vfihtml Namespace ReferencePrinting functions

Data Structures

class	TimeStamp

Typedefs

typedef vfiipc::JSObject	JSObject
typedef bool(*	ScriptProcessor) (void *data, const std::string &script, std::map< std::string, std::string > &value, std::string &out, std::string &err, std::string &exitcode)
typedef bool(*	ScriptProcessorExt) (void *data, const std::string &script, std::map< std::string, std::string > &value, std::string &out, std::string &err, std::string &exitcode, ExtContext *context)
typedef std::map< std::string, std::string >	stringmap

Functions

DllSpec void	htmlSetScriptProcessor (const char *name, ScriptProcessor proc, void *data)
DllSpec void	htmlSetScriptProcessor (const char *name, ScriptProcessorExt proc, void *data=0)
DllSpec TimeStamp	operator+ (const TimeStamp &a, const TimeStamp &b)
DllSpec TimeStamp	operator- (const TimeStamp &a, const TimeStamp &b)
bool	operator== (const TimeStamp &a, const TimeStamp &b)
bool	operator!= (const TimeStamp &a, const TimeStamp &b)
bool	operator< (const TimeStamp &a, const TimeStamp &b)
bool	operator<= (const TimeStamp &a, const TimeStamp &b)
bool	operator> (const TimeStamp &a, const TimeStamp &b)
bool	operator>= (const TimeStamp &a, const TimeStamp &b)
TimeStamp	operator* (int t, TimeStamp::Unit_MS)
TimeStamp	operator* (int t, TimeStamp::Unit_S)

Typedef Documentation

? JSObject

typedef vfiipc::JSObject JSObject

? ScriptProcessor

typedef bool(* ScriptProcessor) (void *data, const std::string &script, std::map< std::string, std::string > &value , std::string &out, std::string &err, std::string &exitcode)

script processor callback

Parameters

[in]	data	transparent callback data pointer
[in]	script	script to be processed, leading and trailing whitespace has been removed.
[in]	value	name value pairs that are used as initial value of input fields, checkboxes, etc. and that are used during script processing.
[out]	out	resulting HTML code that was generated by the script. On invocation it is empty.
[out]	err	error messages that were generated during script execution (comparable to output to stderr). On invocation it is empty.
[out]	exitcode	content of parameter that was passed to exit()

Returns

true if script was successful and if the script is to be replaced by the content of *out*, false if error or exit with exit code

? ScriptProcessorExt

typedef bool(* ScriptProcessorExt) (void *data, const std::string &script, std::map< std::string, std::string > & value, std::string &out, std::string &err, std::string &exitcode, ExtContext *context)

script processor callback for ADK-GUIPRT internal processors with extended capabilities

Parameters

[in]	data	transparent callback data pointer
[in]	script	script to be processed, leading and trailing whitespace has been removed.
[in]	value	name value pairs that are used as initial value of input fields, checkboxes, etc. and that are used during script processing.
[out]	out	resulting HTML code that was generated by the script. On invocation it is empty.
[out]	err	error messages that were generated during script execution (comparable to output to stderr). On invocation it is empty.
[out]	exitcode	content of parameter that was passed to exit()

[in,out] context internal extended context

Returns

true if script was successful and if the script is to be replaced by the content of *out*, false if error or exit with exit code

? stringmap

typedef std::map<std::string,std::string> stringmap

map<string,string>

Function Documentation

? htmlSetScriptProcessor() [1/2]

DllSpec void vfihtml::htmlSetScriptProcessor (const char * *name*,
ScriptProcessor *proc*,
void * *data*
)

set script processor

Parameters

[in] name name of the scripting processor, i.e. the name immediately following '<?'

[in] proc processing function pointer, use NULL to delete a script processor.

[in] data callback data passed on as first parameter to the processing function

? htmlSetScriptProcessor() [2/2]

DllSpec void vfihtml::htmlSetScriptProcessor (const char * *name*,
ScriptProcessorExt *proc*,
void * *data* = `0`
)

set script processor

Parameters

[in] name name of the scripting processor, i.e. the name immediately following '<?'

[in] proc processing function pointer, use NULL to delete a script processor.

[in] data unused, provided just for compatibility

? operator!=()

bool vfihtml::operator!= (const TimeStamp & *a*,
const TimeStamp & *b* inline
)

compare two time values

Parameters
[in] a first time value
[in] b second time value

Returns
result of a!=b

? operator*() [1/2]

TimeStamp vfihtml::operator* (int *t*,
TimeStamp::Unit_MS inline
)

convert int to time value by multiplying with MS unit

Parameters
[in] t time in milliseconds

Returns
time value

? operator*() [2/2]

TimeStamp vfihtml::operator* (int *t*,
TimeStamp::Unit_S inline
)

convert int to time value by multiplying with S unit

Parameters
[in] t time in seconds

Returns
time value

? operator+()

DllSpec TimeStamp vfihtml::operator+ (const TimeStamp & *a*,
const TimeStamp & *b*
)

add two time values

Parameters
[in] a first time value
[in] b second time value

Returns
result of a+b

? operator-()

DllSpec TimeStamp vfihtml::operator- (const TimeStamp & *a*,
const TimeStamp & *b*
)

substract two time values

Parameters
[in] a first time value
[in] b second time value

Returns
result of a-b

? operator<()

bool vfihtml::operator< (const TimeStamp & *a*,
const TimeStamp & *b* inline
)

compare two time values

Parameters
[in] a first time value
[in] b second time value

Returns
result of a<b

? operator<=()

bool vfihtml::operator<= (const TimeStamp & *a*,
const TimeStamp & *b* inline
)

compare two time values

Parameters
[in] a first time value
[in] b second time value

Returns
result of a<=b

? operator==()

bool vfihtml::operator== (const TimeStamp & *a*,
const TimeStamp & *b* inline
)

compare two time values

Parameters
[in] a first time value
[in] b second time value

Returns
result of a==b

? operator>()

bool vfihtml::operator> (const TimeStamp & *a*,
const TimeStamp & *b* inline
)

compare two time values

Parameters
[in] a first time value
[in] b second time value

Returns
result of a>b

? operator>=()

bool vfihtml::operator>= (const TimeStamp & *a*,
const TimeStamp & *b* inline
)

compare two time values

Parameters
[in] a first time value
[in] b second time value

Returns
result of a>=b