

```

/**
 * @file BT.jsx
 * @author Joey Damico
 * @date September 25, 2019
 * @summary React JSX Component Class that is for BT Interlocking
 *
 * @description Extends the React Component Class and is the UI part
of the BT Interlocking,
 * this class controls all the drawings of routes, and also gives a
visual representation
 * of that status of the interlocking
 */

// Import React Component
import React, { Component } from 'react';
// Import CSS style sheet
import '../css/Bergen_County_Line/bt.css';

// Import Images
// Switch Images
import CX_135 from '../public/images/CX_135.png';
import CX_135_Lined_Top from '../public/images/
CX_135_Lined_Top.png';
import CX_135_Lined_Bottom from '../public/images/
CX_135_Lined_Bottom.png';
import CX_135_Lined_Both from '../public/images/
CX_135_Lined_Both.png';
import CX_135_R from '../public/images/CX_135_R.png';
import CX_135_R_Lined from '../public/images/
CX_135_R_Lined.png';
import CX_135_Lined_Top_Occupied_Bottom from '../public/
images/CX_135_Lined_Top_Occupied_Bottom.png';
import CX_135_Occupied_Top_Lined_Bottom from '../public/
images/CX_135_Occupied_Top_Lined_Bottom.png';
import CX_135_Occupied_Top from '../public/images/
CX_135_Occupied_Top.png';
import CX_135_Occupied_Bottom from '../public/images/
CX_135_Occupied_Bottom.png';
import CX_135_Occupied_Both from '../public/images/
CX_135_Occupied_Both.png';
import CX_135_R_Occupied from '../public/images/
CX_135_R_Occupied.png';

import CX_225 from '../public/images/CX_225.png';
import CX_225_Lined_Top from '../public/images/
CX_225_Lined_Top.png';
import CX_225_Lined_Bottom from '../public/images/
CX_225_Lined_Bottom.png';
import CX_225_Lined_Both from '../public/images/
CX_225_Lined_Both.png';

```

```

import CX_225_R from '../../../../../public/images/CX_225_R.png';
import CX_225_R_Lined from '../../../../../public/images/
CX_225_R_Lined.png';
import CX_225_Lined_Top_Occupied_Bottom from '../../../../../public/
images/CX_225_Lined_Top_Occupied_Bottom.png';
import CX_225_Occupied_Top_Lined_Bottom from '../../../../../public/
images/CX_225_Occupied_Top_Lined_Bottom.png';
import CX_225_Occupied_Top from '../../../../../public/images/
CX_225_Occupied_Top.png';
import CX_225_Occupied_Bottom from '../../../../../public/images/
CX_225_Occupied_Bottom.png';
import CX_225_Occupied_Both from '../../../../../public/images/
CX_225_Occupied_Both.png';
import CX_225_R_Occupied from '../../../../../public/images/
CX_225_R_Occupied.png';

import SW_U_E from '../../../../../public/images/SW_U_E.png';
import SW_U_E_Lined from '../../../../../public/images/SW_U_E_Lined.png';
import SW_U_E_Occupied from '../../../../../public/images/
SW_U_E_Occupied.png';
import SW_U_E_R from '../../../../../public/images/SW_U_E_R.png';
import SW_U_E_R_Lined from '../../../../../public/images/
SW_U_E_R_Lined.png';
import SW_U_E_R_Occupied from '../../../../../public/images/
SW_U_E_R_Occupied.png';

// Signal Images
import SIG_W from '../../../../../public/images/SIG_W.png';
import SIG_W_Clear from '../../../../../public/images/SIG_W_Clear.png';
import SIG_W_Stop from '../../../../../public/images/SIG_W_Stop.png';
import SIG_E from '../../../../../public/images/SIG_E.png';
import SIG_E_Clear from '../../../../../public/images/SIG_E_Clear.png';
import SIG_E_Stop from '../../../../../public/images/SIG_E_Stop.png';

// Color Constants For Drawing Routes
const Empty = '#999999';
const Green = '#75fa4c';
const Red = '#eb3323';

/**
 * The React JSX Component Class for the BT Interlocking
 *
 * This class is a JSX React Component for the BT Interlocking, this
will control all the UI for the comonent,
 * and the click events that will pass reference between the backend
and the user. This also controls drawing the
 * route drawings to show if a route(s) is setup in the interlocking
or if the route is occupied
 */

