

## Arcade

Generated by Doxygen 1.9.7



<b>1 Namespace Index</b>	<b>1</b>
1.1 Namespace List	1
<b>2 Hierarchical Index</b>	<b>3</b>
2.1 Class Hierarchy	3
<b>3 Class Index</b>	<b>5</b>
3.1 Class List	5
<b>4 File Index</b>	<b>7</b>
4.1 File List	7
<b>5 Namespace Documentation</b>	<b>9</b>
5.1 arcade Namespace Reference	9
5.1.1 Detailed Description	9
<b>6 Class Documentation</b>	<b>11</b>
6.1 arcade::ADisplayModule Class Reference	11
6.1.1 Member Enumeration Documentation	12
6.1.1.1 DisplayStatus	12
6.1.2 Constructor & Destructor Documentation	13
6.1.2.1 ADisplayModule()	13
6.1.2.2 ~ADisplayModule()	13
6.1.3 Member Function Documentation	13
6.1.3.1 display()	13
6.1.3.2 getDisplayStatus()	13
6.1.3.3 getInput()	13
6.1.3.4 getName()	13
6.1.3.5 getType()	14
6.1.3.6 init()	14
6.1.3.7 sendGameData()	14
6.1.3.8 setDisplayStatus()	14
6.1.3.9 stop()	14
6.1.4 Member Data Documentation	15
6.1.4.1 _displayStatus	15
6.1.4.2 _event	15
6.1.4.3 _gameData	15
6.1.4.4 _input	15
6.1.4.5 _texture	15
6.1.4.6 _window	15
6.2 arcade::AGameModule Class Reference	15
6.2.1 Member Enumeration Documentation	17
6.2.1.1 GameStatus	17
6.2.2 Constructor & Destructor Documentation	17

6.2.2.1 AGameModule()	17
6.2.2.2 ~AGameModule()	17
6.2.3 Member Function Documentation	17
6.2.3.1 getDisplayStatus()	17
6.2.3.2 getName()	18
6.2.3.3 getType()	18
6.2.3.4 init()	18
6.2.3.5 sendGameData()	18
6.2.3.6 sendInput()	18
6.2.3.7 setGameStatus()	19
6.2.3.8 stop()	19
6.2.4 Member Data Documentation	19
6.2.4.1 _gameData	19
6.2.4.2 _gameStatus	19
6.2.4.3 _input	19
6.3 arcade::CoreModule Class Reference	19
6.3.1 Member Enumeration Documentation	21
6.3.1.1 CoreStatus	21
6.3.2 Constructor & Destructor Documentation	21
6.3.2.1 CoreModule()	21
6.3.2.2 ~CoreModule()	21
6.3.3 Member Function Documentation	21
6.3.3.1 getCoreStatus()	21
6.3.3.2 getDisplayModule()	22
6.3.3.3 getGameModule()	22
6.3.3.4 getLib()	22
6.3.3.5 getName()	22
6.3.3.6 getType()	22
6.3.3.7 init()	22
6.3.3.8 setCoreStatus()	23
6.3.3.9 setModule()	23
6.3.3.10 stop()	23
6.3.4 Member Data Documentation	23
6.3.4.1 _coreStatus	23
6.3.4.2 _displayModule	24
6.3.4.3 _gameModule	24
6.4 DLLoader< T > Class Template Reference	24
6.4.1 Constructor & Destructor Documentation	24
6.4.1.1 DLLoader()	24
6.4.1.2 ~DLLoader()	24
6.4.2 Member Function Documentation	25
6.4.2.1 getInstance()	25

6.4.3 Member Data Documentation	25
6.4.3.1 handle	25
6.5 arcade::IModule::GameData Struct Reference	25
6.5.1 Detailed Description	25
6.5.2 Member Data Documentation	25
6.5.2.1 display_info	25
6.5.2.2 sprite_value	25
6.6 arcade::IModule Class Reference	26
6.6.1 Detailed Description	26
6.6.2 Member Enumeration Documentation	27
6.6.2.1 KeyboardInput	27
6.6.2.2 LibName	27
6.6.2.3 ModuleType	28
6.6.3 Constructor & Destructor Documentation	28
6.6.3.1 IModule()	28
6.6.3.2 ~IModule()	28
6.6.4 Member Function Documentation	28
6.6.4.1 getName()	28
6.6.4.2 getType()	29
6.6.4.3 init()	29
6.6.4.4 stop()	29
6.7 arcade::NCurses Class Reference	29
6.7.1 Constructor & Destructor Documentation	31
6.7.1.1 NCurses()	31
6.7.1.2 ~NCurses()	31
6.7.2 Member Function Documentation	31
6.7.2.1 display()	31
6.7.2.2 getName()	31
6.7.2.3 init()	31
6.7.2.4 stop()	32
6.8 arcade::Pacman Class Reference	32
6.8.1 Constructor & Destructor Documentation	33
6.8.1.1 Pacman()	33
6.8.1.2 ~Pacman()	34
6.8.2 Member Function Documentation	34
6.8.2.1 getName()	34
6.8.2.2 init()	34
6.8.2.3 stop()	34
6.9 arcade::Sdl2 Class Reference	34
6.9.1 Constructor & Destructor Documentation	36
6.9.1.1 Sdl2()	36
6.9.1.2 ~Sdl2()	36

6.9.2 Member Function Documentation	36
6.9.2.1 display()	36
6.9.2.2 getName()	37
6.9.2.3 init()	37
6.9.2.4 stop()	37
6.10 arcade::Sfml Class Reference	37
6.10.1 Constructor & Destructor Documentation	39
6.10.1.1 Sfml()	39
6.10.1.2 ~Sfml()	39
6.10.2 Member Function Documentation	39
6.10.2.1 display()	39
6.10.2.2 getInput()	40
6.10.2.3 getName()	40
6.10.2.4 init()	40
6.10.2.5 stop()	40
6.10.3 Member Data Documentation	40
6.10.3.1 _font	40
6.10.3.2 _texture	40
6.11 arcade::Snake Class Reference	40
6.11.1 Constructor & Destructor Documentation	42
6.11.1.1 Snake()	42
6.11.1.2 ~Snake()	42
6.11.2 Member Function Documentation	42
6.11.2.1 getName()	42
6.11.2.2 init()	43
6.11.2.3 stop()	43
<b>7 File Documentation</b>	<b>45</b>
7.1 /home/tmندی/Documents/Tek2/OOP/Arcade/include/CoreModule.hpp File Reference	45
7.2 CoreModule.hpp	45
7.3 /home/tmندی/Documents/Tek2/OOP/Arcade/include/DLLoader.hpp File Reference	46
7.4 DLLoader.hpp	46
7.5 /home/tmندی/Documents/Tek2/OOP/Arcade/include/game/AGameModule.hpp File Reference	47
7.6 AGameModule.hpp	47
7.7 /home/tmندی/Documents/Tek2/OOP/Arcade/include/game/Pacman.hpp File Reference	48
7.8 Pacman.hpp	48
7.9 /home/tmندی/Documents/Tek2/OOP/Arcade/include/game/Snake.hpp File Reference	48
7.10 Snake.hpp	49
7.11 /home/tmندی/Documents/Tek2/OOP/Arcade/include/graphic/ADisplayModule.hpp File Reference	49
7.12 ADisplayModule.hpp	50
7.13 /home/tmندی/Documents/Tek2/OOP/Arcade/include/graphic/NCurses.hpp File Reference	50
7.14 NCurses.hpp	51

7.15 /home/tmendy/Documents/Tek2/OOP/Arcade/include/graphic/Sdl2.hpp File Reference . . . . .	51
7.16 Sdl2.hpp . . . . .	51
7.17 /home/tmendy/Documents/Tek2/OOP/Arcade/include/graphic/Sfml.hpp File Reference . . . . .	52
7.18 Sfml.hpp . . . . .	52
7.19 /home/tmendy/Documents/Tek2/OOP/Arcade/include/IModule.hpp File Reference . . . . .	53
7.20 IModule.hpp . . . . .	53
7.21 /home/tmendy/Documents/Tek2/OOP/Arcade/include/Macros.hpp File Reference . . . . .	54
7.21.1 Macro Definition Documentation . . . . .	54
7.21.1.1 ERROR . . . . .	54
7.21.1.2 KO . . . . .	54
7.21.1.3 OK . . . . .	55
7.22 Macros.hpp . . . . .	55
7.23 /home/tmendy/Documents/Tek2/OOP/Arcade/lib/CoreModule.cpp File Reference . . . . .	55
7.23.1 Function Documentation . . . . .	55
7.23.1.1 getLib() . . . . .	55
7.24 /home/tmendy/Documents/Tek2/OOP/Arcade/lib/games/AGameModule.cpp File Reference . . . . .	56
7.25 /home/tmendy/Documents/Tek2/OOP/Arcade/lib/games/pacman/Pacman.cpp File Reference . . . . .	56
7.25.1 Function Documentation . . . . .	56
7.25.1.1 entryPoint() . . . . .	56
7.25.1.2 init() . . . . .	56
7.25.1.3 stop() . . . . .	56
7.26 /home/tmendy/Documents/Tek2/OOP/Arcade/lib/games/snake/Snake.cpp File Reference . . . . .	57
7.26.1 Function Documentation . . . . .	57
7.26.1.1 entryPoint() . . . . .	57
7.27 /home/tmendy/Documents/Tek2/OOP/Arcade/lib/graphics/ADisplayModule.cpp File Reference . . . . .	57
7.28 /home/tmendy/Documents/Tek2/OOP/Arcade/lib/graphics/ncurses/NCurses.cpp File Reference . . . . .	57
7.28.1 Function Documentation . . . . .	57
7.28.1.1 entryPoint() . . . . .	57
7.29 /home/tmendy/Documents/Tek2/OOP/Arcade/lib/graphics/sdl2/Sdl2.cpp File Reference . . . . .	58
7.29.1 Function Documentation . . . . .	58
7.29.1.1 entryPoint() . . . . .	58
7.30 /home/tmendy/Documents/Tek2/OOP/Arcade/lib/graphics/sfml/Sfml.cpp File Reference . . . . .	58
7.30.1 Function Documentation . . . . .	58
7.30.1.1 entryPoint() . . . . .	58
7.31 /home/tmendy/Documents/Tek2/OOP/Arcade/src/Main.cpp File Reference . . . . .	58
7.31.1 Function Documentation . . . . .	59
7.31.1.1 arcadeRe() . . . . .	59
7.31.1.2 help() . . . . .	59
7.31.1.3 is_good_graphic_lib() . . . . .	59
7.31.1.4 main() . . . . .	60





# Chapter 1

## Namespace Index

### 1.1 Namespace List

Here is a list of all namespaces with brief descriptions:

<a href="#">arcade</a>	Namespace for the arcade project . . . . .	9
------------------------	--	---



## Chapter 2

# Hierarchical Index

### 2.1 Class Hierarchy

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

DLLoader< T > . . . . .	24
arcade::IModule::GameData . . . . .	25
arcade::IModule . . . . .	26
arcade::ADisplayModule . . . . .	11
arcade::NCurses . . . . .	29
arcade::Sdl2 . . . . .	34
arcade::Sfml . . . . .	37
arcade::AGameModule . . . . .	15
arcade::Pacman . . . . .	32
arcade::Snake . . . . .	40
arcade::CoreModule . . . . .	19



## Chapter 3

# Class Index

### 3.1 Class List

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

<a href="#">arcade::ADisplayModule</a>	11
<a href="#">arcade::AGameModule</a>	15
<a href="#">arcade::CoreModule</a>	19
<a href="#">DLLoader&lt; T &gt;</a>	24
<a href="#">arcade::IModule::GameData</a>	
Information about the game from the game module to the graphic module	25
<a href="#">arcade::IModule</a>	
Interface for the modules	26
<a href="#">arcade::NCurses</a>	29
<a href="#">arcade::Pacman</a>	32
<a href="#">arcade::Sdl2</a>	34
<a href="#">arcade::Sfml</a>	37
<a href="#">arcade::Snake</a>	40



# Chapter 4

## File Index

### 4.1 File List

Here is a list of all files with brief descriptions:

