

## Model Railroad System

2.2.2

Generated by Doxygen 1.9.1



<b>1 Signal Abstract Types (Classes)</b>	<b>1</b>
1.1 Source files	1
1.2 Common methods and functionality	1
<b>2 Module Index</b>	<b>3</b>
2.1 Modules	3
<b>3 Namespace Index</b>	<b>5</b>
3.1 Namespace List	5
<b>4 Class Index</b>	<b>7</b>
4.1 Class List	7
<b>5 Module Documentation</b>	<b>9</b>
5.1 Using an Arduino Uno and a MAX72XX to Operate Signals	9
5.1.1 Detailed Description	9
5.1.2 Typedef Documentation	10
5.1.2.1 onetwoaspectlist	10
5.1.3 Enumeration Type Documentation	10
5.1.3.1 signalcolors	10
5.2 Using SR4s to Operate Signals	10
5.2.1 Detailed Description	11
5.2.2 Typedef Documentation	12
5.2.2.1 threeaspectlist	12
5.2.2.2 twoaspectlist	12
5.2.3 Enumeration Type Documentation	12
5.2.3.1 signalcolors	12
<b>6 Namespace Documentation</b>	<b>15</b>
6.1 arduiomax72xx_signals Namespace Reference	15
6.2 azatrax_signals Namespace Reference	15
<b>7 Class Documentation</b>	<b>17</b>
7.1 azatrax_signals::OneHead3Color Class Reference	17
7.1.1 Detailed Description	17
7.1.2 Constructor & Destructor Documentation	18
7.1.2.1 OneHead3Color()	18
7.1.3 Member Function Documentation	18
7.1.3.1 setaspect()	18
7.1.3.2 validate()	19
7.1.4 Member Data Documentation	19

---

7.1.4.1 signal	19
7.2 arduiomax72xx_signals::OneTwoHead3Color Class Reference	19
7.2.1 Detailed Description	20
7.2.2 Constructor & Destructor Documentation	20
7.2.2.1 OneTwoHead3Color()	20
7.2.3 Member Function Documentation	21
7.2.3.1 setaspect()	21
7.2.3.2 validate()	21
7.2.4 Member Data Documentation	21
7.2.4.1 aspectmap	21
7.2.4.2 driver	22
7.3 azatrax_signals::ThreeHead3over2over2 Class Reference	22
7.3.1 Detailed Description	22
7.3.2 Constructor & Destructor Documentation	23
7.3.2.1 ThreeHead3over2over2()	23
7.3.3 Member Function Documentation	23
7.3.3.1 setaspect()	23
7.3.3.2 validate()	24
7.3.4 Member Data Documentation	24
7.3.4.1 signallower	24
7.3.4.2 signalupper	24
7.4 azatrax_signals::TwoHead2over2 Class Reference	24
7.4.1 Detailed Description	25
7.4.2 Constructor & Destructor Documentation	25
7.4.2.1 TwoHead2over2()	25
7.4.3 Member Function Documentation	26
7.4.3.1 setaspect()	26
7.4.3.2 validate()	26
7.4.4 Member Data Documentation	26
7.4.4.1 signal	26
7.5 azatrax_signals::TwoHead3over2 Class Reference	27
7.5.1 Detailed Description	27
7.5.2 Constructor & Destructor Documentation	27
7.5.2.1 TwoHead3over2()	28
7.5.3 Member Function Documentation	29
7.5.3.1 setaspect()	29
7.5.3.2 validate()	29
7.5.4 Member Data Documentation	30
7.5.4.1 signallower	30

---

---

7.5.4.2 signalupper . . . . .	30
<b>Index</b>	<b>31</b>



# Chapter 1

## Signal Abstract Types (Classes)

This folder contains a collection of Tcl code to implement signals, using various methods.

### 1.1 Source files

Using Azatrax's SR4's to control signals is illustrated in the file [Azatrax\\_Signals.tcl](#). One or two SR4's can control one, two, or three headed signals, either common anode to common cathode.

