dwm Status Bar

Generated by Doxygen 1.8.20

Chapter 1

Overview

dwmbar is a status bar for dwm similar to dwmblocks. I wrote it in C++ just to troll the suckless people. It has some built-in modules, but can also be extended with external scripts.

Each module can be set to update after a separate interval. Modules run as separate threads and alert the main thread to print to the root window when a change occurs. You can also run a module by issuing a real-time signal with pkill, e.g.

```
pkill --signal RTMIN+1 -x dwmbar
```

The signal ID is set per module during configuration (see below). Modules that are running on a schedule can still be activated by a signal.

dwm supports two status bars (bottom and top) if you have the dwm-extrabar patch.

1.1 Install

To install clone this repository and use make:

```
cd dwmBar
make
sudo make install
```

This will put the dwmbar binary in /usr/local/bin/ and assumes gcc is the compiler on the system. If you have llvm instead, use

```
make CXX=c++
```

1.2 Dependencies

The project depends on a C++ compiler that understands C++11. It also requires libX11 for printing to the root window. Some included modules also require procfs to be mounted. This is available by default in most linux distributions, but may need to be explicitly mounted in BSD.

1.3 Configure

`dwmbar` is configured by editing the `config.hpp` file. Comments within the file explain what to do and the available options. If you want to customize further, full interface documentation is `here`, or you can run `doxygen` in the source code directory. The example configuration included here is the one I use on my main machine. The external shell scripts I use are included in the `scripts` directory.

Here is a screenshot from my system:

Chapter 2

Hierarchical Index

2.1 Class Hierarchy

This inheritance list is sorted roughly, but not completely, alphabetically:

Chapter 3

Class Index

3.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

Chapter 4

File Index

4.1 File List

Here is a list of all documented files with brief descriptions:

Chapter 5

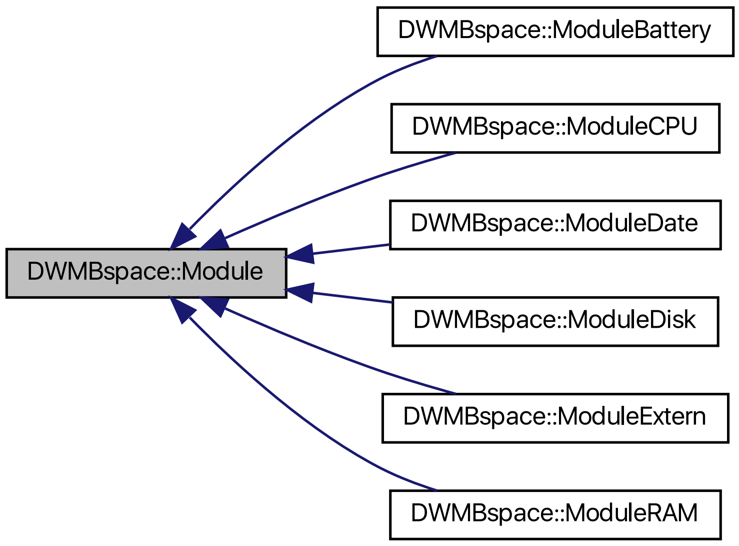
Class Documentation

5.1 DWMBspace::Module Class Reference

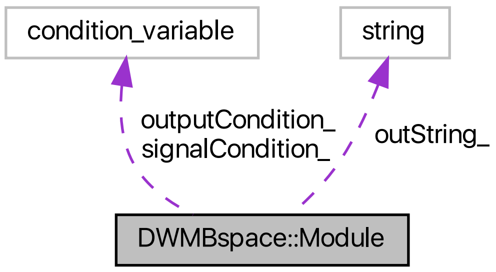
Base module class.

```
#include <modules.hpp>
```

Inheritance diagram for DWMBspace::Module:



Collaboration diagram for DWMBspace::Module:



Public Member Functions

- virtual ∼Module ()

    Destructor.
- void operator() () const

Protected Member Functions

- Module ()
- Module (const uint32_t &interval, string ∗output, condition_variable ∗cVar, condition_variable ∗sigVar)
- virtual void runModule_ () const =0

    Run the module once.

