namespace FB

FB Namespace Reference

Primary location of FireBreath classes and utility functions. More...

Namespaces

DOM
These are convenience objects used to access and manipulate **DOM** objects. They should never be created directly; instead, get the **Window** object or the **Document** object from the **BrowserHost** or create it with DOM::Element::create(obj) (or similar)

Classes

class **AsyncDrawService**
asynchronous drawing service base class. More...

class **BrowserPlugin**
Browser-specific plugin base class. More...

class **BrowserStream**
This is the abstract base class (interface class) for a browser stream. More...

class **BrowserStreamRequest**
Information about an HTTP request to be made. More...

class **DefaultBrowserStreamHandler**
Simple implementation of a stream event handler from which you can derive your own stream events handler. More...

class **FactoryBase**
This is the base factory for a plugin. Every plugin *must* implement this class and override at least the **createPlugin()** method. To futher customize your plugin, you can override other methods to replace the **PluginWindow** or the NpapiPlugin class. More...

class **PluginCore**
Base class for all FireBreath Plugins. More...

class **PluginEvent**
Plugin event base class. More...

class **AttachedEvent**
Fired when a **PluginEventSink** is attached to a **PluginEventSource** (such as a **PluginCore** derived plugin object being attached to a **PluginWindow** to get events) More...

class **DetachedEvent**
Fired when a **PluginEventSink** is detached from a **PluginEventSource** (such as a **PluginCore** derived plugin object being detached from a **PluginWindow** to get events) More...

class **ChangedEvent**
Fired when a **PluginEventSink** has changed in some fundamental way that the plugin needs to know about, such as a different drawing context being provided by the browser to a **PluginWindow**. More...

class **ResizedEvent**
Fired when the plugin window is resized. More...

class **ClipChangedEvent**
Fired when the clipping of the plugin drawing area changes. More...

class **RefreshEvent**
Fired when the plugin should repaint itself (such as on windows when WM_PAINT is received) More...

class **QuickDrawDraw**
Mac QuickDraw Draw event. More...

class **CoreGraphicsDraw**
Mac CoreGraphics Draw event. More...

class	**FocusChangedEvent** Fired when the focus changes. More...
class	**TimerEvent** Fired when a timer event is received; currently this only works on windows and is fired when WM_TIMER is received. More...
class	**KeyEvent** Fired for a key event. More...
class	**KeyUpEvent** Fired for a key up event. More...
class	**KeyDownEvent** Fired for a key down event. More...
class	**TextEvent** Used at least on Mac, possibly elsewhere; gives us text. More...
class	**MacEventCarbon** Carbon Mac event. More...
class	**MacEventCocoa** Cocoa mac event. More...
class	**MouseEvent** Fired when a mouse event occurs. More...
class	**MouseMoveEvent** Fired when the mouse moves. More...
class	**MouseButtonEvent** Fired when a mouse button event occurs. More...
class	**MouseDownEvent** Fired when a mouse down event occurs. More...
class	**MouseDoubleClickEvent** Fired when a mouse double click event occurs. More...
class	**MouseUpEvent** Fired when a mouse up event occurs. More...
class	**MouseScrollEvent** Fired when the user moves the scrollwheel. More...
class	**MouseEnteredEvent** Fired when the user moves mouse over the plugin. More...
class	**MouseExitedEvent** Fired when the user moves mouse away from the plugin. More...
class	**StreamEvent** Base class for all stream events. More...
class	**StreamCreatedEvent** This event is fired when the given stream was created. More...
class	**StreamFailedOpenEvent** This event is fired when a stream failed to open, e.g. the url was invalid or a seekable stream was requested while the server provided only a non-seekable stream. More...
class	**StreamCompletedEvent** This event is fired when a stream has completed downloading. More...
class	**WindowsEvent**

