError
    exc={'type': sys.exc_type.__name__,
```

AttributeError: 'NoneType' object has no attribute '__name__'

#3 - 2012-01-18 15:55 - Jan Klopper

- Status changed from Feedback to Closed

- % Done changed from 70 to 100

so ok, that was something else triggering the out of debugging pagemaker style errors.

looks ok now. thnx

#4 - 2012-01-18 17:33 - Elmer de Looff

Yeah, it seems logging was stealing our exception info. Or technically, it called sys.exc_clear() after processing the exception. This has been fixed in uWeb by passing the exception to be handled to the InternalServerError method, which also hands it to logging, as opposed to logging grabbing the information from sys.exc_info().