Protected Attributes

- uint32_t refreshInterval_
- string ∗ outString_
- condition_variable ∗ outputCondition_

    Pointer to a condition variable to signal change in state.
- condition_variable ∗ signalCondition_

    Pointer to a condition variable to accept signal events.

5.1.1 Detailed Description

Base module class.

Establishes the common parameters for all modules. Modules are functors that write output to a `string` variable.

5.1.2 Constructor & Destructor Documentation

5.1.2.1 Module() [1/2]

```
DWMBspace::Module::Module ( )  [inline], [protected]
```

Default constructor

5.1.2.2 Module() [2/2]

```
DWMBspace::Module::Module (
             const uint32_t & interval,
             string * output,
             condition_variable * cVar,
             condition_variable * sigVar )  [inline], [protected]
```

Constructor

Parameters

in	*interval*	refresh time interval in seconds
in,out	*output*	pointer to the output storing string
in,out	*cVar*	pointer to the condition variable for change signaling
in,out	*sigVar*	pointer to the condition variable to monitor real-time signals

5.1.3 Member Function Documentation

5.1.3.1 operator()()

```
void Module::operator() ( ) const
```

Run the module

Runs the module, refreshing at the specified interval or after receiving a refresh signal.

5.1.3.2 runModule_()

```
virtual void DWMBspace::Module::runModule_ ( ) const  [protected], [pure virtual]
```

Run the module once.

Retrieves the data specific to the module and formats the output.

Implemented in DWMBspace::ModuleExtern, DWMBspace::ModuleDisk, DWMBspace::ModuleRAM, DWMBspace::ModuleCPU, DWMBspace::ModuleBattery, and DWMBspace::ModuleDate.

5.1.4 Member Data Documentation

5.1.4.1 outputCondition_

```
condition_variable* DWMBspace::Module::outputCondition_ [protected]
```

Pointer to a condition variable to signal change in state.

The module is using this to communicate to the main thread.

5.1.4.2 outString_

```
string* DWMBspace::Module::outString_ [protected]
```

Pointer to the `string` that receives output

5.1.4.3 refreshInterval_

```
uint32_t DWMBspace::Module::refreshInterval_ [protected]
```

Refresh interval in seconds

5.1.4.4 signalCondition_

```
condition_variable* DWMBspace::Module::signalCondition_ [protected]
```

Pointer to a condition variable to accept signal events.

The module is waiting for this if it relies on a real-time signal to refresh.

The documentation for this class was generated from the following files:

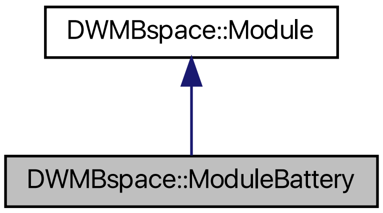
- modules.hpp
- modules.cpp

5.2 DWMBspace::ModuleBattery Class Reference

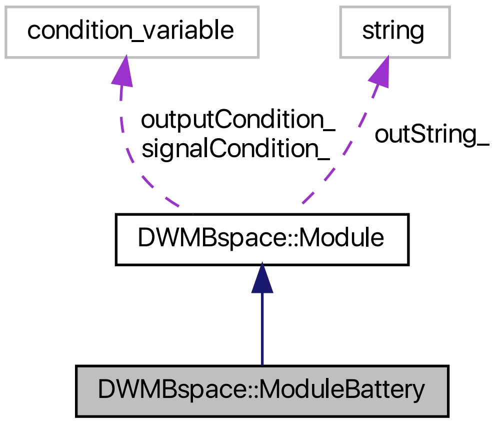
Battery state.

```
#include <modules.hpp>
```

Inheritance diagram for DWMBspace::ModuleBattery:



Collaboration diagram for DWMBspace::ModuleBattery:



Public Member Functions

- ModuleBattery ()

    Default constructor.
- ModuleBattery (const uint32_t &interval, string ∗output, condition_variable ∗cVar, condition_variable ∗sigVar)
- ∼ModuleBattery ()

    Destructor.

Protected Member Functions

- void runModule_ () const override
  Run the module once.

Additional Inherited Members

5.2.1 Detailed Description

Battery state.

Displays the battery state.