/home/tmندی/Documents/Tek2/OOP/Arcade/include/CoreModule.hpp . . . . .	45
/home/tmندی/Documents/Tek2/OOP/Arcade/include/DLLoader.hpp . . . . .	46
/home/tmندی/Documents/Tek2/OOP/Arcade/include/IModule.hpp . . . . .	53
/home/tmندی/Documents/Tek2/OOP/Arcade/include/Macros.hpp . . . . .	54
/home/tmندی/Documents/Tek2/OOP/Arcade/include/game/AGameModule.hpp . . . . .	47
/home/tmندی/Documents/Tek2/OOP/Arcade/include/game/Pacman.hpp . . . . .	48
/home/tmندی/Documents/Tek2/OOP/Arcade/include/game/Snake.hpp . . . . .	48
/home/tmندی/Documents/Tek2/OOP/Arcade/include/graphic/ADisplayModule.hpp . . . . .	49
/home/tmندی/Documents/Tek2/OOP/Arcade/include/graphic/NCurses.hpp . . . . .	50
/home/tmندی/Documents/Tek2/OOP/Arcade/include/graphic/Sdl2.hpp . . . . .	51
/home/tmندی/Documents/Tek2/OOP/Arcade/include/graphic/Sfml.hpp . . . . .	52
/home/tmندی/Documents/Tek2/OOP/Arcade/lib/CoreModule.cpp . . . . .	55
/home/tmندی/Documents/Tek2/OOP/Arcade/lib/games/AGameModule.cpp . . . . .	56
/home/tmندی/Documents/Tek2/OOP/Arcade/lib/games/pacman/Pacman.cpp . . . . .	56
/home/tmندی/Documents/Tek2/OOP/Arcade/lib/games/snake/Snake.cpp . . . . .	57
/home/tmندی/Documents/Tek2/OOP/Arcade/lib/graphics/ADisplayModule.cpp . . . . .	57
/home/tmندی/Documents/Tek2/OOP/Arcade/lib/graphics/ncurses/NCurses.cpp . . . . .	57
/home/tmندی/Documents/Tek2/OOP/Arcade/lib/graphics/sdl2/Sdl2.cpp . . . . .	58
/home/tmندی/Documents/Tek2/OOP/Arcade/lib/graphics/sfml/Sfml.cpp . . . . .	58
/home/tmندی/Documents/Tek2/OOP/Arcade/src/Main.cpp . . . . .	58





## Chapter 5

# Namespace Documentation

### 5.1 arcade Namespace Reference

namespace for the arcade project

#### Classes

- class [ADisplayModule](#)
- class [AGameModule](#)
- class [CoreModule](#)
- class [IModule](#)

*Interface for the modules.*

- class [NCurses](#)
- class [Pacman](#)
- class [Sdl2](#)
- class [Sfml](#)
- class [Snake](#)

#### 5.1.1 Detailed Description

namespace for the arcade project



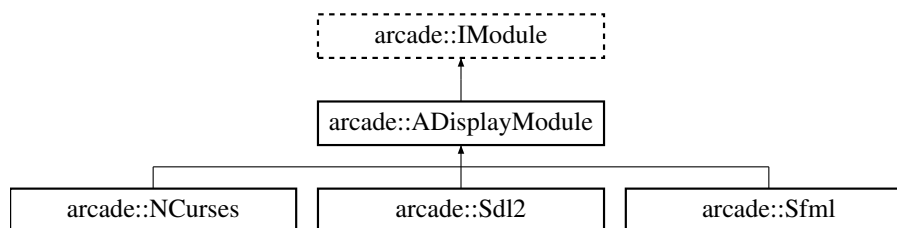
## Chapter 6

# Class Documentation

### 6.1 arcade::ADisplayModule Class Reference

```
#include <ADisplayModule.hpp>
```

Inheritance diagram for arcade::ADisplayModule:



#### Public Types

- enum `DisplayStatus` {  
    `RUNNING` , `PAUSED` , `SELECTION` , `GAMEOVER` ,  
    `WIN` }

#### Public Types inherited from `arcade::IModule`

- enum `KeyboardInput` {  
    `UP` , `DOWN` , `LEFT` , `RIGHT` ,  
    `SPACE` , `ENTER` , `ESCAPE` , `A` ,  
    `B` , `C` , `D` , `E` ,  
    `F` , `G` , `H` , `I` ,  
    `J` , `K` , `L` , `M` ,  
    `N` , `O` , `P` , `Q` ,  
    `R` , `S` , `T` , `U` ,  
    `V` , `W` , `X` , `Y` ,  
    `Z` , `NONE` }  
    *all the possible keyboard inputs*
- enum `ModuleType` { `GRAPHIC` , `GAME` , `CORE` }  
    *all the possible module types*
- enum `LibName` {  
    `SNAKE` , `NIBBLER` , `PACMAN` , `QIX` ,  
    `CENTIPEDE` , `SOLARFOX` , `NCURSES` , `SDL` ,  
    `SFML` , `OPENGL` , `UNKNOWN` }  
    *all the possible library names*

## Public Member Functions

- [ADisplayModule](#) ()  
*Construct a new [arcade::ADisplayModule::ADisplayModule](#) object.*
- [~ADisplayModule](#) ()  
*Destroy the [arcade::ADisplayModule::ADisplayModule](#) object.*
- virtual void [init](#) ()=0
- virtual void [stop](#) ()=0
- virtual void [display](#) ()=0
- void [setDisplayStatus](#) ([DisplayStatus](#) status)  
*set the status of the display module*
- [DisplayStatus](#) [getDisplayStatus](#) () const  
*return the status of the display module*
- virtual const [arcade::IModule::LibName](#) [getName](#) () const =0
- const [arcade::IModule::ModuleType](#) [getType](#) () const  
*return the type of the module*
- [arcade::IModule::KeyboardInput](#) [getInput](#) () const  
*get input from the user*
- void [sendGameData](#) ([arcade::IModule::GameData](#) data)  
*receive send data of the game module to the display module*

## Public Member Functions inherited from [arcade::IModule](#)

- [IModule](#) ()
- virtual [~IModule](#) ()
- virtual void [init](#) ()=0
- virtual void [stop](#) ()=0
- virtual const [LibName](#) [getName](#) () const =0
- virtual const [ModuleType](#) [getType](#) () const =0

## Protected Attributes

- void \* [\\_window](#)
- void \* [\\_texture](#)
- void \* [\\_event](#)
- [arcade::IModule::GameData](#) [\\_gameData](#)
- [arcade::IModule::KeyboardInput](#) [\\_input](#)
- [DisplayStatus](#) [\\_displayStatus](#)

## 6.1.1 Member Enumeration Documentation

### 6.1.1.1 DisplayStatus

```
enum arcade::ADisplayModule::DisplayStatus
```

#### Enumerator

RUNNING	
PAUSED	
SELECTION	
GAMEOVER	
WIN	

## 6.1.2 Constructor & Destructor Documentation

### 6.1.2.1 `ADisplayModule()`

```
arcade::ADisplayModule::ADisplayModule ( )
```

Construct a new `arcade::ADisplayModule::ADisplayModule` object.

### 6.1.2.2 `~ADisplayModule()`

```
arcade::ADisplayModule::~~ADisplayModule ( )
```

Destroy the `arcade::ADisplayModule::ADisplayModule` object.

## 6.1.3 Member Function Documentation

### 6.1.3.1 `display()`

```
virtual void arcade::ADisplayModule::display ( ) [pure virtual]
```

Implemented in `arcade::NCurses`, `arcade::Sdl2`, and `arcade::Sfml`.

### 6.1.3.2 `getDisplayStatus()`

```
arcade::ADisplayModule::DisplayStatus arcade::ADisplayModule::getDisplayStatus ( ) const
```

return the status of the display module

Returns

`arcade::ADisplayModule::DisplayStatus` status of the display module

### 6.1.3.3 `getInput()`

```
arcade::IModule::KeyboardInput arcade::ADisplayModule::getInput ( ) const
```

get input from the user

Returns

`arcade::IModule::KeyboardInput`

### 6.1.3.4 `getName()`

```
virtual const arcade::IModule::LibName arcade::ADisplayModule::getName ( ) const [pure virtual]
```

Implements `arcade::IModule`.

Implemented in `arcade::NCurses`, `arcade::Sdl2`, and `arcade::Sfml`.

#### 6.1.3.5 getType()

```
const arcade::IModule::ModuleType arcade::ADisplayModule::getType ( ) const [virtual]
```

return the type of the module

##### Returns

const [arcade::IModule::ModuleType](#)

Implements [arcade::IModule](#).

#### 6.1.3.6 init()

```
virtual void arcade::ADisplayModule::init ( ) [pure virtual]
```

Implements [arcade::IModule](#).

Implemented in [arcade::NCurses](#), [arcade::Sdl2](#), and [arcade::Sfml](#).

#### 6.1.3.7 sendGameData()

```
void arcade::ADisplayModule::sendGameData (
    arcade::IModule::GameData data )
```

receive send data of the game module to the display module

##### Parameters

<i>data</i>	of the game module (score, map, player position)
-------------	--

#### 6.1.3.8 setDisplayStatus()

```
void arcade::ADisplayModule::setDisplayStatus (
    DisplayStatus status )
```

set the status of the display module

##### Parameters

<i>status</i>	of the display module
---------------	-----------------------

#### 6.1.3.9 stop()

```
virtual void arcade::ADisplayModule::stop ( ) [pure virtual]
```

Implements [arcade::IModule](#).

Implemented in [arcade::NCurses](#), [arcade::Sdl2](#), and [arcade::Sfml](#).

## 6.1.4 Member Data Documentation

### 6.1.4.1 \_displayStatus

`DisplayStatus` `arcade::ADisplayModule::_displayStatus` [protected]

### 6.1.4.2 \_event

`void*` `arcade::ADisplayModule::_event` [protected]

### 6.1.4.3 \_gameData

`arcade::IModule::GameData` `arcade::ADisplayModule::_gameData` [protected]

### 6.1.4.4 \_input

`arcade::IModule::KeyboardInput` `arcade::ADisplayModule::_input` [protected]

### 6.1.4.5 \_texture

`void*` `arcade::ADisplayModule::_texture` [protected]

### 6.1.4.6 \_window

`void*` `arcade::ADisplayModule::_window` [protected]

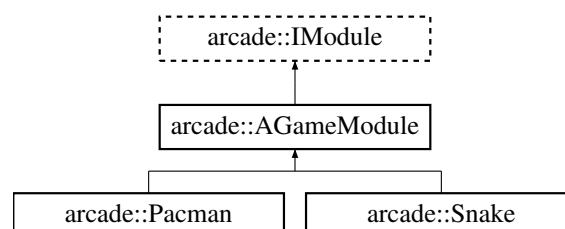
The documentation for this class was generated from the following files:

- [/home/tmendy/Documents/Tek2/OOP/Arcade/include/graphic/ADisplayModule.hpp](#)
- [/home/tmendy/Documents/Tek2/OOP/Arcade/lib/graphics/ADisplayModule.cpp](#)

## 6.2 arcade::AGameModule Class Reference

```
#include <AGameModule.hpp>
```

Inheritance diagram for `arcade::AGameModule`:



## Public Types

- enum `GameStatus` { `RUNNING` , `PAUSED` , `GAMEOVER` , `WIN` }

## Public Types inherited from `arcade::IModule`

- enum `KeyboardInput` {  
`UP` , `DOWN` , `LEFT` , `RIGHT` ,  
`SPACE` , `ENTER` , `ESCAPE` , `A` ,  
`B` , `C` , `D` , `E` ,  
`F` , `G` , `H` , `I` ,  
`J` , `K` , `L` , `M` ,  
`N` , `O` , `P` , `Q` ,  
`R` , `S` , `T` , `U` ,  
`V` , `W` , `X` , `Y` ,  
`Z` , `NONE` }  
*all the possible keyboard inputs*
- enum `ModuleType` { `GRAPHIC` , `GAME` , `CORE` }  
*all the possible module types*
- enum `LibName` {  
`SNAKE` , `NIBBLER` , `PACMAN` , `QIX` ,  
`CENTIPEDE` , `SOLARFOX` , `NCURSES` , `SDL` ,  
`SFML` , `OPENGL` , `UNKNOWN` }  
*all the possible library names*

## Public Member Functions

- `AGameModule` ()  
*Construct a new arcade::A Game Module::A Game Module object.*
- `~AGameModule` ()  
*Destroy the arcade::A Game Module::A Game Module object.*
- virtual void `init` ()=0
- virtual void `stop` ()=0
- void `setGameStatus` (`GameStatus` status)
- `GameStatus` `getDisplayStatus` () const  
*get the status of the game*
- virtual const `arcade::IModule::LibName` `getName` () const =0
- const `arcade::IModule::ModuleType` `getType` () const  
*get the name of the game library*
- void `sendInput` (`arcade::IModule::KeyboardInput` input)  
*receive input from the graphic module*
- `arcade::IModule::GameData` `sendGameData` ()  
*send the game data to the graphic module*

## Public Member Functions inherited from `arcade::IModule`

- `IModule` ()
- virtual `~IModule` ()
- virtual void `init` ()=0
- virtual void `stop` ()=0
- virtual const `LibName` `getName` () const =0
- virtual const `ModuleType` `getType` () const =0



### Protected Attributes

- [arcade::IModule::GameData \\_gameData](#)
- [arcade::IModule::KeyboardInput \\_input](#)
- [GameStatus \\_gameStatus](#)

## 6.2.1 Member Enumeration Documentation

### 6.2.1.1 GameStatus

```
enum arcade::AGameModule::GameStatus
```

#### Enumerator

RUNNING	
PAUSED	
GAMEOVER	
WIN	

## 6.2.2 Constructor & Destructor Documentation

### 6.2.2.1 AGameModule()

```
arcade::AGameModule::AGameModule ( )
```

Construct a new arcade::A Game Module::A Game Module object.