Using an Arduinio with a MAX72XX LED Driver to control signals is illustrated in the file [ArduinioMAX72XX\\_Signals.tcl](#). Upto eight LEDs per signal is possible, although the code assumes a maximum of six LEDs in a three over three two headed signal.

Using Dr. Bruce Chubb's SMINI or SUSIC/USIC to control signals is illustrated in the file [Chubb\\_Signals.tcl](#). Output ports on these nodes can control one, two, or three headed signals.

Using CTI's Acela Network Bridge with CTI controler boards to control signals is illustrated in the file [CTI\\_Signals.tcl](#).

### 1.2 Common methods and functionality

All of the signal type constructors have a common structure. The constructors take the form:

```
typename objectname [optional options]
```

Eg:

```
azatrax_signals::OneHead3Color cp27w -signalname CP27W -signalsn 040001234
```

There is one common option, `-signalname`. This is the name of the signal object on the track work schematic. When the signal aspect is changed, the track work symbol is changed to display the signal's new aspect.

There is one common method, `setaspect`, which is used to set the signal's aspect. For a one headed signal, this method takes a single word (eg a single element list) that is the signal aspect. This will be one of the colors red, yellow, green, or dark. For a two headed signal, this method takes a list of two elements, each of which is one of the colors red, yellow, green, or dark. A three headed signal will take a a list of three elements, each of which is one of the colors red, yellow, green, or dark. It should be noted that not all possible combinations are allowed, only those aspects that make sense. This usually means that only one head will display a color other than red, with the other heads displaying red. That is {red yellow} or {green red} or {red red yellow} are allowed, but not {green yellow} or {green red yellow}.





## Chapter 2

# Module Index

### 2.1 Modules

Here is a list of all modules:

Using an Arduino Uno and a MAX72XX to Operate Signals . . . . .	9
Using SR4s to Operate Signals . . . . .	10



## Chapter 3

# Namespace Index

### 3.1 Namespace List

Here is a list of all namespaces with brief descriptions:

<a href="#">arduniomax72xx_signals</a> . . . . .	15
<a href="#">azatrax_signals</a> . . . . .	15



## Chapter 4

# Class Index

### 4.1 Class List

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

<a href="#">azatrax_signals::OneHead3Color</a>	
Single head signals, 3 color	17
<a href="#">arduniomax72xx_signals::OneTwoHead3Color</a>	
One or two heads signals, 3 colors per head	19
<a href="#">azatrax_signals::ThreeHead3over2over2</a>	
Three head signals, 3 over 2 over 2	22
<a href="#">azatrax_signals::TwoHead2over2</a>	
Two head signals, 2 over 2	24
<a href="#">azatrax_signals::TwoHead3over2</a>	
Two head signals, 3 over 2	27



## Chapter 5

# Module Documentation

### 5.1 Using an Arduino Uno and a MAX72XX to Operate Signals

Classes to operate signals using an Arduino Uno and a MAX72XX.

#### Classes

- class `ardunimax72xx_signals::OneTwoHead3Color`  
*One or two heads signals, 3 colors per head.*

#### Typedefs

- typedef listtype `ardunimax72xx_signals::onetwoaspectlist`

#### Enumerations

- enum `ardunimax72xx_signals::signalcolors` { `ardunimax72xx_signals::dark` , `ardunimax72xx_signals::red` , `ardunimax72xx_signals::yellow` , `ardunimax72xx_signals::green` }  
*Basic signal colors.*

#### 5.1.1 Detailed Description

Classes to operate signals using an Arduino Uno and a MAX72XX.

The module contains code to operate various sorts of signals using the hardware and code in the `ArdunioMAX72XX` folder. See the documentation there for information on how things are wired, etc.

