

# Service Objects

## Introduction

A **Service** object creates and handles threads that help event managers dispatch events. Typically, one will use a Service Object to create threads that will call [mpiEventMgrService](#) whenever an XMP interrupt occurs. They are a convenient way to have a program automatically deal with event managers and events. Thread handling is something that is different on every operating system. Service objects may therefore have different behaviors on different operating systems. Programmers that are experienced in multi-threaded application programming will probably want to program their own threads that will call [mpiEventMgrService](#).

**NOTE:** The Service object is not part of the standard MPI. In order to use the Service Object, the file, *apputil.h* needs to be included by your code and the *apputil* library needs to be linked to your application.

## Methods

### Create, Delete, Validate Methods

[serviceCreate](#)

Create a Service for EventMgr and the threads necessary for it to run.

[serviceDelete](#)

Stop all threads belonging to the Service and deletes the Service.

### Configuration and Information Methods

[serviceEnable](#)

Enable or disables the Service.

# ServiceCreate

## Declaration

```
const Service serviceCreate(MPIEventMgr eventMgr,
                           long          priority,
                           long          sleep)
```

**Required Header:** service.h

## Description

**ServiceCreate** creates threads for each control associated with **eventMgr**, flushes **eventMgr**, and starts threads with priority that call `mpiEventMgrService(eventMgr, .)` every **sleep** milliseconds.

### priority

is a platform-specific variable.

<i>If "priority" is</i>	<i>Then</i>
-1	The operating system will choose some default priority for the service's threads. See <a href="#">Default Thread Parameters</a> table below.
>0	<i>ServiceCreate</i> will attempt to assign the priority to all of the service's threads.

### Default Thread Parameters

These are the default thread priorities (when -1 is specified).

OS	Priority
Windows	THREAD_PRIORITY_TIME_CRITICAL
RTSS	RT_PRIORITY_MAX
VxWorks	0
Linux	Maximum (defined by pthread schedule policy max priority)
Solaris	Maximum (defined by pthread schedule policy max priority)

sleep		
	<i>If "sleep" is</i>	<b>Then</b>
	-1	<i>ServiceCreate</i> will attempt to create interrupt driven threads.
	0	<i>ServiceCreate</i> will create threads that call <b>mpiEventMgrService(eventMgr,...)</b> as quickly as possible.
>0	<i>ServiceCreate</i> will create threads that attempt to call <b>mpiEventMgrService(eventMgr,...)</b> every <b>sleep</b> milliseconds.	

## Return Values

<b>handle</b>	to a Service object
<b>MPIHandleVOID</b>	if the Service could not be created

## See Also

[mpiEventMgrService](#) | [ServiceDelete](#)

# ServiceDelete

## Declaration

```
long serviceDelete(Service service)
```

**Required Header:** service.h

## Description

**ServiceDelete** alerts all threads that they should end, waits for all threads to end, and frees the memory allocated to **service**.

## Return Values

[MPIMessageOK](#)

## See Also

[ServiceCreate](#)

# ServiceEnable

## Declaration

```
long serviceEnable(Service service,
                  long   enabled)
```

**Required Header:** service.h

## Description

**ServiceEnable** enables or disables all threads belonging to Service.

<i>If "enabled" is</i>	<b>Then</b>
False	<i>serviceEnable</i> will disable <b>service</b> .
True	<i>serviceEnable</i> will enable <b>service</b> .

## Return Values

<b>handle</b>	to a Service object
<b>MPIHandleVOID</b>	if the Service could not be created

## See Also