### 6.2.2.2 ~AGameModule()

```
arcade::AGameModule::~~AGameModule ( )
```

Destroy the arcade::A Game Module::A Game Module object.

## 6.2.3 Member Function Documentation

### 6.2.3.1 getDisplayStatus()

```
arcade::AGameModule::GameStatus arcade::AGameModule::getDisplayStatus ( ) const
```

get the status of the game

#### Returns

[arcade::AGameModule::GameStatus](#)

### 6.2.3.2 getName()

```
virtual const arcade::IModule::LibName arcade::AGameModule::getName ( ) const [pure virtual]
```

Implements [arcade::IModule](#).

Implemented in [arcade::Pacman](#), and [arcade::Snake](#).

### 6.2.3.3 getType()

```
const arcade::IModule::ModuleType arcade::AGameModule::getType ( ) const [virtual]
```

get the name of the game library

#### Returns

const [arcade::IModule::ModuleType](#)

Implements [arcade::IModule](#).

### 6.2.3.4 init()

```
virtual void arcade::AGameModule::init ( ) [pure virtual]
```

Implements [arcade::IModule](#).

Implemented in [arcade::Pacman](#), and [arcade::Snake](#).

### 6.2.3.5 sendGameData()

```
arcade::IModule::GameData arcade::AGameModule::sendGameData ( )
```

send the game data to the graphic module

#### Returns

[arcade::IModule::GameData](#)

### 6.2.3.6 sendInput()

```
void arcade::AGameModule::sendInput (
    arcade::IModule::KeyboardInput input )
```

receive input from the graphic module

#### Parameters

<i>input</i>	KeyboardInput
--------------	---------------

### 6.2.3.7 setGameStatus()

```
void arcade::AGameModule::setGameStatus (
    GameStatus status )
```

### 6.2.3.8 stop()

```
virtual void arcade::AGameModule::stop ( ) [pure virtual]
```

Implements [arcade::IModule](#).

Implemented in [arcade::Pacman](#), and [arcade::Snake](#).

## 6.2.4 Member Data Documentation

### 6.2.4.1 \_gameData

```
arcade::IModule::GameData arcade::AGameModule::_gameData [protected]
```

### 6.2.4.2 \_gameStatus

```
GameStatus arcade::AGameModule::_gameStatus [protected]
```

### 6.2.4.3 \_input

```
arcade::IModule::KeyboardInput arcade::AGameModule::_input [protected]
```

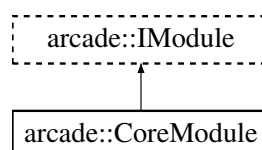
The documentation for this class was generated from the following files:

- /home/tmendy/Documents/Tek2/OOP/Arcade/include/game/[AGameModule.hpp](#)
- /home/tmendy/Documents/Tek2/OOP/Arcade/lib/games/[AGameModule.cpp](#)

## 6.3 arcade::CoreModule Class Reference

```
#include <CoreModule.hpp>
```

Inheritance diagram for arcade::CoreModule:



## Public Types

- enum [CoreStatus](#) { [RUNNING](#) , [SELECTION](#) }

## Public Types inherited from [arcade::IModule](#)

- enum [KeyboardInput](#) {  
[UP](#) , [DOWN](#) , [LEFT](#) , [RIGHT](#) ,  
[SPACE](#) , [ENTER](#) , [ESCAPE](#) , [A](#) ,  
[B](#) , [C](#) , [D](#) , [E](#) ,  
[F](#) , [G](#) , [H](#) , [I](#) ,  
[J](#) , [K](#) , [L](#) , [M](#) ,  
[N](#) , [O](#) , [P](#) , [Q](#) ,  
[R](#) , [S](#) , [T](#) , [U](#) ,  
[V](#) , [W](#) , [X](#) , [Y](#) ,  
[Z](#) , [NONE](#) }  
*all the possible keyboard inputs*
- enum [ModuleType](#) { [GRAPHIC](#) , [GAME](#) , [CORE](#) }  
*all the possible module types*
- enum [LibName](#) {  
[SNAKE](#) , [NIBBLER](#) , [PACMAN](#) , [QIX](#) ,  
[CENTIPEDE](#) , [SOLARFOX](#) , [NCURSES](#) , [SDL](#) ,  
[SFML](#) , [OPENGL](#) , [UNKNOWN](#) }  
*all the possible library names*

## Public Member Functions

- [CoreModule](#) ()  
*Construct a new arcade::Core Module::Core Module object.*
- [~CoreModule](#) ()  
*Destroy the arcade::Core Module::Core Module object.*
- void [init](#) ()  
*load the libraries in the given path*
- void [stop](#) ()  
*stop the core module*
- const [LibName](#) [getName](#) () const  
*get the name of the library*
- const [ModuleType](#) [getType](#) () const  
*get the type of the library*
- void [setCoreStatus](#) ([CoreStatus](#) status)  
*get the status of the core module*
- [CoreStatus](#) [getCoreStatus](#) () const  
*get the status of the core module*
- std::unique\_ptr< [ADisplayModule](#) > [getDisplayModule](#) ()  
*get the display module*
- std::unique\_ptr< [AGameModule](#) > [getGameModule](#) ()  
*get the game module*
- void [setModule](#) ([arcade::IModule::LibName](#) name, [arcade::IModule::ModuleType](#) type)  
*set graphic or game module to the core module*
- std::vector< std::string > [getLib](#) (std::string pathLib)

## Public Member Functions inherited from arcade::IModule

- [IModule](#) ()
- virtual [~IModule](#) ()
- virtual void [init](#) ()=0
- virtual void [stop](#) ()=0
- virtual const [LibName](#) [getName](#) () const =0
- virtual const [ModuleType](#) [getType](#) () const =0

## Protected Attributes

- [CoreStatus](#) [\\_coreStatus](#)
- std::unique\_ptr< [arcade::ADisplayModule](#) > [\\_displayModule](#)
- std::unique\_ptr< [arcade::AGameModule](#) > [\\_gameModule](#)

## 6.3.1 Member Enumeration Documentation

### 6.3.1.1 CoreStatus

```
enum arcade::CoreModule::CoreStatus
```

Enumerator

RUNNING	
SELECTION	

## 6.3.2 Constructor & Destructor Documentation

### 6.3.2.1 CoreModule()

```
arcade::CoreModule::CoreModule ( )
```

Construct a new arcade::Core Module::Core Module object.

### 6.3.2.2 ~CoreModule()

```
arcade::CoreModule::~~CoreModule ( )
```

Destroy the arcade::Core Module::Core Module object.

## 6.3.3 Member Function Documentation

### 6.3.3.1 getCoreStatus()

```
arcade::CoreModule::CoreStatus arcade::CoreModule::getCoreStatus ( ) const
```

get the status of the core module

Returns

CoreStatus

#### 6.3.3.2 getDisplayModule()

```
std::unique_ptr< arcade::ADisplayModule > arcade::CoreModule::getDisplayModule ( )
```

get the display module

Returns

```
std::unique_ptr<arcade::ADisplayModule>
```

#### 6.3.3.3 getGameModule()

```
std::unique_ptr< arcade::AGameModule > arcade::CoreModule::getGameModule ( )
```

get the game module

Returns

```
std::unique_ptr<arcade::AGameModule>
```

#### 6.3.3.4 getLib()

```
std::vector< std::string > arcade::CoreModule::getLib (
    std::string pathLib )
```

#### 6.3.3.5 getName()

```
const arcade::IModule::LibName arcade::CoreModule::getName ( ) const [virtual]
```

get the name of the library

Returns

```
const arcade::IModule::LibName
```

Implements [arcade::IModule](#).

#### 6.3.3.6 getType()

```
const arcade::IModule::ModuleType arcade::CoreModule::getType ( ) const [virtual]
```

get the type of the library

Returns

```
const arcade::IModule::ModuleType
```

Implements [arcade::IModule](#).

#### 6.3.3.7 init()

```
void arcade::CoreModule::init ( ) [virtual]
```

load the libraries in the given path

## Parameters

<i>pathLib</i>	path to the libraries
----------------	-----------------------

Implements [arcade::IModule](#).

**6.3.3.8 setCoreStatus()**

```
void arcade::CoreModule::setCoreStatus (
    CoreStatus status )
```

get the status of the core module

## Returns

CoreStatus

**6.3.3.9 setModule()**

```
void arcade::CoreModule::setModule (
    arcade::IModule::LibName name,
    arcade::IModule::ModuleType type )
```

set graphic or game module to the core module

## Parameters

<i>name</i>	of the module (snake, pacman, ncurses, sdl2)
<i>type</i>	of the module (graphic or game)

**6.3.3.10 stop()**

```
void arcade::CoreModule::stop ( ) [virtual]
```

stop the core module

Implements [arcade::IModule](#).

**6.3.4 Member Data Documentation****6.3.4.1 \_coreStatus**

```
CoreStatus arcade::CoreModule::_coreStatus [protected]
```

### 6.3.4.2 `_displayModule`

```
std::unique_ptr<arcade::ADisplayModule> arcade::CoreModule::_displayModule [protected]
```

### 6.3.4.3 `_gameModule`

```
std::unique_ptr<arcade::AGameModule> arcade::CoreModule::_gameModule [protected]
```

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

- [/home/tmendy/Documents/Tek2/OOP/Arcade/include/CoreModule.hpp](#)
- [/home/tmendy/Documents/Tek2/OOP/Arcade/lib/CoreModule.cpp](#)

## 6.4 `DLLoader< T >` Class Template Reference

```
#include <DLLoader.hpp>
```

### Public Member Functions

- [DLLoader](#) (const std::string &libPath)
- [~DLLoader](#) ()
- T \* [getInstance](#) (const std::string &funcName)

### Private Attributes

- void \* [handle](#)

## 6.4.1 Constructor & Destructor Documentation

### 6.4.1.1 `DLLoader()`

```
template<typename T >
DLLoader< T >::DLLoader (
    const std::string & libPath ) [inline]
```

### 6.4.1.2 `~DLLoader()`

```
template<typename T >
DLLoader< T >::~~DLLoader ( ) [inline]
```



## 6.4.2 Member Function Documentation

### 6.4.2.1 getInstance()

```
template<typename T >
T * DLLoader< T >::getInstance (
    const std::string & funcName ) [inline]
```

## 6.4.3 Member Data Documentation

### 6.4.3.1 handle

```
template<typename T >
void* DLLoader< T >::handle [private]
```

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

- [/home/tmendy/Documents/Tek2/OOP/Arcade/include/DLLoader.hpp](#)

## 6.5 arcade::IModule::GameData Struct Reference

information about the game from the game module to the graphic module

```
#include <IModule.hpp>
```

### Public Attributes

- `std::vector< std::vector< int > >` [display\\_info](#)
- `std::map< unsigned int, std::string >` [sprite\\_value](#)

### 6.5.1 Detailed Description

information about the game from the game module to the graphic module

## 6.5.2 Member Data Documentation

### 6.5.2.1 display\_info

```
std::vector<std::vector<int> > arcade::IModule::GameData::display_info
```

### 6.5.2.2 sprite\_value

```
std::map<unsigned int, std::string> arcade::IModule::GameData::sprite_value
```

The documentation for this struct was generated from the following file:

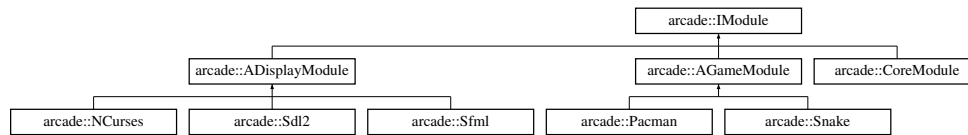
- [/home/tmendy/Documents/Tek2/OOP/Arcade/include/IModule.hpp](#)

## 6.6 arcade::IModule Class Reference

Interface for the modules.

```
#include <IModule.hpp>
```

Inheritance diagram for arcade::IModule:



### Classes

- struct [GameData](#)  
*information about the game from the game module to the graphic module*

### Public Types

- enum [KeyboardInput](#) {  
UP, DOWN, LEFT, RIGHT,  
SPACE, ENTER, ESCAPE, A,  
B, C, D, E,  
F, G, H, I,  
J, K, L, M,  
N, O, P, Q,  
R, S, T, U,  
V, W, X, Y,  
Z, NONE }
- all the possible keyboard inputs*
- enum [ModuleType](#) { GRAPHIC, GAME, CORE }
- all the possible module types*
- enum [LibName](#) {  
SNAKE, NIBBLER, PACMAN, QIX,  
CENTIPEDE, SOLARFOX, NCURSES, SDL,  
SFML, OPENGGL, UNKNOWN }
- all the possible library names*

### Public Member Functions

- [IModule](#) ()
- virtual [~IModule](#) ()
- virtual void [init](#) ()=0
- virtual void [stop](#) ()=0
- virtual const [LibName](#) [getName](#) () const =0
- virtual const [ModuleType](#) [getType](#) () const =0

### 6.6.1 Detailed Description

Interface for the modules.

