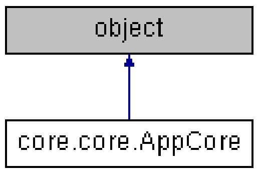
core.core.AppCore Class Reference

Inheritance diagram for core.core.AppCore:



Public Member Functions

def	**__init__**
def	init_memory
def	init_test_environnement
def	is_config_file_correct
def	get_srcarch
def	get_number_options
def	get_all_defconfig
def	get_all_topmenus_name
def	get_current_opt_name
def	get_current_opt_conditions
def	get_current_opt_value
def	get_current_opt_prompt
def	get_current_opt_help
def	get_current_opt_type
def	get_current_choice_symbols_name
def	is_current_opt_bool
def	is_current_opt_tristate
def	is_current_opt_symbol
def	is_current_opt_choice
def	is_selection_opt_choice_possible
def	is_current_opt_modifiable
def	has_option_selected
def	get_current_opt_index
def	get_current_opt_visibility
def	get_current_opt_parent_topmenu
def	get_first_option_menu
def	get_id_option_menu
def	get_id_option_name
def	get_current_opt_parent_topmenu_str
def	get_current_opt_conflict
def	is_valid_symbol

def is_choice_symbol

def get_prompt_parent_choice

def get_current_opt_verbose

def get_all_sections

def goto_search_result

def goto_opt

def goto_back_is_possible

def goto_back_opt

def goto_next_opt

def set_current_opt_value

def get_tree_representation

def search_options_from_pattern

def get_name_in_str

def get_all_symbols_condition

def finish_write_config

Public Attributes

path

arch

archs

src_arch

config_file

arch_defconfig

kconfig_infos

top_level_items

menus

top_menus

sections

items

cursor

history

Detailed Description

```
AppCore class
```

Member Function Documentation

def core.core.AppCore.finish_write_config (self,
output_file
)

```
Finish the configuration, write the .config file
```

def core.core.AppCore.get_all_defconfig (self)

```
Return all arch available into a 2D list
```

def core.core.AppCore.get_all_sections (self)

```
Return all kernel's sections into a list
```

def core.core.AppCore.get_all_symbols_condition (self)

```
docstring for get_all_symbols_condition
```

def core.core.AppCore.get_all_topmenus_name (self)

```
Return all menus name into a list
```

def core.core.AppCore.get_current_choice_symbols_name (self)

```
Return all current choice's symbols and their value into a list
[[name, value, modifiable], [name, value, modifiable] ..]
If current item is not a choice, return None or
Return False if no option is selected
```

def core.core.AppCore.get_current_opt_conditions (self)

```
Return all current symbol's conditions or
    Return False if no option is selected
```

def core.core.AppCore.get_current_opt_conflict (self)

```
Return a list of symbols which are in conflict with the current
option.
If current option is a choice, return a multi-dimensional list
of all choice's symbols' or
Return False if no option is selected
```

def core.core.AppCore.get_current_opt_help (self)

```
Return the current option's help or
    Return False if no option is selected
```

def core.core.AppCore.get_current_opt_index (self)

```
Return the current option's index
```

def core.core.AppCore.get_current_opt_name (self)

```
Return the current option's name or
    Return an empty string if no option is selected
```

def core.core.AppCore.get_current_opt_parent_topmenu (self)

```
Return the current option's first menu position or
    Return False if no option is selected
```

def core.core.AppCore.get_current_opt_parent_topmenu_str (self)

```
Return the current option's first menu position or
    Return False if no option is selected
```

def core.core.AppCore.get_current_opt_prompt (self)

```
Return the current option's prompt or
    Return False if no option is selected
```

def core.core.AppCore.get_current_opt_type (self)

```
Return the current option's type or
    Return False if no option is selected
```

def core.core.AppCore.get_current_opt_value (self)

```
Return the current option's value or
    Return False if no option is selected or
    Return False if the option is not a Symbol
```