```

```

class BT extends Component {
  /**
   * @summary Object that holds the state or status information for
the component
   *
   * @description This object holds all the information for the
interlocking that is required to display the routes
   * correctly Anything that has "this.props." is passed down from
the CTC interlocking class
   */
  state = {
    // Switch Status
    sw_1: this.props.status.sw_1,
    sw_3: this.props.status.sw_3,
    sw_5: this.props.status.sw_5,
    // Image File for the switch - Will change depending on route
    sw_1_src: CX_135,
    sw_3_src: CX_225,
    sw_5_src: SW_U_E,
    // Colors for tail tracks - Will change depending on route
    tail_1_w: Empty,
    tail_2_w: Empty,
    tail_1_e: Empty,
    tail_2_e: Empty,
    tail_3_e: Empty,
    // Image File for the signals - Will change depending on route
    sig_2w1_src: SIG_W,
    sig_2w2_src: SIG_W,
    sig_4w_src: SIG_W,
    sig_2e_src: SIG_E,
    sig_4e_src: SIG_E,
    // Information For Interlocking Routes
    occupied_1: this.props.status.occupied_trk_1,
    occupied_2: this.props.status.occupied_trk_2,
    route_1: this.props.status.routed_1,
    route_2: this.props.status.routed_2,
    routes: this.props.status.routes
  };

  /**
   * componentWillReceiveProps()
   * @summary Function that updates the state of the component The
data that is being changed is passed down from the CTC classes in the
simulation backend
   *
   * @param nextProps, the new data to set the component state too
   */
  componentWillReceiveProps(nextProps){
    this.setState({
      sw_1: nextProps.status.sw_1,

```

```

        sw_3: nextProps.status.sw_3,
        sw_5: nextProps.status.sw_5,

        occupied_1: nextProps.status.occupied_trk_1,
        occupied_2: nextProps.status.occupied_trk_2,
        route_1: nextProps.status.routed_1,
        route_2: nextProps.status.routed_2,
        routes: nextProps.status.routes
    });
}
// ---- END componentWillReceiveProps() ----

/**
 * render()
 * @summary standard React function that draws the interlocking to
the screen
 */
render() {
    // Clear all the drawings from the interlocking so if a train
clears the route is gone
    this.reset_drawings();
    // Set the switch images based off the state of each crossover
    this.set_switch_images();
    // Draw all the current routes in the interlocking
    this.set_route_drawings();

    // Returns the HTML to draw the interlocking and it's current
state to the screen
    return (
        <div>
            {/* Tags */}
            <div className="bt_title">BT</div>
            <div className="bt_milepost">MP 14.2</div>
            {/* West Side Tail Tracks */}
            <div className="bt_1_west" style={{background:
this.state.tail_1_w}}></div>
            <div className="bt_2_west" style={{background:
this.state.tail_2_w}}></div>
            {/* Switches */}
            <div className="bt_SW_1"
onClick={this.props.throw_sw_1}><img src={this.state.sw_1_src}/></div>
            <div className="bt_SW_3"
onClick={this.props.throw_sw_3}><img src={this.state.sw_3_src}/></div>
            <div className="bt_SW_5"
onClick={this.props.throw_sw_5}><img src={this.state.sw_5_src}/></div>
            {/* East Side Tail Tracks */}
            <div className="bt_1_east" style={{background:
this.state.tail_1_e}}></div>
            <div className="bt_2_east" style={{background:
this.state.tail_2_e}}></div>