## 6.6.2 Member Enumeration Documentation

### 6.6.2.1 KeyboardInput

enum `arcade::IModule::KeyboardInput`

all the possible keyboard inputs

Enumerator

UP	
DOWN	
LEFT	
RIGHT	
SPACE	
ENTER	
ESCAPE	
A	
B	
C	
D	
E	
F	
G	
H	
I	
J	
K	
L	
M	
N	
O	
P	
Q	
R	
S	
T	
U	
V	
W	
X	
Y	
Z	
NONE	

### 6.6.2.2 LibName

enum `arcade::IModule::LibName`

all the possible library names

### Enumerator

SNAKE	
NIBBLER	
PACMAN	
QIX	
CENTIPEDE	
SOLARFOX	
NCURSES	
SDL	
SFML	
OPENGL	
UNKNOWN	

### 6.6.2.3 ModuleType

```
enum arcade::IModule::ModuleType
```

all the possible module types

### Enumerator

GRAPHIC	
GAME	
CORE	

## 6.6.3 Constructor & Destructor Documentation

### 6.6.3.1 IModule()

```
arcade::IModule::IModule ( ) [inline]
```

### 6.6.3.2 ~IModule()

```
virtual arcade::IModule::~~IModule ( ) [inline], [virtual]
```

## 6.6.4 Member Function Documentation

### 6.6.4.1 getName()

```
virtual const LibName arcade::IModule::getName ( ) const [pure virtual]
```

Implemented in [arcade::CoreModule](#), [arcade::Pacman](#), [arcade::Snake](#), [arcade::NCurses](#), [arcade::Sdl2](#), [arcade::Sfml](#), [arcade::AGameModule](#), and [arcade::ADisplayModule](#).

### 6.6.4.2 getType()

```
virtual const ModuleType arcade::IModule::getType ( ) const [pure virtual]
```

Implemented in [arcade::CoreModule](#), [arcade::AGameModule](#), and [arcade::ADisplayModule](#).

### 6.6.4.3 init()

```
virtual void arcade::IModule::init ( ) [pure virtual]
```

Implemented in [arcade::CoreModule](#), [arcade::Pacman](#), [arcade::Snake](#), [arcade::NCurses](#), [arcade::Sdl2](#), [arcade::Sfml](#), [arcade::AGameModule](#), and [arcade::ADisplayModule](#).

### 6.6.4.4 stop()

```
virtual void arcade::IModule::stop ( ) [pure virtual]
```

Implemented in [arcade::CoreModule](#), [arcade::Pacman](#), [arcade::Snake](#), [arcade::NCurses](#), [arcade::Sdl2](#), [arcade::Sfml](#), [arcade::AGameModule](#), and [arcade::ADisplayModule](#).

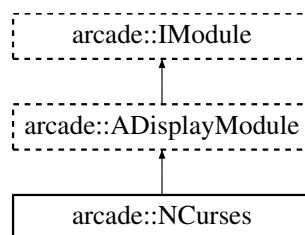
The documentation for this class was generated from the following file:

- [/home/tmendy/Documents/Tek2/OOP/Arcade/include/IModule.hpp](#)

## 6.7 arcade::NCurses Class Reference

```
#include <NCurses.hpp>
```

Inheritance diagram for arcade::NCurses:



### Public Member Functions

- [NCurses \(\)](#)
- [~NCurses \(\)](#)
- void [init \(\)](#)
- void [stop \(\)](#)
- void [display \(\)](#)
- const [arcade::IModule::LibName getName \(\)](#) const

## Public Member Functions inherited from [arcade::ADisplayModule](#)

- [ADisplayModule](#) ()  
*Construct a new [arcade::ADisplayModule::ADisplayModule](#) object.*
- [~ADisplayModule](#) ()  
*Destroy the [arcade::ADisplayModule::ADisplayModule](#) object.*
- virtual void [init](#) ()=0
- virtual void [stop](#) ()=0
- virtual void [display](#) ()=0
- void [setDisplayStatus](#) ([DisplayStatus](#) status)  
*set the status of the display module*
- [DisplayStatus](#) [getDisplayStatus](#) () const  
*return the status of the display module*
- virtual const [arcade::IModule::LibName](#) [getName](#) () const =0
- const [arcade::IModule::ModuleType](#) [getType](#) () const  
*return the type of the module*
- [arcade::IModule::KeyboardInput](#) [getInput](#) () const  
*get input from the user*
- void [sendGameData](#) ([arcade::IModule::GameData](#) data)  
*receive send data of the game module to the display module*

## Public Member Functions inherited from [arcade::IModule](#)

- [IModule](#) ()
- virtual [~IModule](#) ()
- virtual void [init](#) ()=0
- virtual void [stop](#) ()=0
- virtual const [LibName](#) [getName](#) () const =0
- virtual const [ModuleType](#) [getType](#) () const =0

## Additional Inherited Members

## Public Types inherited from [arcade::ADisplayModule](#)

- enum [DisplayStatus](#) {  
    [RUNNING](#) , [PAUSED](#) , [SELECTION](#) , [GAMEOVER](#) ,  
    [WIN](#) }

## Public Types inherited from [arcade::IModule](#)

- enum [KeyboardInput](#) {  
    [UP](#) , [DOWN](#) , [LEFT](#) , [RIGHT](#) ,  
    [SPACE](#) , [ENTER](#) , [ESCAPE](#) , [A](#) ,  
    [B](#) , [C](#) , [D](#) , [E](#) ,  
    [F](#) , [G](#) , [H](#) , [I](#) ,  
    [J](#) , [K](#) , [L](#) , [M](#) ,  
    [N](#) , [O](#) , [P](#) , [Q](#) ,  
    [R](#) , [S](#) , [T](#) , [U](#) ,  
    [V](#) , [W](#) , [X](#) , [Y](#) ,  
    [Z](#) , [NONE](#) }
- all the possible keyboard inputs*

- enum [ModuleType](#) { [GRAPHIC](#) , [GAME](#) , [CORE](#) }  
*all the possible module types*
- enum [LibName](#) {  
[SNAKE](#) , [NIBBLER](#) , [PACMAN](#) , [QIX](#) ,  
[CENTIPEDE](#) , [SOLARFOX](#) , [NCURSES](#) , [SDL](#) ,  
[SFML](#) , [OPENGL](#) , [UNKNOWN](#) }  
*all the possible library names*

### Protected Attributes inherited from [arcade::ADisplayModule](#)

- void \* [\\_window](#)
- void \* [\\_texture](#)
- void \* [\\_event](#)
- [arcade::IModule::GameData](#) [\\_gameData](#)
- [arcade::IModule::KeyboardInput](#) [\\_input](#)
- [DisplayStatus](#) [\\_displayStatus](#)

## 6.7.1 Constructor & Destructor Documentation

### 6.7.1.1 NCurses()

```
arcade::NCurses::NCurses ( )
```

### 6.7.1.2 ~NCurses()

```
arcade::NCurses::~~NCurses ( )
```

## 6.7.2 Member Function Documentation

### 6.7.2.1 display()

```
void arcade::NCurses::display ( ) [virtual]
```

Implements [arcade::ADisplayModule](#).

### 6.7.2.2 getName()

```
const arcade::IModule::LibName arcade::NCurses::getName ( ) const [virtual]
```

Implements [arcade::ADisplayModule](#).

### 6.7.2.3 init()

```
void arcade::NCurses::init ( ) [virtual]
```

Implements [arcade::ADisplayModule](#).

#### 6.7.2.4 stop()

```
void arcade::NCurses::stop ( ) [virtual]
```

Implements [arcade::ADisplayModule](#).

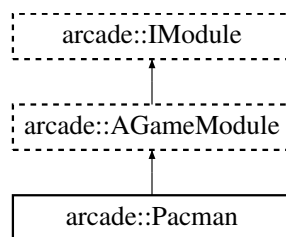
The documentation for this class was generated from the following files:

- [/home/tmendy/Documents/Tek2/OOP/Arcade/include/graphic/NCurses.hpp](#)
- [/home/tmendy/Documents/Tek2/OOP/Arcade/lib/graphics/ncurses/NCurses.cpp](#)

## 6.8 arcade::Pacman Class Reference

```
#include <Pacman.hpp>
```

Inheritance diagram for arcade::Pacman:



### Public Member Functions

- [Pacman \( \)](#)  
*Construct a new [arcade::Pacman::Pacman](#) object.*
- [~Pacman \( \)](#)  
*Destroy the [arcade::Pacman::Pacman](#) object.*
- void [init \( \)](#)
- void [stop \( \)](#)
- const [arcade::IModule::LibName getName \( \) const](#)  
*return the name of the game*

### Public Member Functions inherited from [arcade::AGameModule](#)

- [AGameModule \( \)](#)  
*Construct a new [arcade::AGameModule::AGameModule](#) object.*
- [~AGameModule \( \)](#)  
*Destroy the [arcade::AGameModule::AGameModule](#) object.*
- virtual void [init \( \)=0](#)
- virtual void [stop \( \)=0](#)
- void [setGameStatus \(GameStatus status\)](#)
- [GameStatus getDisplayStatus \( \) const](#)  
*get the status of the game*
- virtual const [arcade::IModule::LibName getName \( \) const =0](#)
- const [arcade::IModule::ModuleType getType \( \) const](#)  
*get the name of the game library*
- void [sendInput \(arcade::IModule::KeyboardInput input\)](#)  
*receive input from the graphic module*
- [arcade::IModule::GameData sendGameData \( \)](#)  
*send the game data to the graphic module*



## Public Member Functions inherited from [arcade::IModule](#)

- [IModule](#) ()
- virtual [~IModule](#) ()
- virtual void [init](#) ()=0
- virtual void [stop](#) ()=0
- virtual const [LibName](#) [getName](#) () const =0
- virtual const [ModuleType](#) [getType](#) () const =0

## Additional Inherited Members

## Public Types inherited from [arcade::AGameModule](#)

- enum [GameStatus](#) { [RUNNING](#) , [PAUSED](#) , [GAMEOVER](#) , [WIN](#) }

## Public Types inherited from [arcade::IModule](#)

- enum [KeyboardInput](#) {  
[UP](#) , [DOWN](#) , [LEFT](#) , [RIGHT](#) ,  
[SPACE](#) , [ENTER](#) , [ESCAPE](#) , [A](#) ,  
[B](#) , [C](#) , [D](#) , [E](#) ,  
[F](#) , [G](#) , [H](#) , [I](#) ,  
[J](#) , [K](#) , [L](#) , [M](#) ,  
[N](#) , [O](#) , [P](#) , [Q](#) ,  
[R](#) , [S](#) , [T](#) , [U](#) ,  
[V](#) , [W](#) , [X](#) , [Y](#) ,  
[Z](#) , [NONE](#) }  
*all the possible keyboard inputs*
- enum [ModuleType](#) { [GRAPHIC](#) , [GAME](#) , [CORE](#) }  
*all the possible module types*
- enum [LibName](#) {  
[SNAKE](#) , [NIBBLER](#) , [PACMAN](#) , [QIX](#) ,  
[CENTIPEDE](#) , [SOLARFOX](#) , [NCURSES](#) , [SDL](#) ,  
[SFML](#) , [OPENGL](#) , [UNKNOWN](#) }  
*all the possible library names*

## Protected Attributes inherited from [arcade::AGameModule](#)

- [arcade::IModule::GameData](#) [\\_gameData](#)
- [arcade::IModule::KeyboardInput](#) [\\_input](#)
- [GameStatus](#) [\\_gameStatus](#)

## 6.8.1 Constructor & Destructor Documentation

### 6.8.1.1 Pacman()

```
arcade::Pacman::Pacman ( )
```

Construct a new [arcade::Pacman::Pacman](#) object.

### 6.8.1.2 ~Pacman()

```
arcade::Pacman::~~Pacman ( )
```

Destroy the [arcade::Pacman::Pacman](#) object.

## 6.8.2 Member Function Documentation

### 6.8.2.1 getName()

```
const arcade::IModule::LibName arcade::Pacman::getName ( ) const [virtual]
```

return the name of the game

Returns

const [arcade::IModule::LibName](#)

Implements [arcade::AGameModule](#).

### 6.8.2.2 init()

```
void arcade::Pacman::init ( ) [inline], [virtual]
```

Implements [arcade::AGameModule](#).