## 5.1.2 Typedef Documentation

### 5.1.2.1 onetwoaspectlist

```
typedef listtype arduiomax72xx_signals::onetwoaspectlist
```

Definition at line 45 of file ArduioMAX72XX\_Signals.tcl.

## 5.1.3 Enumeration Type Documentation

### 5.1.3.1 signalcolors

```
enum arduiomax72xx_signals::signalcolors
```

Basic signal colors.

The four values are dark, red, yellow, and green.

#### Enumerator

dark	Dark, all lamps off, implies red.
red	Red, generally stop or stop and proceed.
yellow	Yellow, generally approach.
green	Green, generally clear.

Definition at line 20 of file ArduioMAX72XX\_Signals.tcl.

## 5.2 Using SR4s to Operate Signals

Classes to operate signals using SR4s.

### Classes

- class `azatrax_signals::OneHead3Color`  
*Single head signals, 3 color.*
- class `azatrax_signals::TwoHead3over2`



*Two head signals, 3 over 2.*

- class `azatrax_signals::TwoHead2over2`

*Two head signals, 2 over 2.*

- class `azatrax_signals::ThreeHead3over2over2`

*Three head signals, 3 over 2 over 2.*

## Typedefs

- typedef listtype `azatrax_signals::twoaspectlist`

*Aspects for two headed signals.*

- typedef listtype `azatrax_signals::threeaspectlist`

*Aspects for three headed signals.*

## Enumerations

- enum `azatrax_signals::signalcolors` { `azatrax_signals::dark` , `azatrax_signals::red` , `azatrax_signals::yellow` , `azatrax_signals::green` }

*Basic signal colors.*

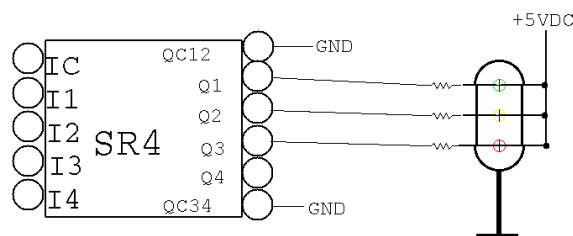
### 5.2.1 Detailed Description

Classes to operate signals using SR4s.

This file contains code to operate various sorts of signals using Azatrax SR4s.

Typical wiring for LED common anode signals:

Common Anode (+) LED Signals Connected to an SR4



**Figure 5.1** Connecting a three LED common anode signal to a SR4.

See the specific classes for how they expect the signals to be wired.

## 5.2.2 Typedef Documentation

### 5.2.2.1 threeaspectlist

```
typedef listtype azatrax_signals::threeaspectlist
```

Aspects for three headed signals.

This is a list of three aspect colors, the first element for the upper head and the second element for the middle head, and finally the third element for the bottom head.

Definition at line 219 of file Azatrax\_Signals.tcl.

### 5.2.2.2 twoaspectlist

```
azatrax_signals::twoaspectlist
```

Aspects for two headed signals.

This is a list of two aspect colors, the first element for the upper head and the second element for the lower head.

Definition at line 97 of file Azatrax\_Signals.tcl.

## 5.2.3 Enumeration Type Documentation

### 5.2.3.1 signalcolors

```
enum azatrax_signals::signalcolors
```

Basic signal colors.

The four values are dark, red, yellow, and green.

Enumerator

dark	Dark, all lamps off, implies red.
red	Red, generally stop or stop and proceed.
yellow	Yellow, generally approach.
green	Green, generally clear.

Definition at line 23 of file Azatrax\_Signals.tcl.