	Generic windows event. All windows events going through the winproc get first fired as a WindowEvent and then fired as the specific type, allowing you to handle them either way. More...
class	**X11Event** Generic X11 event. More...
class	**X11NativeGdkEvent** Class encapsulation for native Gdk event. More...
class	**X11NativeGdkEventExpose** Class encapsulation for native Gdk expose event. More...
class	**PluginEventSink** Plugin event sink; all objects that can receive **PluginEvent** s should inherit from this class. Most notably, **PluginCore** extends this class. More...
class	**PluginEventSource** Base class for any object, such as **BrowserStream** or **PluginWindow**, that needs to fire events to a **PluginEventSink** object (such as a **PluginCore** derived plugin class) More...
class	**PluginWindow** **PluginWindow** is the base class for all **PluginWindow** objects. More...
class	**HttpStreamResponse** Data structure to hold the response to an HTTP Get request. More...
class	**SimpleStreamHelper** Helper class for making simple HTTP requests. More...
class	**Timer** **Timer** Utility. More...
class	**TimerService** **TimerService** Utility. More...
class	**WinMessageWindow** Creates a message window. Don't touch this if you don't understand what you are doing. More...
struct	**CatchAll** When used as a parameter on a **JSAPIAuto** function this matches 0 or more variants – in other words, all other parameters from this point on regardless of type. More...
struct	**MethodInfo** Used by **FB::JSAPISimple** to store information about a method. More...
struct	**PropertyInfo** Used by **FB::JSAPISimple** to store information about a property. More...
struct	**PropertyFunctors** used by **FB::JSAPIAuto** to store property implementation details, created by **FB::make_property()**. More...
struct	**AsyncLogRequest** This class is used by **BrowserHost** for the **BrowserHost::AsyncHtmlLog** method. More...
class	**BrowserHost** Browser host base class. More...
class	**JSAPI** JavaScript API class – provides a javascript interface that can be exposed to the browser. More...
class	**scoped_zonelock** Provides a helper class for locking. More...
class	**JSAPIAuto** **JSAPI** class with automatic argument type enforcement. More...

class	**JSAPIImpl** JavaScript API base class implementation – provides basic functionality for C++ **JSAPI** objects. More...
class	**JSAPIProxy** JavaScript API Wrapper – this can wrap any type of **JSAPI** object and be passed back to any browser that. More...
class	**JSAPISimple** Simple **JSAPI** implementation for those who for whatever reason don't want to use **JSAPIAuto**. More...
struct	**script_error** Exception type; when thrown in a **JSAPI** method, a javascript exception will be thrown. More...
struct	**invalid_arguments** Thrown by a **JSAPI** object when the argument(s) provided to a SetProperty or Invoke call are found to be invalid. **JSAPIAuto** will throw this automatically if the argument cannot be convert_cast to the type expected by the function. More...
struct	**object_invalidated** Thrown by a **JSAPI** object when a call is made on it after the object has been invalidated. More...
struct	**invalid_member** Thrown when an Invoke, SetProperty, or GetProperty call is made for a member that is invalid (does not exist, not accessible, only supports Get or Set, etc) More...
class	**JSObject** Wraps a Javascript Object. More...
class	**SafeQueue** Basic thread-safe queue class. More...
class	**TypeIDMap** Bidirectional map between an identifier and a variant. More...
class	**URI** Data structure for dealing with **URI** strings. More...
struct	**bad_variant_cast** Thrown when **variant::cast<type>** or **variant::convert_cast<type>** fails because the type of the value stored in the variant is not compatible with the operation. More...
class	**variant** Accepts any datatype, used in all interactions with javascript. Provides tools for getting back out the type you put in or for coercing that type into another type (if possible). More...
class	**PluginEventMacCarbon** Mac OS X Carbon specific implementation of PluginEventMac. More...
class	**PluginEventMacCocoa** Mac OS X Cocoa specific implementation of PluginEventMac. More...
class	**PluginWindowlessWin** Windows specific implementation of **PluginWindow**. More...
class	**PluginWindowWin** Windows specific implementation of **PluginWindow**. More...
class	**PluginWindowX11** X11 specific implementation of **PluginWindow**. More...

Typedefs

typedef boost::shared_ptr < **FactoryBase** >	**FactoryBasePtr** Defines an alias representing a boost::shared_ptr<FactoryBase> More...
typedef boost::shared_ptr < **PluginCore** >	**PluginCorePtr** Defines an alias representing a boost::shared_ptr<PluginCore> More...