### 6.8.2.3 stop()

```
void arcade::Pacman::stop ( ) [inline], [virtual]
```

Implements [arcade::AGameModule](#).

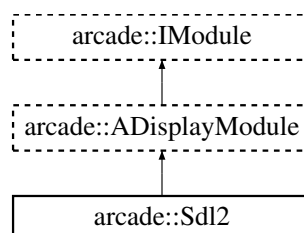
The documentation for this class was generated from the following files:

- [/home/tmendy/Documents/Tek2/OOP/Arcade/include/game/Pacman.hpp](#)
- [/home/tmendy/Documents/Tek2/OOP/Arcade/lib/games/pacman/Pacman.cpp](#)

## 6.9 arcade::Sdl2 Class Reference

```
#include <Sdl2.hpp>
```

Inheritance diagram for arcade::Sdl2:



**Public Member Functions**

- [Sdl2](#) ()
- [~Sdl2](#) ()
- void [init](#) ()  
*initailize the SDL2 module and create a window*
- void [stop](#) ()  
*stop the SDL2 module and destroy the window*
- void [display](#) ()  
*display information on the window*
- const [arcade::IModule::LibName](#) [getName](#) () const  
*return the name of the module*

**Public Member Functions inherited from [arcade::ADisplayModule](#)**

- [ADisplayModule](#) ()  
*Construct a new [arcade::ADisplayModule::ADisplayModule](#) object.*
- [~ADisplayModule](#) ()  
*Destroy the [arcade::ADisplayModule::ADisplayModule](#) object.*
- virtual void [init](#) ()=0
- virtual void [stop](#) ()=0
- virtual void [display](#) ()=0
- void [setDisplayStatus](#) ([DisplayStatus](#) status)  
*set the status of the display module*
- [DisplayStatus](#) [getDisplayStatus](#) () const  
*return the status of the display module*
- virtual const [arcade::IModule::LibName](#) [getName](#) () const =0
- const [arcade::IModule::ModuleType](#) [getType](#) () const  
*return the type of the module*
- [arcade::IModule::KeyboardInput](#) [getInput](#) () const  
*get input from the user*
- void [sendGameData](#) ([arcade::IModule::GameData](#) data)  
*receive send data of the game module to the display module*

**Public Member Functions inherited from [arcade::IModule](#)**

- [IModule](#) ()
- virtual [~IModule](#) ()
- virtual void [init](#) ()=0
- virtual void [stop](#) ()=0
- virtual const [LibName](#) [getName](#) () const =0
- virtual const [ModuleType](#) [getType](#) () const =0

**Additional Inherited Members****Public Types inherited from [arcade::ADisplayModule](#)**

- enum [DisplayStatus](#) {  
    [RUNNING](#) , [PAUSED](#) , [SELECTION](#) , [GAMEOVER](#) ,  
    [WIN](#) }

## Public Types inherited from [arcade::IModule](#)

- enum [KeyboardInput](#) {  
[UP](#) , [DOWN](#) , [LEFT](#) , [RIGHT](#) ,  
[SPACE](#) , [ENTER](#) , [ESCAPE](#) , [A](#) ,  
[B](#) , [C](#) , [D](#) , [E](#) ,  
[F](#) , [G](#) , [H](#) , [I](#) ,  
[J](#) , [K](#) , [L](#) , [M](#) ,  
[N](#) , [O](#) , [P](#) , [Q](#) ,  
[R](#) , [S](#) , [T](#) , [U](#) ,  
[V](#) , [W](#) , [X](#) , [Y](#) ,  
[Z](#) , [NONE](#) }  
*all the possible keyboard inputs*
- enum [ModuleType](#) { [GRAPHIC](#) , [GAME](#) , [CORE](#) }  
*all the possible module types*
- enum [LibName](#) {  
[SNAKE](#) , [NIBBLER](#) , [PACMAN](#) , [QIX](#) ,  
[CENTIPEDE](#) , [SOLARFOX](#) , [NCURSES](#) , [SDL](#) ,  
[SFML](#) , [OPENGL](#) , [UNKNOWN](#) }  
*all the possible library names*

## Protected Attributes inherited from [arcade::ADisplayModule](#)

- void \* [\\_window](#)
- void \* [\\_texture](#)
- void \* [\\_event](#)
- [arcade::IModule::GameData](#) [\\_gameData](#)
- [arcade::IModule::KeyboardInput](#) [\\_input](#)
- [DisplayStatus](#) [\\_displayStatus](#)

## 6.9.1 Constructor & Destructor Documentation

### 6.9.1.1 Sdl2()

```
arcade::Sdl2::Sdl2 ( )
```

### 6.9.1.2 ~Sdl2()

```
arcade::Sdl2::~~Sdl2 ( )
```

## 6.9.2 Member Function Documentation

### 6.9.2.1 display()

```
void arcade::Sdl2::display ( ) [virtual]
```

display information on the window

Implements [arcade::ADisplayModule](#).

### 6.9.2.2 getName()

```
const arcade::IModule::LibName arcade::Sdl2::getName ( ) const [virtual]
```

return the name of the module

Returns

const [arcade::IModule::LibName](#)

Implements [arcade::ADisplayModule](#).

### 6.9.2.3 init()

```
void arcade::Sdl2::init ( ) [virtual]
```

initailize the SDL2 module and create a window

Implements [arcade::ADisplayModule](#).

### 6.9.2.4 stop()

```
void arcade::Sdl2::stop ( ) [virtual]
```

stop the SDL2 module and destroy the window

Implements [arcade::ADisplayModule](#).

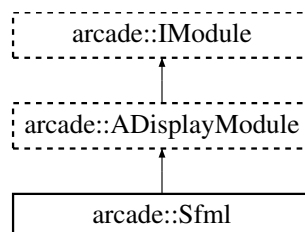
The documentation for this class was generated from the following files:

- [/home/tmendy/Documents/Tek2/OOP/Arcade/include/graphic/Sdl2.hpp](#)
- [/home/tmendy/Documents/Tek2/OOP/Arcade/lib/graphics/sdl2/Sdl2.cpp](#)

## 6.10 arcade::Sfml Class Reference

```
#include <Sfml.hpp>
```

Inheritance diagram for arcade::Sfml:



## Public Member Functions

- [Sfml](#) ()
- [~Sfml](#) ()
- [arcade::IModule::KeyboardInput](#) getInput ()
- void [init](#) ()
- void [stop](#) ()
- void [display](#) ()
- const [arcade::IModule::LibName](#) getName () const

## Public Member Functions inherited from [arcade::ADisplayModule](#)

- [ADisplayModule](#) ()  
*Construct a new [arcade::ADisplayModule::ADisplayModule](#) object.*
- [~ADisplayModule](#) ()  
*Destroy the [arcade::ADisplayModule::ADisplayModule](#) object.*
- virtual void [init](#) ()=0
- virtual void [stop](#) ()=0
- virtual void [display](#) ()=0
- void [setDisplayStatus](#) ([DisplayStatus](#) status)  
*set the status of the display module*
- [DisplayStatus](#) [getDisplayStatus](#) () const  
*return the status of the display module*
- virtual const [arcade::IModule::LibName](#) getName () const =0
- const [arcade::IModule::ModuleType](#) getType () const  
*return the type of the module*
- [arcade::IModule::KeyboardInput](#) getInput () const  
*get input from the user*
- void [sendGameData](#) ([arcade::IModule::GameData](#) data)  
*receive send data of the game module to the display module*

## Public Member Functions inherited from [arcade::IModule](#)

- [IModule](#) ()
- virtual [~IModule](#) ()
- virtual void [init](#) ()=0
- virtual void [stop](#) ()=0
- virtual const [LibName](#) getName () const =0
- virtual const [ModuleType](#) getType () const =0

## Private Attributes

- sf::Texture [\\_texture](#)
- sf::Font [\\_font](#)

## Additional Inherited Members

## Public Types inherited from [arcade::ADisplayModule](#)

- enum [DisplayStatus](#) {  
    [RUNNING](#) , [PAUSED](#) , [SELECTION](#) , [GAMEOVER](#) ,  
    [WIN](#) }

## Public Types inherited from [arcade::IModule](#)

- enum [KeyboardInput](#) {  
UP, DOWN, LEFT, RIGHT,  
SPACE, ENTER, ESCAPE, A,  
B, C, D, E,  
F, G, H, I,  
J, K, L, M,  
N, O, P, Q,  
R, S, T, U,  
V, W, X, Y,  
Z, NONE }  
*all the possible keyboard inputs*
- enum [ModuleType](#) { [GRAPHIC](#), [GAME](#), [CORE](#) }  
*all the possible module types*
- enum [LibName](#) {  
SNAKE, NIBBLER, PACMAN, QIX,  
CENTIPEDE, SOLARFOX, NCURSES, SDL,  
SFML, OPENGL, UNKNOWN }  
*all the possible library names*

## Protected Attributes inherited from [arcade::ADisplayModule](#)

- void \* [\\_window](#)
- void \* [\\_texture](#)
- void \* [\\_event](#)
- [arcade::IModule::GameData](#) [\\_gameData](#)
- [arcade::IModule::KeyboardInput](#) [\\_input](#)
- [DisplayStatus](#) [\\_displayStatus](#)

## 6.10.1 Constructor & Destructor Documentation

### 6.10.1.1 Sfml()

```
arcade::Sfml::Sfml ( )
```

### 6.10.1.2 ~Sfml()

```
arcade::Sfml::~~Sfml ( )
```

## 6.10.2 Member Function Documentation

### 6.10.2.1 display()

```
void arcade::Sfml::display ( ) [virtual]
```

Implements [arcade::ADisplayModule](#).

### 6.10.2.2 getInput()

```
arcade::IModule::KeyboardInput arcade::Sfml::getInput ( )
```

### 6.10.2.3 getName()

```
const arcade::IModule::LibName arcade::Sfml::getName ( ) const [virtual]
```

Implements [arcade::ADisplayModule](#).

### 6.10.2.4 init()

```
void arcade::Sfml::init ( ) [virtual]
```

Implements [arcade::ADisplayModule](#).

### 6.10.2.5 stop()

```
void arcade::Sfml::stop ( ) [virtual]
```

Implements [arcade::ADisplayModule](#).

## 6.10.3 Member Data Documentation

### 6.10.3.1 \_font

```
sf::Font arcade::Sfml::_font [private]
```

### 6.10.3.2 \_texture

```
sf::Texture arcade::Sfml::_texture [private]
```

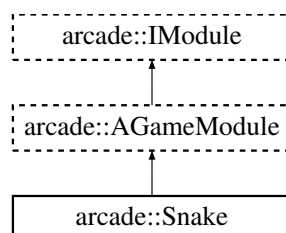
The documentation for this class was generated from the following files:

- [/home/tmendy/Documents/Tek2/OOP/Arcade/include/graphic/Sfml.hpp](#)
- [/home/tmendy/Documents/Tek2/OOP/Arcade/lib/graphics/sfml/Sfml.cpp](#)

## 6.11 arcade::Snake Class Reference

```
#include <Snake.hpp>
```

Inheritance diagram for arcade::Snake:





## Public Member Functions

- [Snake](#) ()  
*Construct a new [arcade::Snake::Snake](#) object.*
- [~Snake](#) ()  
*Destroy the [arcade::Snake::Snake](#) object.*
- void [init](#) ()  
*init the game*
- void [stop](#) ()  
*stop the game*
- const [arcade::IModule::LibName](#) [getName](#) () const  
*return the name of the game library*

## Public Member Functions inherited from [arcade::AGameModule](#)

- [AGameModule](#) ()  
*Construct a new [arcade::A Game Module::A Game Module](#) object.*
- [~AGameModule](#) ()  
*Destroy the [arcade::A Game Module::A Game Module](#) object.*
- virtual void [init](#) ()=0
- virtual void [stop](#) ()=0
- void [setGameStatus](#) ([GameStatus](#) status)
- [GameStatus](#) [getDisplayStatus](#) () const  
*get the status of the game*
- virtual const [arcade::IModule::LibName](#) [getName](#) () const =0
- const [arcade::IModule::ModuleType](#) [getType](#) () const  
*get the name of the game library*
- void [sendInput](#) ([arcade::IModule::KeyboardInput](#) input)  
*receive input from the graphic module*
- [arcade::IModule::GameData](#) [sendGameData](#) ()  
*send the game data to the graphic module*

## Public Member Functions inherited from [arcade::IModule](#)

- [IModule](#) ()
- virtual [~IModule](#) ()
- virtual void [init](#) ()=0
- virtual void [stop](#) ()=0
- virtual const [LibName](#) [getName](#) () const =0
- virtual const [ModuleType](#) [getType](#) () const =0

## Additional Inherited Members

## Public Types inherited from [arcade::AGameModule](#)

- enum [GameStatus](#) { [RUNNING](#) , [PAUSED](#) , [GAMEOVER](#) , [WIN](#) }

## Public Types inherited from [arcade::IModule](#)

- enum [KeyboardInput](#) {  
[UP](#) , [DOWN](#) , [LEFT](#) , [RIGHT](#) ,  
[SPACE](#) , [ENTER](#) , [ESCAPE](#) , [A](#) ,  
[B](#) , [C](#) , [D](#) , [E](#) ,  
[F](#) , [G](#) , [H](#) , [I](#) ,  
[J](#) , [K](#) , [L](#) , [M](#) ,  
[N](#) , [O](#) , [P](#) , [Q](#) ,  
[R](#) , [S](#) , [T](#) , [U](#) ,  
[V](#) , [W](#) , [X](#) , [Y](#) ,  
[Z](#) , [NONE](#) }
- all the possible keyboard inputs*
- enum [ModuleType](#) { [GRAPHIC](#) , [GAME](#) , [CORE](#) }
- all the possible module types*
- enum [LibName](#) {  
[SNAKE](#) , [NIBBLER](#) , [PACMAN](#) , [QIX](#) ,  
[CENTIPEDE](#) , [SOLARFOX](#) , [NCURSES](#) , [SDL](#) ,  
[SFML](#) , [OPENGL](#) , [UNKNOWN](#) }
- all the possible library names*

## Protected Attributes inherited from [arcade::AGameModule](#)

- [arcade::IModule::GameData](#) \_gameData
- [arcade::IModule::KeyboardInput](#) \_input
- [GameStatus](#) \_gameStatus

## 6.11.1 Constructor & Destructor Documentation

### 6.11.1.1 Snake()

```
arcade::Snake::Snake ( )
```

Construct a new [arcade::Snake::Snake](#) object.