## Chapter 6

# Namespace Documentation

### 6.1 arduiniomax72xx\_signals Namespace Reference

#### Classes

- class [OneTwoHead3Color](#)  
*One or two heads signals, 3 colors per head.*

#### Typedefs

- typedef listtype [onetwoaspectlist](#)

#### Enumerations

- enum [signalcolors](#) { [dark](#) , [red](#) , [yellow](#) , [green](#) }  
*Basic signal colors.*

### 6.2 azatrax\_signals Namespace Reference

#### Classes

- class [OneHead3Color](#)  
*Single head signals, 3 color.*
- class [TwoHead3over2](#)  
*Two head signals, 3 over 2.*
- class [TwoHead2over2](#)  
*Two head signals, 2 over 2.*
- class [ThreeHead3over2over2](#)  
*Three head signals, 3 over 2 over 2.*

## Typedefs

- typedef listtype [twoaspectlist](#)  
*Aspects for two headed signals.*
- typedef listtype [threeaspectlist](#)  
*Aspects for three headed signals.*

## Enumerations

- enum [signalcolors](#) { [dark](#) , [red](#) , [yellow](#) , [green](#) }  
*Basic signal colors.*

## Chapter 7

# Class Documentation

### 7.1 azatrax\_signals::OneHead3Color Class Reference

Single head signals, 3 color.

#### Public Member Functions

- [OneHead3Color](#) (name,...)  
*Constructor: initialize the signal object.*
- [setaspect](#) (aspect)  
*Set signal aspect.*

#### Static Public Member Functions

- static [validate](#) (object)  
*Type validating code Raises an error if object is not either the empty string or a [OneHead3Color](#) type object.*

#### Private Attributes

- [signal](#)  
*Signal driver (SR4)*

#### 7.1.1 Detailed Description

Single head signals, 3 color.

Typically used for simple block signals. One SR4, with Q1 connected to the top lamp (green), Q2 connected to the middle lamp (yellow), and Q3 connected to the bottom lamp (red).

Typical usage:

```
azatrax_signals::OneHead3Color blocksignal1 -signaln 0400001234 -signalname Signal1
```

#### Author

Robert Heller <heller@deepsoft.com>

Definition at line 57 of file Azatrax\_Signals.tcl.

## 7.1.2 Constructor & Destructor Documentation

### 7.1.2.1 OneHead3Color()

```
azatrax_signals::OneHead3Color::OneHead3Color (
    name ,
    ... )
```

Constructor: initialize the signal object.

Create a low level actuator object and install it as a component.

#### Parameters

<i>name</i>	Name of the signal object.
...	Options: <ul style="list-style-type: none"> <li>• -signalsn Serial number of the SR4 that controls this signal.</li> <li>• -signalname Name of the signal on the track work schematic.</li> </ul>

## 7.1.3 Member Function Documentation

### 7.1.3.1 setaspect()

```
azatrax_signals::OneHead3Color::setaspect (
    aspect )
```

Set signal aspect.

#### Parameters

<i>aspect</i>	New aspect color.
---------------	-------------------



### 7.1.3.2 validate()

```
static azatrax_signals::OneHead3Color::validate (
    object ) [static]
```

Type validating code Raises an error if object is not either the empty string or a [OneHead3Color](#) type object.

#### Parameters

<i>object</i>	Some object.
---------------	--------------

## 7.1.4 Member Data Documentation

### 7.1.4.1 signal

```
azatrax_signals::OneHead3Color::signal [private]
```

Signal driver (SR4)

Definition at line 64 of file Azatrax\_Signals.tcl.

## 7.2 arduniomax72xx\_signals::OneTwoHead3Color Class Reference

One or two heads signals, 3 colors per head.

### Public Member Functions

- [OneTwoHead3Color](#) (name,...)  
*Constructor: initialize the signal object.*
- [setaspect](#) (aspect)  
*Set signal aspect.*

### Static Public Member Functions

- static [validate](#) (object)  
*Type validating code Raises an error if object is not either the empty string or a [OneTwoHead3Color](#) type object.*

## Private Attributes

- [driver](#)

*The SignalDriverMax72xx object.*

## Static Private Attributes

- static [aspectmap](#)

*Aspect map.*

### 7.2.1 Detailed Description

One or two heads signals, 3 colors per head.