typedef std::vector< variant >	**VariantList** Defines an alias representing list of variants. More...
typedef std::map< std::string, variant >	**VariantMap** Defines an alias representing a string -> variant map. More...
typedef std::set< std::string >	**StringSet** Defines an alias representing a set of std::strings. More...
typedef boost::weak_ptr < FB::JSAPI >	**JSAPIWeakPtr** Defines an alias for a **JSAPI** weak_ptr (you should never use a JSAPI* directly) More...
typedef boost::shared_ptr < FB::JSAPI >	**JSAPIPtr** Defines an alias for a **JSAPI** shared_ptr (you should never use a JSAPI* directly) More...
typedef boost::weak_ptr < FB::JSObject >	**JSObjectWeakPtr** Defines an alias representing a **JSObject** weak_ptr (you should never use a JSObject* directly) More...
typedef boost::shared_ptr < FB::JSObject >	**JSObjectPtr** Defines an alias representing a **JSObject** shared_ptr (you should never use a JSObject* directly) More...
typedef boost::shared_ptr < FB::BrowserHost >	**BrowserHostPtr** Defines an alias representing a **BrowserHost** shared_ptr (you should never use a BrowserHost* directly) More...
typedef BrowserHost	**BrowserHostWrapper** Defines a alias for backwards compatibility. More...
typedef JSObject	**BrowserObjectAPI** Defines a alias for backwards compatibility. More...
typedef JSAPIPtr	**JSOutObject** Defines an alias for JSOutObject -> JSAPIPtr. More...
typedef variant(JSAPI::*	**InvokeType**)(const std::string &, const std::vector< variant > &) Defines an alias representing a function pointer to **JSAPI::Invoke**. More...
typedef variant(JSAPI::*	**ConstructType**)(const std::vector< variant > &) Defines an alias representing a function pointer to **JSAPI::Invoke**. More...
typedef void(JSAPI::*	**SetPropertyType**)(const std::string &, const variant &) Defines an alias representing a function pointer to **JSAPI::SetProperty**. More...
typedef variant(JSAPI::*	**GetPropertyType**)(const std::string &) Defines an alias representing a function pointer to **JSAPI::GetProperty**. More...
typedef void(JSAPI::*	**RemovePropertyType**)(const std::string &) Defines an alias representing a function pointer to **JSAPI::GetProperty**. More...
typedef variant(JSAPI::*	**CallMethodPtr**)(const std::vector< variant > &) Defines an alias representing a function ptr for a method on a **FB::JSAPISimple** object. More...
typedef std::map< std::string, MethodInfo >	**MethodMap** Defines an alias representing a map of methods used by **FB::JSAPISimple**. More...
typedef variant(JSAPI::*	**GetPropPtr**)()

	Defines an alias representing a function pointer for a property getter on a **FB::JSAPISimple** object. More...
typedef void(JSAPI::*	**SetPropPtr**)(const **variant** &value) Defines an alias representing a function pointer for a property setter on a **FB::JSAPISimple** object. More...
typedef std::map< std::string, **PropertyInfo** >	**PropertyMap** Defines an alias representing a map of properties used by **FB::JSAPISimple**. More...
typedef int	**SecurityZone** Used to set a SecurityZone for a method or property – used by **JSAPIAuto**. More...
typedef boost::function < **variant**(const std::vector < **variant** > &)>	**CallMethodFunctor** Defines an alias representing a method functor used by **FB::JSAPIAuto**, created by FB:: make_method(). More...
typedef std::map< std::string, MethodFunctors >	**MethodFunctorMap** Defines an alias representing a map of method functors used by **FB::JSAPIAuto**. More...
typedef boost::function < **FB::variant**()>	**GetPropFunctor** Defines an alias representing a property getter functor used by **FB::JSAPIAuto**. More...
typedef boost::function< void(const **FB::variant** &)>	**SetPropFunctor** Defines an alias representing a property setter functor used by **FB::JSAPIAuto**. More...
typedef std::map< std::string, **PropertyFunctors** >	**PropertyFunctorsMap** Defines an alias representing a map of property functors used by **FB::JSAPIAuto**. More...

Enumerations

enum	**FBKeyCode** Values that represent different keys in a platform agnostic way. More...
enum	**SecurityLevel** Default SecurityZone values; you can use these or provide your own. More...

Functions

FBKeyCode	**WinKeyCodeToFBKeyCode** (unsigned int wparam) Converts a windows key code to a FireBreath key code. More...
template<class T , class U > boost::shared_ptr< T >	**ptr_cast** (boost::shared_ptr< U > const &r) Convenience function for doing a dynamic cast of one boost::shared_ptr to another. More...
template<class API > boost::shared_ptr< API >	**get_jsapi** (const **FB::JSObjectPtr** &jso) Get a JSAPI-derived interface from a **JSObject**. More...
template<class C , typename F1 , typename F2 > **PropertyFunctors**	**make_property** (C *instance, F1 getter, F2 setter) Generate read-write property functors for use with registerProperty() of **FB::JSAPIAuto**. More...
template<class C , typename F > **PropertyFunctors**	**make_property** (C *instance, F getter) Generate read-only property functors for use with registerProperty() of **FB::JSAPIAuto**. More...
std::string	**wstring_to_utf8** (const std::wstring &src) Accepts a std::wstring and returns a UTF8-encoded std::string. More...