def core.core.AppCore.get_current_opt_verbose (self)

```
Return a option's verbose output or
    Return False if no option is selected
```

def core.core.AppCore.get_current_opt_visibility (self)

```
Return the current option's visibility or
    Return False if no option is selected
```

def core.core.AppCore.get_first_option_menu (self)

```
Return the first option menu
```

def core.core.AppCore.get_id_option_menu (self, id_menu)

```
Return the 'id' option by menu
```

def core.core.AppCore.get_id_option_name (self, name)

```
Return the 'id' option by name
-1 if name doesn't exist
```

def core.core.AppCore.get_name_in_str (self, string)

```
Return a string between «...» in a string
```

def core.core.AppCore.get_number_options (self)

```
Return the number of options
```

def core.core.AppCore.get_prompt_parent_choice (self, name)

```
Return symbol choice's parent's prompt
```

def core.core.AppCore.get_srcarch (self)

```
Return the current srcarch
```

def core.core.AppCore.get_tree_representation (self)

```
Return in a structure,
the current configuration's representation
sc = symbol or choice
[sc, sc, [menu, [sc], menu, [menu, [sc]]], sc]
```

def core.core.AppCore.goto_back_is_possible (self)

```
Return if we can go back
```

def core.core.AppCore.goto_back_opt (self)

```
Goto method, go to the previous option on history's save top
Return 0 if it is done, else return -1 if it cannot be done
```

def core.core.AppCore.goto_next_opt (self)

```
Goto method, go to the next symbol option
(not menus/comment/string/hex..) which may be modified or not.
Return True is we can go to the next option
```

def core.core.AppCore.goto_opt (self, opt_id)

```
Goto method, go to the option 'opt_id'
Increment the history save
```

def core.core.AppCore.goto_search_result (self, name)

```
Goto method, go to the name's option if it exists
```

def core.core.AppCore.has_option_selected (self)

```
Return False if no option is selected
```

def core.core.AppCore.init_memory (self,
path,
arch,
src_arch,
config_file = "",
callback = None
)

```
If config_file == "", load default config
Return -1 if configuration file does not exist or is not correct
(argument or default)
```

def core.core.AppCore.init_test_environnement (self,
path
)

```
Test if the kernel path is correct
Return a 2D list with all [[[src_arch, arch]], [[src_arch, defconfig]]]
and their defconfig if it is the case, else return a error code (-1)
```

def core.core.AppCore.is_choice_symbol (self,
name
)

```
Return true if the symbol is in a choice
```

def core.core.AppCore.is_config_file_correct (self,
config_file
)

```
Return True if the config_file is correct or not
```

def core.core.AppCore.is_current_opt_bool (self)

```
Return True if current option is bool or
    Return False if no option is selected
```

def core.core.AppCore.is_current_opt_choice (self)

```
Return True if current option is a choice or
    Return False if no option is selected
```

def core.core.AppCore.is_current_opt_modifiable (self)

```
Return True if current option is modifiable or
    Return False if no option is selected
```

def core.core.AppCore.is_current_opt_symbol (self)

```
Return True if current option is a symbol or
    Return False if no option is selected
```

def core.core.AppCore.is_current_opt_tristate (self)

```
Return True if current option is bool or
    Return False if no option is selected
```

def core.core.AppCore.is_selection_opt_choice_possible (self)

```
Return True if choice_selection is modifiable
```

def core.core.AppCore.is_valid_symbol (self, name)

```
Return true if the symbol is a valid symbol (Bool, Tristate)
```

def core.core.AppCore.search_options_from_pattern (self, pattern, n, d, h)

```
Return a list of option's name found with a pattern
```

def core.core.AppCore.set_current_opt_value (self, value_user_cursor)

```
Set the current option's value with value_user_cursor ("y", "n",
    "m" (if tristate))
In case of choice, if no choice is selected : value_user_cursor == "N" or
Return False if no option is selected
```

The documentation for this class was generated from the following file:

- core.py