Typical usage:

```
# Load the low-level code
package require SignalDriverMax72xx_Host
# Connect to the Ardunio
SignalDriverMax72xx controlpoint1 -portname /dev/ttyACM0
# Allocate a signal
arduniomax72xx_signals::OneTwoHead3Color CP1w2 -driver controlpoint1 -signal 0
# Set aspect to Green over Red (clear)
CP1w2 setaspect {green red}
```

#### Author

Robert Heller <heller@deepsoft.com>

Definition at line 64 of file ArdunioMAX72XX\_Signals.tcl.

### 7.2.2 Constructor & Destructor Documentation

#### 7.2.2.1 OneTwoHead3Color()

```
arduniomax72xx_signals::OneTwoHead3Color::OneTwoHead3Color (
    name ,
    ... )
```

Constructor: initialize the signal object.

Create a low level actuator object and install it as a component.

#### Parameters

<i>name</i>	Name of the signal object.
...	Options:
	<ul style="list-style-type: none"> <li>• -driver SignalDriverMax72xx object.</li> <li>• -signal Signal number on the MAX7200 board.</li> <li>• -signalname Name of the signal on the track work schematic.</li> </ul>

## 7.2.3 Member Function Documentation

### 7.2.3.1 setaspect()

```
arduinimax72xx_signals::OneTwoHead3Color::setaspect (
    aspect )
```

Set signal aspect.

#### Parameters

<i>aspect</i>	New aspect color.
---------------	-------------------

### 7.2.3.2 validate()

```
static arduinimax72xx_signals::OneTwoHead3Color::validate (
    object ) [static]
```

Type validating code Raises an error if object is not either the empty string or a [OneTwoHead3Color](#) type object.

#### Parameters

<i>object</i>	Some object.
---------------	--------------

## 7.2.4 Member Data Documentation

### 7.2.4.1 aspectmap

```
arduinimax72xx_signals::OneTwoHead3Color::aspectmap [static], [private]
```

Aspect map.

Definition at line 96 of file ArduinioMAX72XX\_Signals.tcl.

### 7.2.4.2 driver

```
arduniomax72xx_signals::OneTwoHead3Color::driver [private]
```

The SignalDriverMax72xx object.

Definition at line 72 of file ArdunioMAX72XX\_Signals.tcl.

## 7.3 azatrax\_signals::ThreeHead3over2over2 Class Reference

Three head signals, 3 over 2 over 2.

### Public Member Functions

- [ThreeHead3over2over2](#) (name,...)  
*Constructor: initialize the signal object.*
- [setaspect](#) (aspect)  
*Set signal aspect.*

### Static Public Member Functions

- static [validate](#) (object)  
*Type validating code Raises an error if object is not either the empty string or a TwoHead3over2over2 type object.*

### Private Attributes

- [signalupper](#)  
*Signal driver (SR4)*
- [signallower](#)  
*Signal driver (SR4)*

### 7.3.1 Detailed Description

Three head signals, 3 over 2 over 2.

Typically used for simple interlocking signals. Two SR4s, with one driving the top head: with Q1 connected to the top lamp (green), Q2 connected to the middle lamp (yellow), and Q3 connected to the bottom lamp (red). The second SR4 wired to the middle and lower heads, its Q1 connected to the top lamp (green or yellow) of the middle head, and Q2 to the bottom lamp (red) of the middle head. Then Q3 is connected to the top lamp (green or yellow) of the bottom head, and Q4 connected to the bottom lamp (red) of the bottom head.

Typical usage:

```
azatrax_signals::TwoHead3over2over2 interlocksignal1 \  
-signalupper 0400001234 \  
-signalnlower 0400001235 \  
-signalname Signal1
```

### Author

Robert Heller <heller@deepsoft.com>

Definition at line 243 of file Azatrax\_Signals.tcl.