5.2.2 Constructor & Destructor Documentation

5.2.2.1 ModuleBattery()

```
DWMBspace::ModuleBattery::ModuleBattery (
             const uint32_t & interval,
             string * output,
             condition_variable * cVar,
             condition_variable * sigVar )  [inline]
```

Constructor

Parameters

in	*interval*	refresh time interval in seconds
in,out	*output*	pointer to the output storing string
in,out	*cVar*	pointer to the condition variable for change signaling
in,out	*sigVar*	pointer to the condition variable to monitor real-time signals

5.2.3 Member Function Documentation

5.2.3.1 runModule_()

```
void ModuleBattery::runModule_ ( ) const  [override], [protected], [virtual]
```

Run the module once.

Retrieves the data specific to the module and formats the output.

Implements DWMBspace::Module.

The documentation for this class was generated from the following files:

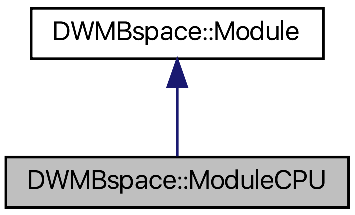
- modules.hpp
- modules.cpp

5.3 DWMBspace::ModuleCPU Class Reference

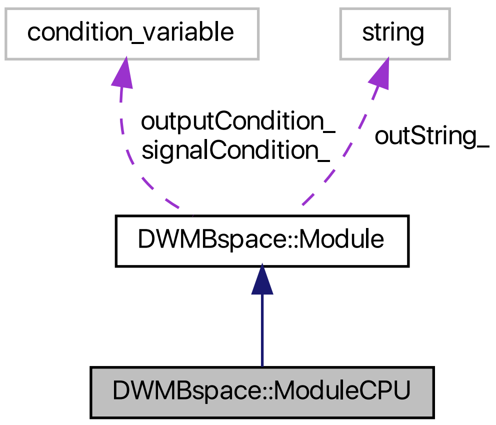
CPU status.

```
#include <modules.hpp>
```

Inheritance diagram for DWMBspace::ModuleCPU:



Collaboration diagram for DWMBspace::ModuleCPU:



Public Member Functions

- ModuleCPU ()

  Default constructor.
- ModuleCPU (const uint32_t &interval, string ∗output, condition_variable ∗cVar, condition_variable ∗sigVar)
- ∼ModuleCPU ()

  Destructor.

Protected Member Functions

- void runModule_ () const override

  Run the module once.

Protected Attributes

- float previousTotalLoad_

  Previous total CPU time.
- float previousIdleLoad_

  Previous idle CPU time.

5.3.1 Detailed Description

CPU status.

Displays CPU temperature and load.

5.3.2 Constructor & Destructor Documentation

5.3.2.1 ModuleCPU()

```
DWMBspace::ModuleCPU::ModuleCPU (
             const uint32_t & interval,
             string * output,
             condition_variable * cVar,
             condition_variable * sigVar )  [inline]
```

Constructor

Parameters

in	*interval*	refresh time interval in seconds
in,out	*output*	pointer to the output storing string
in,out	*cVar*	pointer to the condition variable for change signaling
in,out	*sigVar*	pointer to the condition variable to monitor real-time signals

5.3.3 Member Function Documentation

5.3.3.1 runModule_()

```
void ModuleCPU::runModule_ ( ) const  [override], [protected], [virtual]
```

Run the module once.

Retrieves the data specific to the module and formats the output.

Implements DWMBspace::Module.

The documentation for this class was generated from the following files:

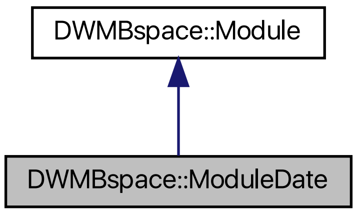
- modules.hpp
- modules.cpp

5.4 DWMBspace::ModuleDate Class Reference

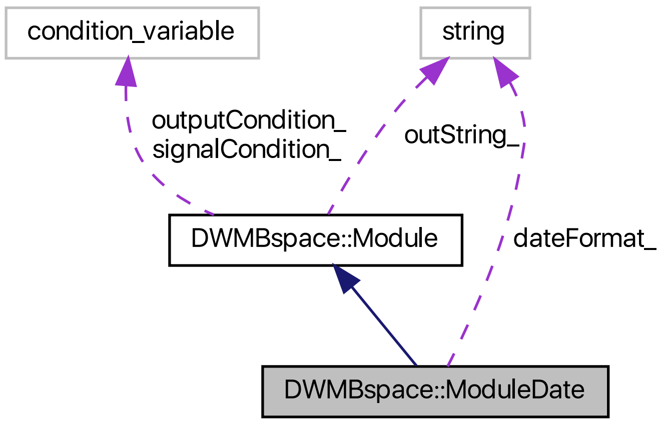
Time and date.

```
#include <modules.hpp>
```