std::wstring	**utf8_to_wstring** (const std::string &src) Accepts a UTF8-encoded std::string and returns a std::wstring. More...
std::wstring	**wstring_tolower** (const std::wstring &src) Converts a std::wstring to lowercase. More...
template<typename Cont > **FB::VariantList**	**make_variant_list** (const Cont &cont) Create a **FB::VariantList** from any STL-style container (i.e. exposes begin() and end()) More...
template<class InputIterator > **FB::VariantList**	**make_variant_list** (InputIterator begin, InputIterator end) Create a **FB::VariantList** from the range [begin, end). More...
template<class InputIterator > void	**make_variant_list** (InputIterator begin, InputIterator end, FB::VariantList::iterator result) Fill a **FB::VariantList** with the contents of the range [begin, end). More...
template<class Cont > Cont	**convert_variant_list** (const **FB::VariantList** &v) Convert a **FB::VariantList** to STL-style container (i.e. supports value_type and back-insert-iterators). More...
template<class Cont > void	**convert_variant_list** (const **FB::VariantList** &from, Cont &to) Fill to with the contents of from, converted to Cont::value_type. More...
template<class Cont > Cont	**convert_variant_list** (FB::VariantList::const_iterator begin, FB::VariantList::const_iterator end) Convert a range of **FB::VariantList** to STL-style container (i.e. supports value_type and back-insert-iterators). More...
template<typename To , class OutputIterator > void	**convert_variant_list** (FB::VariantList::const_iterator begin, FB::VariantList::const_iterator end, OutputIterator result) Fills the range [result, result+(end-begin)) with the contents of the range [begin,end), converted to To. More...
FB::detail::VariantListInserter	**variant_list_of** (**FB::variant** v) Allows convenient creation of an **FB::VariantList**. More...
FB::VariantList	**variant_list_of** () Convenience overload to create an empty **FB::VariantList**. More...
template<typename Dict > **FB::VariantMap**	**make_variant_map** (const Dict &dict) Create a **FB::VariantMap** from any STL-style dictionary (i.e. exposes begin() and end()) More...
template<class InputIterator > **FB::VariantMap**	**make_variant_map** (InputIterator begin, InputIterator end) Create a **FB::VariantMap** from the range [begin, end). More...
template<class InputIterator > void	**make_variant_map** (InputIterator begin, InputIterator end, FB::VariantMap::iterator result) Fill a **FB::VariantMap** with the contents of the range [begin, end). More...
template<class Dict > Dict	**convert_variant_map** (const **FB::VariantMap** &v) Convert a **FB::VariantMap** to STL-style dictionary (i.e. supports value_type and insert-iterators). More...
template<class Dict > void	**convert_variant_map** (const **FB::VariantMap** &from, Dict &to)

	Fill to with the contents of from, converted to Cont::value_type. More...

template<class Dict > Dict	**convert_variant_map** (FB::VariantMap::const_iterator begin, FB::VariantMap::const_iterator end) Convert a range of **FB::VariantMap** to STL-style dictionary (i.e. supports value_type and insert-iterators). More...

template<typename To , class OutputIterator > void	**convert_variant_map** (FB::VariantMap::const_iterator begin, FB::VariantMap::const_iterator end, OutputIterator result) Fills the range [result, result+(end-begin)) with the contents of the range [begin,end), converted to To. More...

template<typename T > FB::detail::VariantMapInserter< T >	**variant_map_of** (const T &t, const **FB::variant** &v) Allows convenient creation of an **FB::VariantMap**. More...

template<typename T > std::map< T, **FB::variant** >	**variant_map_of** () Convenience overload to create an empty variant map. More...

FBKeyCode	**GDKKeyCodeToFBKeyCode** (unsigned int key) Converts a GDK key code to a FireBreath key code. More...

Detailed Description

Primary location of FireBreath classes and utility functions.

The five most important classes to understand when implementing a FireBreath plugin are:

- **FB::PluginCore**
- **FB::JSAPI** / **FB::JSAPIAuto**
- **FB::BrowserHost**
- **FB::JSObject**
- **FB::variant**