



---

## OTP\_Mibs

Copyright © 2003-2019 Ericsson AB. All Rights Reserved.  
OTP\_Mibs 1.1.2  
July 17, 2019

---

**Copyright © 2003-2019 Ericsson AB. All Rights Reserved.**

Licensed under the Apache License, Version 2.0 (the "License"); you may not use this file except in compliance with the License. You may obtain a copy of the License at <http://www.apache.org/licenses/LICENSE-2.0> Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License. Ericsson AB. All Rights Reserved..

**July 17, 2019**

# 1 OTP\_Mibs User's Guide

---

The **OTP\_Mibs** application provides an SNMP management information base for Erlang nodes.

## 1.1 Introduction

### 1.1.1 Purpose

The purpose of the **OTP\_Mibs** application is to provide an SNMP management information base for Erlang nodes.

### 1.1.2 Pre-requisites

It is assumed that the reader is familiar with the Erlang programming language, concepts of OTP and has a basic knowledge of SNMP.

## 1.2 Mibs

### 1.2.1 Structure

The OTP mibs are stored in the `$OTP_ROOT/lib/otp_mibs/mibs/` directory. They are defined in SNMPv2 SMI syntax. An SNMPv1 version of the mib is delivered in the `mibs/v1` directory. The compiled MIB is located under `priv/mibs`, and the generated `.hrl` file under the `include` directory. To compile a MIB that **IMPORTS** a MIB in the **OTP\_Mibs** application, give the option `{il, ["otp_mibs/priv/mibs"]}` to the MIB compiler.

### 1.2.2 OTP-MIB

The **OTP-MIB** mib represents information about Erlang nodes such as node name, number of running processes, virtual machine version etc. If the MIB should be used in a system, it should be loaded into an SNMP agent by using the API function `otp_mib:load/1`.

### 1.2.3 OTP-REG

The **OTP-REG** mib defines the unique OTP subtree of object identifiers under the Ericsson subtree. Under the OTP subtree several object identifiers are defined. This module is typically included by OTP applications defining their own mibs, or ASN.1 modules in general, that require unique object identifiers under the OTP subtree.

### 1.2.4 OTP-TC

The **OTP-TC** mib provides the textual convention datatype `OwnerString`.

## 2 Reference Manual

---

The **OTP\_Mibs** application provides an SNMP management information base for Erlang nodes.

## otp\_mib

---

Erlang module

The SNMP application should be used to start an SNMP agent. Then the API functions below can be used to load/unload the OTP-MIB into/from the agent. The instrumentation of the OTP-MIB uses Mnesia, hence Mnesia must be started prior to loading the OTP-MIB.

### Exports

`load(Agent) -> ok | {error, Reason}`

Types:

`Agent = pid() | atom()`

`Reason = term()`

Loads the OTP-MIB.

`unload(Agent) -> ok | {error, Reason}`

Types:

`Agent = pid() | atom()`

`Reason = term()`

Unloads the OTP-MIB.

### See Also

[snmp\(3\)](#)