## 7.3.2 Constructor & Destructor Documentation

### 7.3.2.1 ThreeHead3over2over2()

```
azatrax_signals::ThreeHead3over2over2::ThreeHead3over2over2 (
    name ,
    ... )
```

Constructor: initialize the signal object.

Create a low level actuator object and install it as a component.

#### Parameters

<i>name</i>	Name of the signal object.
...	Options: <ul style="list-style-type: none"><li>• -signalsnupper Serial number of the SR4 that controls the upper head of this signal.</li><li>• -signalsnlower Serial number of the SR4 that controls the lower two heads of this signal.</li><li>• -signalname Name of the signal on the track work schematic.</li></ul>

## 7.3.3 Member Function Documentation

### 7.3.3.1 setaspect()

```
azatrax_signals::ThreeHead3over2over2::setaspect (
    aspect )
```

Set signal aspect.

#### Parameters

<i>aspect</i>	New aspect color.
---------------	-------------------

### 7.3.3.2 validate()

```
static azatrax_signals::ThreeHead3over2over2::validate (
    object ) [static]
```

Type validating code Raises an error if object is not either the empty string or a TwoHead3over2over2 type object.

#### Parameters

<i>object</i>	Some object.
---------------	--------------

## 7.3.4 Member Data Documentation

### 7.3.4.1 signallower

```
azatrax_signals::ThreeHead3over2over2::signallower [private]
```

Signal driver (SR4)

Definition at line 255 of file Azatrax\_Signals.tcl.

### 7.3.4.2 signalupper

```
azatrax_signals::ThreeHead3over2over2::signalupper [private]
```

Signal driver (SR4)

Definition at line 251 of file Azatrax\_Signals.tcl.

## 7.4 azatrax\_signals::TwoHead2over2 Class Reference

Two head signals, 2 over 2.

### Public Member Functions

- [TwoHead2over2](#) (name,...)  
*Constructor: initialize the signal object.*
- [setaspect](#) (aspect)  
*Set signal aspect.*

## Static Public Member Functions

- static [validate](#) (object)

Type validating code Raises an error if object is not either the empty string or a [TwoHead2over2](#) type object.

## Private Attributes

- [signal](#)

Signal driver (SR4)

### 7.4.1 Detailed Description

Two head signals, 2 over 2.

Typically used for simple interlocking signals. One SR4, driving both heads: with Q1 connected to the top lamp (green) or the top head, Q2 connected to the bottom lamp (red) of the top head. Then Q3 connected to the top lamp (green or yellow) of othe lower head, and Q4 to the bottom lamp (red) of the lower head.

Typical usage:

```
azatrax_signals::TwoHead2over2 interlocksignal1 -signalsn 0400001234 \
-signalname Signal1
```

#### Author

Robert Heller <heller@deepsoft.com>

Definition at line 178 of file Azatrax\_Signals.tcl.

### 7.4.2 Constructor & Destructor Documentation

#### 7.4.2.1 TwoHead2over2()

```
azatrax_signals::TwoHead2over2::TwoHead2over2 (
    name ,
    ... )
```

Constructor: initialize the signal object.

Create a low level actuator object and install it as a component.

#### Parameters

<i>name</i>	Name of the signal object.
...	Options:
Generated by Doxygen	signals Serial number of the SR4 • -signalname Name of the signal on the track work schematic.

## 7.4.3 Member Function Documentation

### 7.4.3.1 setaspect()

```
azatrax_signals::TwoHead2over2::setaspect (
    aspect )
```

Set signal aspect.

#### Parameters

<i>aspect</i>	New aspect color.
---------------	-------------------

### 7.4.3.2 validate()

```
static azatrax_signals::TwoHead2over2::validate (
    object ) [static]
```

Type validating code Raises an error if object is not either the empty string or a [TwoHead2over2](#) type object.