Inheritance diagram for DWMBspace::ModuleDate:



Collaboration diagram for DWMBspace::ModuleDate:



Public Member Functions

- ModuleDate ()
- ModuleDate (const uint32_t &interval, const string &dateFormat, string ∗output, condition_variable ∗cVar, condition_variable ∗sigVar)
- ∼ModuleDate ()

  Destructor.

Protected Member Functions

- void runModule_ () const override

  Run the module once.

Protected Attributes

- string dateFormat_

  Time format string.

5.4.1 Detailed Description

Time and date.

5.4.2 Constructor & Destructor Documentation

5.4.2.1 ModuleDate() [1/2]

```
DWMBspace::ModuleDate::ModuleDate ( )  [inline]
```

Default constructor

5.4.2.2 ModuleDate() [2/2]

```
DWMBspace::ModuleDate::ModuleDate (
             const uint32_t & interval,
             const string & dateFormat,
             string * output,
             condition_variable * cVar,
             condition_variable * sigVar )  [inline]
```

Constructor

Parameters

in	*interval*	refresh time interval in seconds
in,out	*output*	pointer to the output storing string
in,out	*cVar*	pointer to the condition variable for change signaling
in,out	*sigVar*	pointer to the condition variable to monitor real-time signals

5.4.3 Member Function Documentation

5.4.3.1 runModule_()

```
void ModuleDate::runModule_ ( ) const  [override], [protected], [virtual]
```

Run the module once.

Retrieves the data specific to the module and formats the output.

Implements DWMBspace::Module.

5.4.4 Member Data Documentation

5.4.4.1 dateFormat_

```
string DWMBspace::ModuleDate::dateFormat_  [protected]
```

Time format string.

Date display format, same as for the Unix `date` command.

The documentation for this class was generated from the following files:

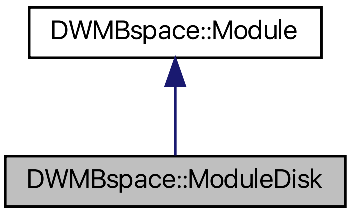
- modules.hpp
- modules.cpp

5.5 DWMBspace::ModuleDisk Class Reference

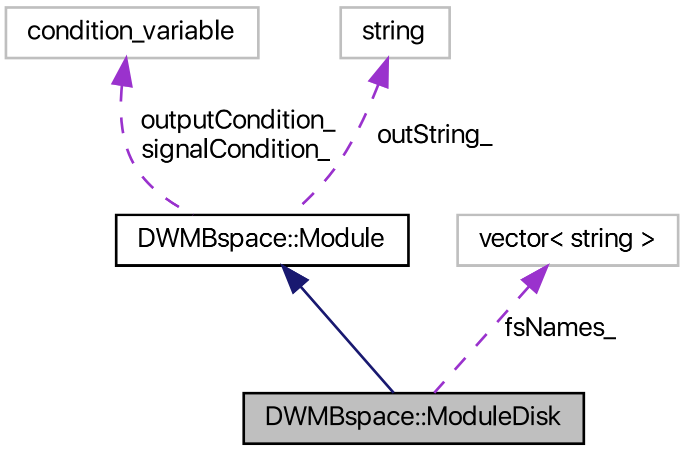
Disk free space.

```
#include <modules.hpp>
```

Inheritance diagram for DWMBspace::ModuleDisk:



Collaboration diagram for DWMBspace::ModuleDisk:



Public Member Functions

- ModuleDisk ()

    Default constructor.
- ModuleDisk (const uint32_t &interval, const vector< string > &fsVector, string ∗output, condition_variable ∗cVar, condition_variable ∗sigVar)
- ∼ModuleDisk ()

    Destructor.