### 6.11.1.2 ~Snake()

```
arcade::Snake::~~Snake ( )
```

Destroy the [arcade::Snake::Snake](#) object.

## 6.11.2 Member Function Documentation

### 6.11.2.1 getName()

```
const arcade::IModule::LibName arcade::Snake::getName ( ) const [virtual]
```

return the name of the game library

Returns

const [arcade::IModule::LibName](#)

Implements [arcade::AGameModule](#).

### 6.11.2.2 init()

```
void arcade::Snake::init ( ) [virtual]
```

init the game

Implements [arcade::AGameModule](#).

### 6.11.2.3 stop()

```
void arcade::Snake::stop ( ) [virtual]
```

stop the game

Implements [arcade::AGameModule](#).

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

- [/home/tmendy/Documents/Tek2/OOP/Arcade/include/game/Snake.hpp](#)
- [/home/tmendy/Documents/Tek2/OOP/Arcade/lib/games/snake/Snake.cpp](#)



# Chapter 7

## File Documentation

### 7.1 /home/tmendy/Documents/Tek2/OOP/Arcade/include/CoreModule.hpp File Reference

```
#include "ADisplayModule.hpp"
#include "AGameModule.hpp"
#include "IModule.hpp"
#include <memory>
#include <iostream>
#include <dirent.h>
```

#### Classes

- class [arcade::CoreModule](#)

#### Namespaces

- namespace [arcade](#)  
*namespace for the arcade project*

### 7.2 CoreModule.hpp

[Go to the documentation of this file.](#)

```
00001 /*
00002 ** EPITECH PROJECT, 2024
00003 ** Arcade
00004 ** File description:
00005 ** CoreModule
00006 */
00007
00008 #ifndef COREMODULE_HPP_
00009 #define COREMODULE_HPP_
00010
00011 #include "ADisplayModule.hpp"
00012 #include "AGameModule.hpp"
00013 #include "IModule.hpp"
00014 #include <memory>
00015 #include <iostream>
00016 #include <dirent.h>
```

```

00017
00018 namespace arcade {
00019 class CoreModule : virtual public arcade::IModule {
00020 public:
00021     CoreModule();
00022     ~CoreModule();
00023     void init();
00024     void stop();
00025     const LibName getName() const;
00026     const ModuleType getType() const;
00027     enum CoreStatus { RUNNING, SELECTION };
00028     void setCoreStatus(CoreStatus status);
00029     CoreStatus getCoreStatus() const;
00030     std::unique_ptr<ADisplayModule> getDisplayModule();
00031     std::unique_ptr<AGameModule> getGameModule();
00032     void setModule(arcade::IModule::LibName name,
00033                   arcade::IModule::ModuleType type);
00034     std::vector<std::string> getLib(std::string pathLib);
00035
00036 protected:
00037     CoreStatus _coreStatus;
00038     std::unique_ptr<arcade::ADisplayModule> _displayModule;
00039     std::unique_ptr<arcade::AGameModule> _gameModule;
00040 };
00041 }; // namespace arcade
00042
00043 #endif /* !COREMODULE_HPP_ */

```

## 7.3 /home/tmendy/Documents/Tek2/OOP/Arcade/include/DLLoader.hpp File Reference

```

#include <dlfcn.h>
#include <iostream>

```

### Classes

- class [DLLoader< T >](#)

## 7.4 DLLoader.hpp

[Go to the documentation of this file.](#)

```

00001 #include <dlfcn.h>
00002 #include <iostream>
00003
00004 template <typename T>
00005 class DLLoader {
00006 private:
00007     void *handle;
00008
00009 public:
00010     DLLoader(const std::string &libPath)
00011     {
00012         handle = dlopen(libPath.c_str(), RTLD_LAZY);
00013         if (!handle) {
00014             std::cerr << dlerror() << std::endl;
00015             exit(1);
00016         }
00017     }
00018
00019     ~DLLoader()
00020     {
00021         if (handle)
00022             dlclose(handle);
00023     }
00024
00025     T *getInstance(const std::string &funcName)
00026     {
00027         void *sym = dlsym(handle, funcName.c_str());

```

```

00028     if (!sym) {
00029         std::cerr << dlerror() << std::endl;
00030         exit(1);
00031     }
00032     return reinterpret_cast<T * (*)()>(sym)();
00033 }
00034 };

```

## 7.5 /home/tmendy/Documents/Tek2/OOP/Arcade/include/game/AGameModule.hpp File Reference

```
#include "IModule.hpp"
```

### Classes

- class [arcade::AGameModule](#)

### Namespaces

- namespace [arcade](#)  
*namespace for the arcade project*

## 7.6 AGameModule.hpp

[Go to the documentation of this file.](#)

```

00001 /*
00002 ** EPITECH PROJECT, 2024
00003 ** Arcade
00004 ** File description:
00005 ** AGameModule
00006 */
00007
00008 #ifndef AGAMEMODULE_HPP_
00009 #define AGAMEMODULE_HPP_
00010
00011 #include "IModule.hpp"
00012
00013 namespace arcade {
00014     class AGameModule : virtual public arcade::IModule {
00015     public:
00016         enum GameStatus { RUNNING, PAUSED, GAMEOVER, WIN };
00017         AGameModule();
00018         ~AGameModule();
00019         virtual void init() = 0;
00020         virtual void stop() = 0;
00021
00022         void setGameStatus(GameStatus status);
00023         GameStatus getDisplayStatus() const;
00024
00025         virtual const arcade::IModule::LibName getName() const = 0;
00026         const arcade::IModule::ModuleType getType() const;
00027
00028         void sendInput(arcade::IModule::KeyboardInput input);
00029         arcade::IModule::GameData sendGameData();
00030
00031     protected:
00032         arcade::IModule::GameData _gameData;
00033         arcade::IModule::KeyboardInput _input;
00034         GameStatus _gameStatus;
00035     };
00036 }; // namespace arcade
00037
00038 #endif /* !AGAMEMODULE_HPP_ */

```

## 7.7 /home/tmendy/Documents/Tek2/OOP/Arcade/include/game/↵ Pacman.hpp File Reference

```
#include "AGameModule.hpp"
```

### Classes

- class [arcade::Pacman](#)

### Namespaces

- namespace [arcade](#)  
*namespace for the arcade project*

## 7.8 Pacman.hpp

[Go to the documentation of this file.](#)

```
00001 /*
00002 ** EPITECH PROJECT, 2024
00003 ** Arcade
00004 ** File description:
00005 ** Pacman
00006 */
00007
00008 #ifndef PACMAN_HPP_
00009 #define PACMAN_HPP_
00010
00011 #include "AGameModule.hpp"
00012
00013 namespace arcade {
00014 class Pacman : virtual public arcade::AGameModule {
00015 public:
00016     Pacman();
00017     ~Pacman();
00018     void init(){};
00019     void stop(){};
00020     const arcade::IModule::LibName getName() const;
00021
00022 protected:
00023 private:
00024 };
00025 }; // namespace arcade
00026
00027 #endif /* !PACMAN_HPP_ */
```

## 7.9 /home/tmendy/Documents/Tek2/OOP/Arcade/include/game/↵ Snake.hpp File Reference

```
#include "AGameModule.hpp"
```

### Classes

- class [arcade::Snake](#)



## Namespaces

- namespace [arcade](#)  
*namespace for the arcade project*

## 7.10 Snake.hpp

[Go to the documentation of this file.](#)

```
00001 /*
00002 ** EPITECH PROJECT, 2024
00003 ** Arcade
00004 ** File description:
00005 ** Snake
00006 */
00007
00008 #ifndef SNAKE_HPP_
00009 #define SNAKE_HPP_
00010
00011 #include "AGameModule.hpp"
00012
00013 namespace arcade {
00014 class Snake : virtual public arcade::AGameModule {
00015 public:
00016     Snake();
00017     ~Snake();
00018     void init();
00019     void stop();
00020     const arcade::IModule::LibName getName() const;
00021
00022 protected:
00023 private:
00024 };
00025 }; // namespace arcade
00026
00027 #endif /* !SNAKE_HPP_ */
```

## 7.11 /home/tmendy/Documents/Tek2/OOP/Arcade/include/graphic/↵ ADisplayModule.hpp File Reference

```
#include "IModule.hpp"
```

## Classes

- class [arcade::ADisplayModule](#)

## Namespaces

- namespace [arcade](#)  
*namespace for the arcade project*

## 7.12 ADisplayModule.hpp

[Go to the documentation of this file.](#)

```

00001  /*
00002  ** EPITECH PROJECT, 2024
00003  ** bsArcade
00004  ** File description:
00005  ** ADisplayModule
00006  */
00007
00008 #ifndef ADISPLAYMODULE_HPP_
00009 #define ADISPLAYMODULE_HPP_
00010
00011 #include "IModule.hpp"
00012
00013 namespace arcade {
00014 class ADisplayModule : virtual public arcade::IModule {
00015 public:
00016     enum DisplayStatus { RUNNING, PAUSED, SELECTION, GAMEOVER, WIN };
00017     ADisplayModule();
00018     ~ADisplayModule();
00019     virtual void init() = 0;
00020     virtual void stop() = 0;
00021
00022     virtual void display() = 0;
00023     void setDisplayStatus(DisplayStatus status);
00024     DisplayStatus getDisplayStatus() const;
00025
00026     virtual const arcade::IModule::LibName getName() const = 0;
00027     const arcade::IModule::ModuleType getType() const;
00028
00029     arcade::IModule::KeyboardInput getInput() const;
00030     void sendGameData(arcade::IModule::GameData data);
00031
00032 protected:
00033     void *_window;
00034     void *_texture;
00035     void *_event;
00036     arcade::IModule::GameData _gameData;
00037     arcade::IModule::KeyboardInput _input;
00038     DisplayStatus _displayStatus;
00039 };
00040 }; // namespace arcade
00041
00042 #endif /* !ADISPLAYMODULE_HPP_ */

```

## 7.13 /home/tmendy/Documents/Tek2/OOP/Arcade/include/graphic/↵ NCurses.hpp File Reference

```

#include "ADisplayModule.hpp"
#include <ncurses.h>

```

### Classes

- class [arcade::NCurses](#)

### Namespaces

- namespace [arcade](#)  
*namespace for the arcade project*

## 7.14 NCurses.hpp

[Go to the documentation of this file.](#)

```
00001 /*
00002  ** EPITECH PROJECT, 2024
00003  ** Arcade
00004  ** File description:
00005  ** NCurses
00006  */
00007
00008 #ifndef NCURSES_HPP_
00009 #define NCURSES_HPP_
00010
00011 #include "ADisplayModule.hpp"
00012 #include <ncurses.h>
00013
00014 namespace arcade {
00015 class NCurses : virtual public arcade::ADisplayModule {
00016 public:
00017     NCurses();
00018     ~NCurses();
00019     void init();
00020     void stop();
00021     void display();
00022     const arcade::IModule::LibName getName() const;
00023
00024 protected:
00025 private:
00026 };
00027 }; // namespace arcade
00028
00029 #endif /* !NCURSES_HPP_ */
```

## 7.15 /home/tmendy/Documents/Tek2/OOP/Arcade/include/graphic/↵ Sdl2.hpp File Reference

```
#include "ADisplayModule.hpp"
#include <SDL2/SDL.h>
```