#### Parameters

<i>object</i>	Some object.
---------------	--------------

## 7.4.4 Member Data Documentation

### 7.4.4.1 signal

```
azatrax_signals::TwoHead2over2::signal [private]
```

Signal driver (SR4)

Definition at line 185 of file Azatrax\_Signals.tcl.



## 7.5 azatrax\_signals::TwoHead3over2 Class Reference

Two head signals, 3 over 2.

### Public Member Functions

- [TwoHead3over2](#) (name,...)  
*Constructor: initialize the signal object.*
- [setaspect](#) (aspect)  
*Set signal aspect.*

### Static Public Member Functions

- static [validate](#) (object)  
*Type validating code Raises an error if object is not either the empty string or a [TwoHead3over2](#) type object.*

### Private Attributes

- [signalupper](#)  
*Signal driver (SR4)*
- [signallower](#)  
*Signal driver (SR4)*

### 7.5.1 Detailed Description

Two head signals, 3 over 2.

Typically used for simple interlocking signals. Two SR4s, with one driving the top head: with Q1 connected to the top lamp (green), Q2 connected to the middle lamp (yellow), and Q3 connected to the bottom lamp (red). The second SR4 wired to the lower head, its Q1 connected to the top lamp (green or yellow), and Q2 to the bottom lamp (red).

Typical usage:

```
azatrax_signals::TwoHead3over2 interlocksignal1 \
-signalupper 0400001234 \
-signalslower 0400001235 \
-signalname Signal1
```

#### Author

Robert Heller <heller@deepsoft.com>

Definition at line 118 of file Azatrax\_Signals.tcl.

### 7.5.2 Constructor & Destructor Documentation

### 7.5.2.1 TwoHead3over2()

```
azatrax_signals::TwoHead3over2::TwoHead3over2 (
    name ,
    ... )
```

Constructor: initialize the signal object.

Create a low level actuator object and install it as a component.

## Parameters

<i>name</i>	Name of the signal object.
...	Options: <ul style="list-style-type: none"><li>• -signalsnupper Serial number of the SR4 that controls the upper head of this signal.</li><li>• -signalsnlower Serial number of the SR4 that controls the lower head of this signal.</li><li>• -signalname Name of the signal on the track work schematic.</li></ul>

### 7.5.3 Member Function Documentation

#### 7.5.3.1 setaspect()

```
azatrax_signals::TwoHead3over2::setaspect (
    aspect )
```

Set signal aspect.

## Parameters

<i>aspect</i>	New aspect color.
---------------	-------------------

#### 7.5.3.2 validate()

```
static azatrax_signals::TwoHead3over2::validate (
    object ) [static]
```

Type validating code Raises an error if object is not either the empty string or a [TwoHead3over2](#) type object.

## Parameters

<i>object</i>	Some object.
---------------	--------------

## 7.5.4 Member Data Documentation

### 7.5.4.1 `signallower`

`azatrax_signals::TwoHead3over2::signallower` [private]

Signal driver (SR4)

Definition at line 130 of file `Azatrax_Signals.tcl`.

### 7.5.4.2 `signalupper`

`azatrax_signals::TwoHead3over2::signalupper` [private]

Signal driver (SR4)

Definition at line 126 of file `Azatrax_Signals.tcl`.