Protected Member Functions

- void runModule_ () const override

    Run the module once.

Protected Attributes

- vector< string > fsNames_

    File system names.

5.5.1 Detailed Description

Disk free space.

Lists free space in a list of file systems in Gb and RAID status if available.

5.5.2 Constructor & Destructor Documentation

5.5.2.1 ModuleDisk()

```
DWMBspace::ModuleDisk::ModuleDisk (
             const uint32_t & interval,
             const vector< string > & fsVector,
             string * output,
             condition_variable * cVar,
             condition_variable * sigVar )  [inline]
```

Constructor

Parameters

in	*interval*	refresh time interval in seconds
in	*fsVector*	vector of file system names
in,out	*output*	pointer to the output storing string
in,out	*cVar*	pointer to the condition variable for change signaling
in,out	*sigVar*	pointer to the condition variable to monitor real-time signals

5.5.3 Member Function Documentation

5.5.3.1 runModule_()

```
void ModuleDisk::runModule_ ( ) const  [override], [protected], [virtual]
```

Run the module once.

Retrieves the data specific to the module and formats the output.

Implements DWMBspace::Module.

The documentation for this class was generated from the following files:

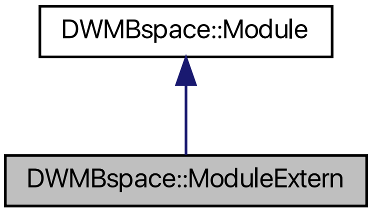
- modules.hpp
- modules.cpp

5.6 DWMBspace::ModuleExtern Class Reference

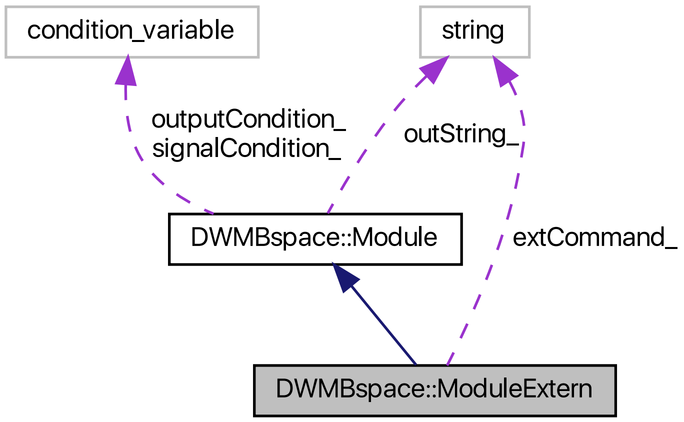
External scripts.

`#include <modules.hpp>`

Inheritance diagram for DWMBspace::ModuleExtern:



Collaboration diagram for DWMBspace::ModuleExtern:



Public Member Functions

- ModuleExtern ()

  Default constructor.
- ModuleExtern (const uint32_t &interval, const string &command, string ∗output, condition_variable ∗cVar, condition_variable ∗sigVar)
- ∼ModuleExtern ()

  Destructor.

Protected Member Functions

- void runModule_ () const override

  Run the module once.

Protected Attributes

- const string extCommand_

  External command string.

Static Protected Attributes

- static const size_t lengthLimit_ = 500

  Output length limit.

5.6.1 Detailed Description

External scripts.

Runs an external script or shell command and displays the output. No formatting of the external output is performed, but it is truncated to 500 characters.

5.6.2 Constructor & Destructor Documentation

5.6.2.1 ModuleExtern()

```
DWMBspace::ModuleExtern::ModuleExtern (
             const uint32_t & interval,
             const string & command,
             string * output,
             condition_variable * cVar,
             condition_variable * sigVar )  [inline]
```

Constructor

Parameters

in	*interval*	refresh time interval in seconds
in	*command*	external command
in,out	*output*	pointer to the output storing string
in,out	*cVar*	pointer to the condition variable for change signaling
in,out	*sigVar*	pointer to the condition variable to monitor real-time signals

5.6.3 Member Function Documentation

5.6.3.1 runModule_()

```
void ModuleExtern::runModule_ ( ) const  [override], [protected], [virtual]
```

Run the module once.