### Classes

- class [arcade::Sdl2](#)

### Namespaces

- namespace [arcade](#)  
*namespace for the arcade project*

## 7.16 Sdl2.hpp

[Go to the documentation of this file.](#)

```
00001 /*
00002  ** EPITECH PROJECT, 2024
00003  ** Arcade
00004  ** File description:
00005  ** SDL2
00006  */
00007
00008 #ifndef SDL2_HPP_
00009 #define SDL2_HPP_
00010
```

```

00011 #include "ADisplayModule.hpp"
00012 #include <SDL2/SDL.h>
00013
00014 namespace arcade {
00015 class Sdl2 : virtual public arcade::ADisplayModule {
00016 public:
00017     Sdl2();
00018     ~Sdl2();
00019     void init();
00020     void stop();
00021     void display();
00022     const arcade::IModule::LibName getName() const;
00023
00024 protected:
00025 private:
00026 };
00027 }; // namespace arcade
00028
00029 #endif /* !SDL2_HPP_ */

```

## 7.17 /home/tmendy/Documents/Tek2/OOP/Arcade/include/graphic/Sfml.hpp File Reference

```

#include "ADisplayModule.hpp"
#include <SFML/Graphics.hpp>

```

### Classes

- class [arcade::Sfml](#)

### Namespaces

- namespace [arcade](#)  
*namespace for the arcade project*

## 7.18 Sfml.hpp

[Go to the documentation of this file.](#)

```

00001 /*
00002 ** EPITECH PROJECT, 2024
00003 ** Arcade
00004 ** File description:
00005 ** SFML
00006 */
00007
00008 #ifndef SFML_HPP_
00009 #define SFML_HPP_
00010
00011 #include "ADisplayModule.hpp"
00012 #include <SFML/Graphics.hpp>
00013
00014 namespace arcade {
00015 class Sfml : virtual public arcade::ADisplayModule {
00016 public:
00017     Sfml();
00018     ~Sfml();
00019
00020     arcade::IModule::KeyboardInput getInput();
00021     void init();
00022     void stop();
00023     void display();
00024     const arcade::IModule::LibName getName() const;
00025
00026 protected:
00027 private:
00028     sf::Texture _texture;
00029     sf::Font _font;
00030 };
00031 }; // namespace arcade
00032
00033 #endif /* !SFML_HPP_ */

```

## 7.19 /home/tmendy/Documents/Tek2/OOP/Arcade/include/IModule.hpp File Reference

```
#include <map>
#include <string>
#include <vector>
```

### Classes

- class [arcade::IModule](#)  
*Interface for the modules.*
- struct [arcade::IModule::GameData](#)  
*information about the game from the game module to the graphic module*

### Namespaces

- namespace [arcade](#)  
*namespace for the arcade project*

## 7.20 IModule.hpp

[Go to the documentation of this file.](#)

```
00001 /*
00002 ** EPITECH PROJECT, 2024
00003 ** Arcade
00004 ** File description:
00005 ** IModule
00006 */
00007
00008 #ifndef IMODULE_HPP_
00009 #define IMODULE_HPP_
00010
00011 #include <map> // Include the necessary header file
00012 #include <string>
00013 #include <vector>
00014
00019 namespace arcade {
00024 class IModule {
00025 public:
00026     IModule(){};
00027     virtual ~IModule(){};
00028     virtual void init() = 0;
00029     virtual void stop() = 0;
00035     struct GameData {
00036         std::vector<std::vector<int>> display_info;
00037         std::map<unsigned int, std::string> sprite_value;
00038     };
00039
00044     enum KeyboardInput {
00045         UP,
00046         DOWN,
00047         LEFT,
00048         RIGHT,
00049         SPACE,
00050         ENTER,
00051         ESCAPE,
00052         A,
00053         B,
00054         C,
00055         D,
00056         E,
00057         F,
00058         G,
00059         H,
```

```

00060     I,
00061     J,
00062     K,
00063     L,
00064     M,
00065     N,
00066     O,
00067     P,
00068     Q,
00069     R,
00070     S,
00071     T,
00072     U,
00073     V,
00074     W,
00075     X,
00076     Y,
00077     Z,
00078     NONE
00079 };
00080
00085     enum ModuleType { GRAPHIC, GAME, CORE };
00086
00091     enum LibName {
00092         // GAME
00093         SNAKE,
00094         NIBBLER,
00095         PACMAN,
00096         QIX,
00097         CENTIPEDE,
00098         SOLARFOX,
00099         // GRAPHIC
00100         NCURSES,
00101         SDL,
00102         SFML,
00103         OPENGL,
00104         // CORE
00105         UNKNOWN
00106     };
00107     virtual const LibName getName() const = 0;
00108     virtual const ModuleType getType() const = 0;
00109 };
00110 }; // namespace arcade
00111
00112 #endif /* !MODULE_HPP_ */

```

## 7.21 /home/tmendy/Documents/Tek2/OOP/Arcade/include/Macros.hpp

### File Reference

#### Macros

- #define OK 0
- #define KO 84
- #define ERROR (-1)

### 7.21.1 Macro Definition Documentation

#### 7.21.1.1 ERROR

```
#define ERROR (-1)
```

#### 7.21.1.2 KO

```
#define KO 84
```

### 7.21.1.3 OK

```
#define OK 0
```

## 7.22 Macros.hpp

[Go to the documentation of this file.](#)

```
00001 /*
00002 ** EPITECH PROJECT, 2024
00003 ** Arcade
00004 ** File description:
00005 ** Macros
00006 */
00007
00008 #ifndef MACROS_HPP_
00009 #define MACROS_HPP_
00010 #define OK 0
00011 #define KO 84
00012 #define ERROR (-1)
00013
00014 #endif /* !MACROS_HPP_ */
```

## 7.23 /home/tmendy/Documents/Tek2/OOP/Arcade/lib/CoreModule.cpp File Reference

```
#include "CoreModule.hpp"
#include <NCurses.hpp>
#include <Pacman.hpp>
#include <Sdl2.hpp>
#include <Snake.hpp>
```

### Functions

- `std::vector< std::string > getLib (std::string pathLib)`  
*get the list of libraries in the given path*

### 7.23.1 Function Documentation

#### 7.23.1.1 [getLib\(\)](#)

```
std::vector< std::string > getLib (
    std::string pathLib )
```

get the list of libraries in the given path

#### Parameters

<i>pathLib</i>	path to the libraries
----------------	-----------------------

**Returns**

std::vector<std::string> list of libraries

## 7.24 `/home/tmendy/Documents/Tek2/OOP/Arcade/lib/games/AGameModule.cpp` File Reference ↩

```
#include "AGameModule.hpp"
```

## 7.25 `/home/tmendy/Documents/Tek2/OOP/Arcade/lib/games/pacman/Pacman.cpp` File Reference ↩

```
#include "Pacman.hpp"
```

**Functions**

- void `init` ()  
*initialize the game*
- void `stop` ()  
*stop the game*
- `arcade::Pacman` \* `entryPoint` ()  
*generate entry point for the game library*

### 7.25.1 Function Documentation

#### 7.25.1.1 `entryPoint()`

```
arcade::Pacman * entryPoint ( )
```

generate entry point for the game library

#### 7.25.1.2 `init()`

```
void init ( )
```

initialize the game

#### 7.25.1.3 `stop()`

```
void stop ( )
```

stop the game



## 7.26 /home/tmendy/Documents/Tek2/OOP/Arcade/lib/games/snake/↵ Snake.cpp File Reference

```
#include "Snake.hpp"
```

### Functions

- `arcade::Snake * entryPoint ()`  
*generate entry point for the game library*

#### 7.26.1 Function Documentation

##### 7.26.1.1 entryPoint()

```
arcade::Snake * entryPoint ( )
```

generate entry point for the game library

## 7.27 /home/tmendy/Documents/Tek2/OOP/Arcade/lib/graphics/↵ ADisplayModule.cpp File Reference

```
#include "ADisplayModule.hpp"
```

## 7.28 /home/tmendy/Documents/Tek2/OOP/Arcade/lib/graphics/ncurses/↵ NCurses.cpp File Reference

```
#include "NCurses.hpp"
```

### Functions

- `arcade::NCurses * entryPoint ()`

#### 7.28.1 Function Documentation

##### 7.28.1.1 entryPoint()

```
arcade::NCurses * entryPoint ( )
```

## 7.29 /home/tmendy/Documents/Tek2/OOP/Arcade/lib/graphics/sdl2/↵ Sdl2.cpp File Reference

```
#include "Sdl2.hpp"
#include <iostream>
```

### Functions

- [arcade::Sdl2 \\* entryPoint \(\)](#)

#### 7.29.1 Function Documentation

##### 7.29.1.1 entryPoint()

```
arcade::Sdl2 * entryPoint ( )
```

## 7.30 /home/tmendy/Documents/Tek2/OOP/Arcade/lib/graphics/sfml/↵ Sfml.cpp File Reference

```
#include "Sfml.hpp"
```

### Functions

- [arcade::Sfml \\* entryPoint \(\)](#)

#### 7.30.1 Function Documentation

##### 7.30.1.1 entryPoint()

```
arcade::Sfml * entryPoint ( )
```

## 7.31 /home/tmendy/Documents/Tek2/OOP/Arcade/src/Main.cpp File Reference

```
#include "DLloader.hpp"
#include "IModule.hpp"
#include "Macros.hpp"
#include <ADisplayModule.hpp>
#include <cstring>
#include <iostream>
#include <libgen.h>
#include <unistd.h>
```

## Functions

- int `arcadeRe` (char \*path\_graphic\_lib)  
*launch the arcade*
- bool `is_good_graphic_lib` (char \*path\_graphic\_lib)  
*check if the library is a good graphic library*
- void `help` (void)  
*display the help*
- int `main` (int ac, char \*\*av)

### 7.31.1 Function Documentation

#### 7.31.1.1 `arcadeRe()`

```
int arcadeRe (  
    char * path_graphic_lib )
```

launch the arcade

##### Parameters

<code>path_graphic_lib</code>	path of the graphic library
-------------------------------	-----------------------------

##### Returns

int OK if the arcade is launched

#### 7.31.1.2 `help()`

```
void help (  
    void )
```

display the help

#### 7.31.1.3 `is_good_graphic_lib()`

```
bool is_good_graphic_lib (  
    char * path_graphic_lib )
```

check if the library is a good graphic library

##### Parameters

<code>path_graphic_lib</code>	path of the graphic library
-------------------------------	-----------------------------

**Returns**

true if the library is a good graphic library  
false if the library is not a good graphic library

**7.31.1.4 main()**

```
int main (  
    int ac,  
    char ** av )
```

# Index

/home/tmendy/Documents/Tek2/OOP/Arcade/include/CoreModule.hpp, 45  
\_gameData  
/home/tmendy/Documents/Tek2/OOP/Arcade/include/DLLoader.hpp, 46  
arcade::ADisplayModule, 15  
arcade::AGameModule, 19  
/home/tmendy/Documents/Tek2/OOP/Arcade/include/IModule.hpp, 53  
arcade::CoreModule, 24  
/home/tmendy/Documents/Tek2/OOP/Arcade/include/Macro.hpp, 54, 55  
arcade::AGameModule, 19  
/home/tmendy/Documents/Tek2/OOP/Arcade/include/game/AGameModule.hpp, 47  
arcade::ADisplayModule, 15  
/home/tmendy/Documents/Tek2/OOP/Arcade/include/game/Pacman.hpp, 48  
arcade::AGameModule, 19  
\_texture  
/home/tmendy/Documents/Tek2/OOP/Arcade/include/game/Snake.hpp, 48, 49  
arcade::ADisplayModule, 15  
arcade::Sfml, 40  
/home/tmendy/Documents/Tek2/OOP/Arcade/include/graphics/ADisplayModule.hpp, 49, 50  
arcade::ADisplayModule, 15  
/home/tmendy/Documents/Tek2/OOP/Arcade/include/graphics/ADisplayModule, 50, 51  
arcade::ADisplayModule, 13  
/home/tmendy/Documents/Tek2/OOP/Arcade/include/graphics/ASdl2Module, 51  
arcade::AGameModule, 17  
/home/tmendy/Documents/Tek2/OOP/Arcade/include/graphics/SfmlModule, 52  
arcade::CoreModule, 21  
/home/tmendy/Documents/Tek2/OOP/Arcade/lib/CoreModule/DLLoader, 55  
DLLoader< T >, 24  
/home/tmendy/Documents/Tek2/OOP/Arcade/lib/games/AGameModule.cpp, 56  
arcade::IModule, 28  
/home/tmendy/Documents/Tek2/OOP/Arcade/lib/games/pacman/NoPacman.cpp, 56  
arcade::NCurses, 31  
/home/tmendy/Documents/Tek2/OOP/Arcade/lib/games/snake/RealSnake.cpp, 57  
arcade::Pacman, 33  
/home/tmendy/Documents/Tek2/OOP/Arcade/lib/graphics/ADisplayModule.cpp, 57  
arcade::Sdl2, 36  
/home/tmendy/Documents/Tek2/OOP/Arcade/lib/graphics/noSfml/NCurses.cpp, 57  
arcade::Sfml, 39  
/home/tmendy/Documents/Tek2/OOP/Arcade/lib/graphics/sdl2/Sdl2.cpp, 58  
arcade::Snake, 42  
/home/tmendy/Documents/Tek2/OOP/Arcade/lib/graphics/sfml/Sfml.cpp, 58  
A  
/home/tmendy/Documents/Tek2/OOP/Arcade/src/Main.cpp, 58  
arcade::IModule, 27  
ADisplayModule  
arcade::ADisplayModule, 13  
AGameModule  
arcade::AGameModule, 17  
arcade, 9  
arcade::ADisplayModule, 11  
\_displayStatus, 15  
\_event, 15  
\_gameData, 15  
\_input, 15  
\_texture, 15