# Index

- ardunimax72xx\_signals, [15](#)
- ardunimax72xx\_signals::OneTwoHead3Color, [19](#)
  - aspectmap, [21](#)
  - driver, [21](#)
  - OneTwoHead3Color, [20](#)
  - setaspect, [21](#)
  - validate, [21](#)
- aspectmap
  - ardunimax72xx\_signals::OneTwoHead3Color, [21](#)
- azatrax\_signals, [15](#)
- azatrax\_signals::OneHead3Color, [17](#)
  - OneHead3Color, [18](#)
  - setaspect, [18](#)
  - signal, [19](#)
  - validate, [18](#)
- azatrax\_signals::ThreeHead3over2over2, [22](#)
  - setaspect, [23](#)
  - signallower, [24](#)
  - signalupper, [24](#)
  - ThreeHead3over2over2, [23](#)
  - validate, [23](#)
- azatrax\_signals::TwoHead2over2, [24](#)
  - setaspect, [26](#)
  - signal, [26](#)
  - TwoHead2over2, [25](#)
  - validate, [26](#)
- azatrax\_signals::TwoHead3over2, [27](#)
  - setaspect, [29](#)
  - signallower, [30](#)
  - signalupper, [30](#)
  - TwoHead3over2, [27](#)
  - validate, [29](#)
- dark
  - Using an Arduinio Uno and a MAX72XX to Operate Signals, [10](#)
  - Using SR4s to Operate Signals, [12](#)
- driver
  - ardunimax72xx\_signals::OneTwoHead3Color, [21](#)
- green
  - Using an Arduinio Uno and a MAX72XX to Operate Signals, [10](#)
  - Using SR4s to Operate Signals, [12](#)
- OneHead3Color
  - azatrax\_signals::OneHead3Color, [18](#)
- onetwoaspectlist
  - Using an Arduinio Uno and a MAX72XX to Operate Signals, [10](#)
- OneTwoHead3Color
  - ardunimax72xx\_signals::OneTwoHead3Color, [20](#)
- red
  - Using an Arduinio Uno and a MAX72XX to Operate Signals, [10](#)
  - Using SR4s to Operate Signals, [12](#)
- setaspect
  - ardunimax72xx\_signals::OneTwoHead3Color, [21](#)
  - azatrax\_signals::OneHead3Color, [18](#)
  - azatrax\_signals::ThreeHead3over2over2, [23](#)
  - azatrax\_signals::TwoHead2over2, [26](#)
  - azatrax\_signals::TwoHead3over2, [29](#)
- signal
  - azatrax\_signals::OneHead3Color, [19](#)
  - azatrax\_signals::TwoHead2over2, [26](#)
- signalcolors
  - Using an Arduinio Uno and a MAX72XX to Operate Signals, [10](#)
  - Using SR4s to Operate Signals, [12](#)
- signallower
  - azatrax\_signals::ThreeHead3over2over2, [24](#)
  - azatrax\_signals::TwoHead3over2, [30](#)
- signalupper
  - azatrax\_signals::ThreeHead3over2over2, [24](#)
  - azatrax\_signals::TwoHead3over2, [30](#)
- threeaspectlist
  - Using SR4s to Operate Signals, [12](#)
- ThreeHead3over2over2
  - azatrax\_signals::ThreeHead3over2over2, [23](#)
- twoaspectlist
  - Using SR4s to Operate Signals, [12](#)
- TwoHead2over2
  - azatrax\_signals::TwoHead2over2, [25](#)
- TwoHead3over2
  - azatrax\_signals::TwoHead3over2, [27](#)
- Using an Arduinio Uno and a MAX72XX to Operate Signals, [9](#)
  - dark, [10](#)

- green, [10](#)
- onetwoaspectlist, [10](#)
- red, [10](#)
- signalcolors, [10](#)
- yellow, [10](#)

#### Using SR4s to Operate Signals, [10](#)

- dark, [12](#)
- green, [12](#)
- red, [12](#)
- signalcolors, [12](#)
- threeaspectlist, [12](#)
- twoaspectlist, [12](#)
- yellow, [12](#)

#### validate

- arduniomax72xx\_signals::OneTwoHead3Color, [21](#)
- azatrax\_signals::OneHead3Color, [18](#)
- azatrax\_signals::ThreeHead3over2over2, [23](#)
- azatrax\_signals::TwoHead2over2, [26](#)
- azatrax\_signals::TwoHead3over2, [29](#)

#### yellow

- Using an Arduino Uno and a MAX72XX to Operate Signals, [10](#)
- Using SR4s to Operate Signals, [12](#)