Runs the external shell command or script and returns the output, truncating to 500.

Implements DWMBspace::Module.

The documentation for this class was generated from the following files:

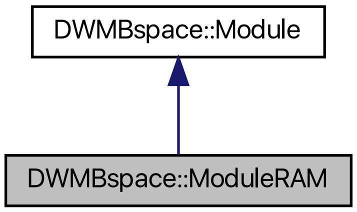
- modules.hpp
- modules.cpp

5.7 DWMBspace::ModuleRAM Class Reference

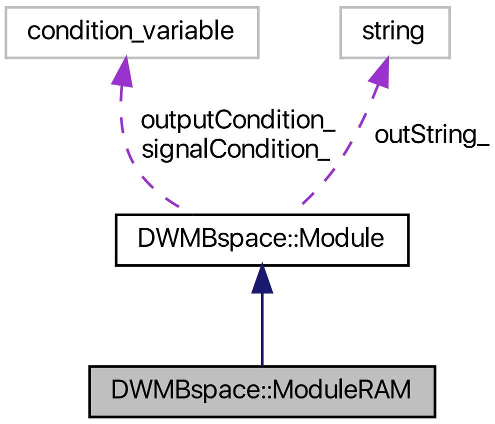
Free memory.

```
#include <modules.hpp>
```

Inheritance diagram for DWMBspace::ModuleRAM:



Collaboration diagram for DWMBspace::ModuleRAM:



Public Member Functions

- ModuleRAM ()

  Default constructor.
- ModuleRAM (const uint32_t &interval, string ∗output, condition_variable ∗cVar, condition_variable ∗sigVar)
- ∼ModuleRAM ()

  Destructor.

Protected Member Functions

- void runModule_ () const override

  Run the module once.

Additional Inherited Members

5.7.1 Detailed Description

Free memory.

Displays the amount of free RAM.

5.7.2 Constructor & Destructor Documentation

5.7.2.1 ModuleRAM()

```
DWMBspace::ModuleRAM::ModuleRAM (
            const uint32_t & interval,
            string * output,
            condition_variable * cVar,
            condition_variable * sigVar )  [inline]
```

Constructor

Parameters

in	*interval*	refresh time interval in seconds
in,out	*output*	pointer to the output storing string
in,out	*cVar*	pointer to the condition variable for change signaling
in,out	*sigVar*	pointer to the condition variable to monitor real-time signals

5.7.3 Member Function Documentation

5.7.3.1 runModule_()

```
void ModuleRAM::runModule_ ( ) const  [override], [protected], [virtual]
```

Run the module once.

Retrieves the data specific to the module and formats the output.

Implements DWMBspace::Module.

The documentation for this class was generated from the following files:

- modules.hpp
- modules.cpp

Chapter 6

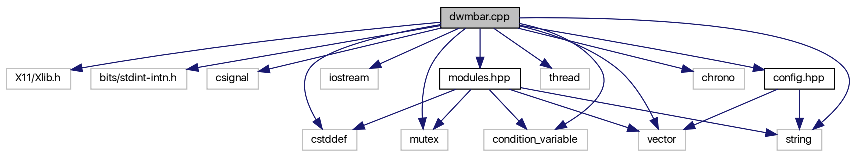
File Documentation

6.1 dwmbar.cpp File Reference

A bar for dwm.

```
#include <X11/Xlib.h>
#include <bits/stdint-intn.h>
#include <csignal>
#include <cstddef>
#include <iostream>
#include <string>
#include <vector>
#include <thread>
#include <mutex>
#include <condition_variable>
#include <chrono>
#include "modules.hpp"
#include "config.hpp"
```

Include dependency graph for dwmbar.cpp:



Functions

- static vector< condition_variable > signalCondition (sigRTNUM)

  Condition variables that will respond to real-time signals.
- void makeBarOutput (const vector< string > &moduleOutput, const string &delimiter, string &barText)

Make bar output.
- void printRoot (const string &barOutput)

  Render the bar.
- void processSignal (int sig)

  Process real-time signals.
- int **main** ()

Variables

- static const int sigRTNUM = 30

  Number of possible real-time signals.

6.1.1 Detailed Description

A bar for dwm.