- [\\_window](#), 15
- [~ADisplayModule](#), 13
- [ADisplayModule](#), 13
- [display](#), 13
- [DisplayStatus](#), 12
- [GAMEOVER](#), 12
- [getDisplayStatus](#), 13
- [getInput](#), 13
- [getName](#), 13
- [getType](#), 13
- [init](#), 14
- [PAUSED](#), 12
- [RUNNING](#), 12
- [SELECTION](#), 12
- [sendGameData](#), 14
- [setDisplayStatus](#), 14
- [stop](#), 14
- [WIN](#), 12
- [arcade::AGameModule](#), 15
  - [\\_gameData](#), 19
  - [\\_gameStatus](#), 19
  - [\\_input](#), 19
  - [~AGameModule](#), 17
  - [AGameModule](#), 17
  - [GAMEOVER](#), 17
  - [GameStatus](#), 17
  - [getDisplayStatus](#), 17
  - [getName](#), 17
  - [getType](#), 18
  - [init](#), 18
  - [PAUSED](#), 17
  - [RUNNING](#), 17
  - [sendGameData](#), 18
  - [sendInput](#), 18
  - [setGameStatus](#), 19
  - [stop](#), 19
  - [WIN](#), 17
- [arcade::CoreModule](#), 19
  - [\\_coreStatus](#), 23
  - [\\_displayModule](#), 23
  - [\\_gameModule](#), 24
  - [~CoreModule](#), 21
  - [CoreModule](#), 21
  - [CoreStatus](#), 21
  - [getCoreStatus](#), 21
  - [getDisplayModule](#), 21
  - [getGameModule](#), 22
  - [getLib](#), 22
  - [getName](#), 22
  - [getType](#), 22
  - [init](#), 22
  - [RUNNING](#), 21
  - [SELECTION](#), 21
  - [setCoreStatus](#), 23
  - [setModule](#), 23
  - [stop](#), 23
- [arcade::IModule](#), 26
  - [~IModule](#), 28
- [A](#), 27
- [B](#), 27
- [C](#), 27
- [CENTIPEDE](#), 28
- [CORE](#), 28
- [D](#), 27
- [DOWN](#), 27
- [E](#), 27
- [ENTER](#), 27
- [ESCAPE](#), 27
- [F](#), 27
- [G](#), 27
- [GAME](#), 28
- [getName](#), 28
- [getType](#), 28
- [GRAPHIC](#), 28
- [H](#), 27
- [I](#), 27
- [IModule](#), 28
- [init](#), 29
- [J](#), 27
- [K](#), 27
- [KeyboardInput](#), 27
- [L](#), 27
- [LEFT](#), 27
- [LibName](#), 27
- [M](#), 27
- [ModuleType](#), 28
- [N](#), 27
- [NCURSES](#), 28
- [NIBBLER](#), 28
- [NONE](#), 27
- [O](#), 27
- [OPENGL](#), 28
- [P](#), 27
- [PACMAN](#), 28
- [Q](#), 27
- [QIX](#), 28
- [R](#), 27
- [RIGHT](#), 27
- [S](#), 27
- [SDL](#), 28
- [SFML](#), 28
- [SNAKE](#), 28
- [SOLARFOX](#), 28
- [SPACE](#), 27
- [stop](#), 29
- [T](#), 27
- [U](#), 27
- [UNKNOWN](#), 28
- [UP](#), 27
- [V](#), 27
- [W](#), 27
- [X](#), 27
- [Y](#), 27
- [Z](#), 27
- [arcade::IModule::GameData](#), 25
- [display\\_info](#), 25

- sprite\_value, 25
- arcade::NCurses, 29
  - ~NCurses, 31
  - display, 31
  - getName, 31
  - init, 31
  - NCurses, 31
  - stop, 31
- arcade::Pacman, 32
  - ~Pacman, 33
  - getName, 34
  - init, 34
  - Pacman, 33
  - stop, 34
- arcade::Sdl2, 34
  - ~Sdl2, 36
  - display, 36
  - getName, 36
  - init, 37
  - Sdl2, 36
  - stop, 37
- arcade::Sfml, 37
  - \_font, 40
  - \_texture, 40
  - ~Sfml, 39
  - display, 39
  - getInput, 39
  - getName, 40
  - init, 40
  - Sfml, 39
  - stop, 40
- arcade::Snake, 40
  - ~Snake, 42
  - getName, 42
  - init, 42
  - Snake, 42
  - stop, 43
- arcadeRe
  - Main.cpp, 59
- B
  - arcade::IModule, 27
- C
  - arcade::IModule, 27
- CENTIPEDE
  - arcade::IModule, 28
- CORE
  - arcade::IModule, 28
- CoreModule
  - arcade::CoreModule, 21
- CoreModule.cpp
  - getLib, 55
- CoreStatus
  - arcade::CoreModule, 21
- D
  - arcade::IModule, 27
- display
  - arcade::ADisplayModule, 13
  - arcade::NCurses, 31
  - arcade::Sdl2, 36
  - arcade::Sfml, 39
- display\_info
  - arcade::IModule::GameData, 25
- DisplayStatus
  - arcade::ADisplayModule, 12
- DLLoader
  - DLLoader< T >, 24
- DLLoader< T >, 24
  - ~DLLoader, 24
  - DLLoader, 24
  - getInstance, 25
  - handle, 25
- DOWN
  - arcade::IModule, 27
- E
  - arcade::IModule, 27
- ENTER
  - arcade::IModule, 27
- entryPoint
  - NCurses.cpp, 57
  - Pacman.cpp, 56
  - Sdl2.cpp, 58
  - Sfml.cpp, 58
  - Snake.cpp, 57
- ERROR
  - Macros.hpp, 54
- ESCAPE
  - arcade::IModule, 27
- F
  - arcade::IModule, 27
- G
  - arcade::IModule, 27
- GAME
  - arcade::IModule, 28
- GAMEOVER
  - arcade::ADisplayModule, 12
  - arcade::AGameModule, 17
- GameStatus
  - arcade::AGameModule, 17
- getCoreStatus
  - arcade::CoreModule, 21
- getDisplayModule
  - arcade::CoreModule, 21
- getDisplayStatus
  - arcade::ADisplayModule, 13
  - arcade::AGameModule, 17
- getGameModule
  - arcade::CoreModule, 22
- getInput
  - arcade::ADisplayModule, 13
  - arcade::Sfml, 39
- getInstance
  - DLLoader< T >, 25

getLib  
     arcade::CoreModule, 22  
     CoreModule.cpp, 55  
 getName  
     arcade::ADisplayModule, 13  
     arcade::AGameModule, 17  
     arcade::CoreModule, 22  
     arcade::IModule, 28  
     arcade::NCurses, 31  
     arcade::Pacman, 34  
     arcade::Sdl2, 36  
     arcade::Sfml, 40  
     arcade::Snake, 42  
 getType  
     arcade::ADisplayModule, 13  
     arcade::AGameModule, 18  
     arcade::CoreModule, 22  
     arcade::IModule, 28  
 GRAPHIC  
     arcade::IModule, 28  
  
 H  
     arcade::IModule, 27  
 handle  
     DLLoader< T >, 25  
 help  
     Main.cpp, 59  
  
 I  
     arcade::IModule, 27  
 IModule  
     arcade::IModule, 28  
 init  
     arcade::ADisplayModule, 14  
     arcade::AGameModule, 18  
     arcade::CoreModule, 22  
     arcade::IModule, 29  
     arcade::NCurses, 31  
     arcade::Pacman, 34  
     arcade::Sdl2, 37  
     arcade::Sfml, 40  
     arcade::Snake, 42  
     Pacman.cpp, 56  
 is\_good\_graphic\_lib  
     Main.cpp, 59  
  
 J  
     arcade::IModule, 27  
  
 K  
     arcade::IModule, 27  
 KeyboardInput  
     arcade::IModule, 27  
 KO  
     Macros.hpp, 54  
  
 L  
     arcade::IModule, 27  
 LEFT  
     arcade::IModule, 27  
  
 LibName  
     arcade::IModule, 27  
  
 M  
     arcade::IModule, 27  
 Macros.hpp  
     ERROR, 54  
     KO, 54  
     OK, 54  
 main  
     Main.cpp, 60  
 Main.cpp  
     arcadeRe, 59  
     help, 59  
     is\_good\_graphic\_lib, 59  
     main, 60  
 ModuleType  
     arcade::IModule, 28  
  
 N  
     arcade::IModule, 27  
 NCURSES  
     arcade::IModule, 28  
 NCurses  
     arcade::NCurses, 31  
 NCurses.cpp  
     entryPoint, 57  
 NIBBLER  
     arcade::IModule, 28  
 NONE  
     arcade::IModule, 27  
  
 O  
     arcade::IModule, 27  
 OK  
     Macros.hpp, 54  
 OPENGL  
     arcade::IModule, 28  
  
 P  
     arcade::IModule, 27  
 PACMAN  
     arcade::IModule, 28  
 Pacman  
     arcade::Pacman, 33  
 Pacman.cpp  
     entryPoint, 56  
     init, 56  
     stop, 56  
 PAUSED  
     arcade::ADisplayModule, 12  
     arcade::AGameModule, 17  
  
 Q  
     arcade::IModule, 27  
 QIX  
     arcade::IModule, 28  
  
 R



- arcade::IModule, [27](#)
- RIGHT
  - arcade::IModule, [27](#)
- RUNNING
  - arcade::ADisplayModule, [12](#)
  - arcade::AGameModule, [17](#)
  - arcade::CoreModule, [21](#)
- S
  - arcade::IModule, [27](#)
- SDL
  - arcade::IModule, [28](#)
- Sdl2
  - arcade::Sdl2, [36](#)
- Sdl2.cpp
  - entryPoint, [58](#)
- SELECTION
  - arcade::ADisplayModule, [12](#)
  - arcade::CoreModule, [21](#)
- sendGameData
  - arcade::ADisplayModule, [14](#)
  - arcade::AGameModule, [18](#)
- sendInput
  - arcade::AGameModule, [18](#)
- setCoreStatus
  - arcade::CoreModule, [23](#)
- setDisplayStatus
  - arcade::ADisplayModule, [14](#)
- setGameStatus
  - arcade::AGameModule, [19](#)
- setModule
  - arcade::CoreModule, [23](#)
- SFML
  - arcade::IModule, [28](#)
- Sfml
  - arcade::Sfml, [39](#)
- Sfml.cpp
  - entryPoint, [58](#)
- SNAKE
  - arcade::IModule, [28](#)
- Snake
  - arcade::Snake, [42](#)
- Snake.cpp
  - entryPoint, [57](#)
- SOLARFOX
  - arcade::IModule, [28](#)
- SPACE
  - arcade::IModule, [27](#)
- sprite\_value
  - arcade::IModule::GameData, [25](#)
- stop
  - arcade::ADisplayModule, [14](#)
  - arcade::AGameModule, [19](#)
  - arcade::CoreModule, [23](#)
  - arcade::IModule, [29](#)
  - arcade::NCurses, [31](#)
  - arcade::Pacman, [34](#)
  - arcade::Sdl2, [37](#)
  - arcade::Sfml, [40](#)
  - arcade::Snake, [43](#)
  - Pacman.cpp, [56](#)
- T
  - arcade::IModule, [27](#)
- U
  - arcade::IModule, [27](#)
- UNKNOWN
  - arcade::IModule, [28](#)
- UP
  - arcade::IModule, [27](#)
- V
  - arcade::IModule, [27](#)
- W
  - arcade::IModule, [27](#)
- WIN
  - arcade::ADisplayModule, [12](#)
  - arcade::AGameModule, [17](#)
- X
  - arcade::IModule, [27](#)
- Y
  - arcade::IModule, [27](#)
- Z
  - arcade::IModule, [27](#)