

class FB JSAPIAuto registerMethod

- Construct
- FireJSEvent
- get_valid
- getAttribute
- getMemberCount
- getMemberNames
- GetMethodObject
- GetProperty
- GetProperty
- HasMethod
- HasMethodObject
- HasProperty
- HasProperty
- Invoke
- isReserved
- JSAPIAuto
- registerAttribute
- registerMethod**
- registerProperty
- RemoveProperty
- RemoveProperty
- setAttribute
- SetProperty
- SetProperty
- setReserved
- ToString
- unregisterMethod
- unregisterProperty

```
void FB::JSAPIAuto::registerMethod ( const std::string & name,
                                   const CallMethodFunctor & func virtual
                                   )
```

Registers a method to be exposed to javascript.

To provide a method to javascript called "add" that accepts two integers and returns an integer, you need one method on your class:

```
int add(int a, int b);
```

Then in the constructor of your class that extends **JSAPIAuto** (still calling it MyPluginAPI), you register it like so:

```
registerMethod("add", make_method(this, &MyPluginAPI::add));
```

You should then be able to call the method from javascript and get a result:

```
// Assumes that this JSAPI object is the Root JSAPI for the plugin
alert(document.getElementById("plugin").add(2,765));
```

Parameters

name The name that the method will have when accessed from javascript.

func The result of a make_method call given the class instance and function ptr

Definition at line 83 of file **JSAPIAuto.cpp**.

```
84 {
85     boost::recursive_mutex::scoped_lock lock(m_zoneMutex);
86     m_methodFunctorMap[name] = func;
87     m_zoneMap[name] = getZone();
88 }
```

FB::JSAPIImpl::getZone

virtual SecurityZone getZone() const
Gets the currently active zone.

Definition: **JSAPIImpl.h:238**