Author

Anthony J. Greenberg

Copyright

Copyright (c) 2020 Anthony J. Greenberg

Version

0.9

Displays information on the bar for the Dynamic Window Manager (dwm). External scripts and some internal functions are supported. Can use two bars (bottom and top) if dwm is patched with `dwm-extrabar`.

6.1.2 Function Documentation

6.1.2.1 makeBarOutput()

```
void makeBarOutput (
            const vector< string > & moduleOutput,
            const string & delimiter,
            string & barText )
```

Make bar output.

Takes individual module outputs and puts them together for printing.

Parameters

`in`	*moduleOutput*	vector of individual module outputs
`in`	*delimiter*	delimiter character(s) between modules
`out`	*barText*	compiled text to be printed to the bar

6.1.2.2 printRoot()

```
void printRoot (
            const string & barOutput )
```

Render the bar.

Renders the bar text by printing the provided string to the root window. This is how dwm handles status bars.

Parameters

`in`	*barOutput*	text to be displayed

6.1.2.3 processSignal()

```
void processSignal (
            int sig )
```

Process real-time signals.

Receive and process real-time signals to trigger relevant modules.

Parameters

`in`	*sig*	signal number (starting at `SIGRTMIN`)

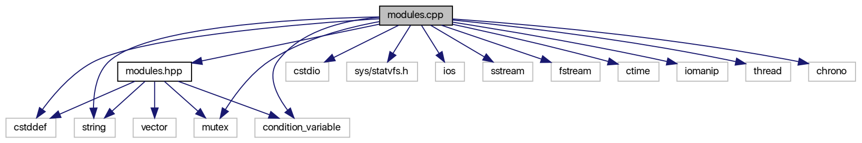
6.2 modules.cpp File Reference

C++ modules for the status bar (implementation)

```
#include <cstddef>
#include <cstdio>
#include <sys/statvfs.h>
```

```
#include <ios>
#include <string>
#include <sstream>
#include <fstream>
#include <ctime>
#include <iomanip>
#include <thread>
#include <mutex>
#include <condition_variable>
#include <chrono>
#include "modules.hpp"
```

Include dependency graph for modules.cpp:



6.2.1 Detailed Description

C++ modules for the status bar (implementation)

Author

Anthony J. Greenberg

Copyright

Copyright (c) 2020 Anthony J. Greenberg

Version

0.9

Implementation of classes that provide output useful for display in the status bar.

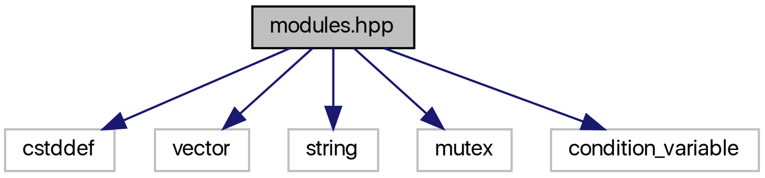
6.3 modules.hpp File Reference

C++ modules for the status bar (definitions)

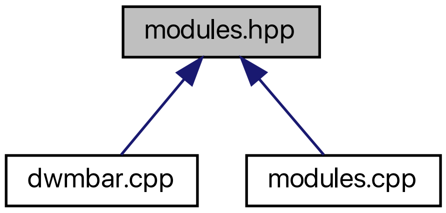
```
#include <cstddef>
#include <vector>
#include <string>
```

```
#include <mutex>
#include <condition_variable>
```

Include dependency graph for modules.hpp:



This graph shows which files directly or indirectly include this file:



Classes

- class DWMBspace::Module

    Base module class.
- class DWMBspace::ModuleDate

    Time and date.
- class DWMBspace::ModuleBattery

    Battery state.
- class DWMBspace::ModuleCPU

    CPU status.
- class DWMBspace::ModuleRAM

    Free memory.
- class DWMBspace::ModuleDisk

    Disk free space.
- class DWMBspace::ModuleExtern

    External scripts.

6.3.1 Detailed Description

C++ modules for the status bar (definitions)

Author

Anthony J. Greenberg

Copyright

Copyright (c) 2020 Anthony J. Greenberg

Version

0.9

Definitions of classes that provide output useful for display in the status bar.

Index