```

```

        <div className="bt_3_east" style={{background:
this.state.tail_3_e}}></div>
        { /* Signals */ }
        <div className="bt_sig_2w-2"
onClick={this.props.click_sig_2w2}><img src={this.state.sig_2w2_src}/
></div>
        <div className="bt_sig_2w-1"
onClick={this.props.click_sig_2w1}><img src={this.state.sig_2w1_src}/
></div>
        <div className="bt_sig_4w"
onClick={this.props.click_sig_4w}><img src={this.state.sig_4w_src}/></
div>
        <div className="bt_sig_2e"
onClick={this.props.click_sig_2e}><img src={this.state.sig_2e_src}/></
div>
        <div className="bt_sig_4e"
onClick={this.props.click_sig_4e}><img src={this.state.sig_4e_src}/></
div>
    </div>
  );
}
// ---- END render() ----

/**
 * @summary Sets the drawing for the route through the
interlocking
 *
 * @description Function takes what routes are currently set in
the Interlocking class and displays that route in the UI, the drawing
 * will change depending on if the interlocking is occupied or not
 */
set_route_drawings() {
  let color_1 = Empty;
  let color_2 = Empty;

  // Setting the color of the tracks depending on if the
interlocking in occupied or not
  if (this.state.route_1) {
    color_1 = Green;
  }
  if (this.state.route_2) {
    color_2 = Green;
  }
  if (this.state.occupied_1) {
    color_1 = Red;
  }
  if (this.state.occupied_2) {
    color_2 = Red;
  }
}

```

```

    // Loop through all the routes
    for (let i = 0; i < this.state.routes.length; i++) {
        if (this.state.routes[i] === "W_1_1__3_ridgewood_bt" ||
this.state.routes[i] === "E_1_1__1_bt_pascack") {
            // Tail Tracks
            this.state.tail_1_e = color_1;
            this.state.tail_1_w = color_1;

            // The route is occupied
            if (this.state.occupied_1) {
                // Switches
                this.state.sw_5_src = SW_U_E_Occupied;

                // Crossovers that could change based off Track #2
                // Track #2 Routed
                if (this.state.route_2) {
                    this.state.sw_1_src =
CX_135_Occupied_Top_Lined_Bottom;
                    this.state.sw_3_src =
CX_225_Occupied_Top_Lined_Bottom;
                }
                // Track #2 Occupied
                else if (this.state.occupied_2) {
                    this.state.sw_1_src = CX_135_Occupied_Both;
                    this.state.sw_3_src = CX_225_Occupied_Both;
                }
                // Nothing Track #2
                else {
                    this.state.sw_1_src = CX_135_Occupied_Top;
                    this.state.sw_3_src = CX_225_Occupied_Top;
                }

                // Signals
                this.state.sig_2w1_src = SIG_W_Stop;
                this.state.sig_2w2_src = SIG_W_Stop;
                this.state.sig_2e_src = SIG_E_Stop;
            }
            // The route is NOT occupied
            else {
                // Switches
                this.state.sw_5_src = SW_U_E_Lined;

                // Crossovers that could change based off of Track
#2
                // Track #2 Routed
                if (this.state.route_2) {
                    this.state.sw_1_src = CX_135_Lined_Both;
                    this.state.sw_3_src = CX_225_Lined_Both;
                }
                // Track #2 Occupied

```

```

        else if (this.state.occupied_2) {
            this.state.sw_1_src =
CX_135_Lined_Top_Occupied_Bottom;
            this.state.sw_3_src =
CX_225_Lined_Top_Occupied_Bottom;
        }
        // Nothing Track #2
        else {
            this.state.sw_1_src = CX_135_Lined_Top;
            this.state.sw_3_src = CX_225_Lined_Top;
        }

        // Signals
        // West Bound Signals
        if (this.state.routes[i] === "W_1_1__|
__3_ridgewood_bt") {
            this.state.sig_2w1_src = SIG_W_Clear;
            this.state.sig_2w2_src = SIG_W_Stop;
            this.state.sig_2e_src = SIG_E_Stop;
        }
        // East Bound Signals
        else {
            this.state.sig_2w1_src = SIG_W_Stop;
            this.state.sig_2w2_src = SIG_W_Stop;
            this.state.sig_2e_src = SIG_E_Clear;
        }
    }
    }
    else if (this.state.routes[i] === "W_3_1__|
__3_ridgewood_bt" || this.state.routes[i] === "E_1_3__|__3_bt_nysw") {
        // Tail Tracks
        this.state.tail_3_e = color_1;
        this.state.tail_1_w = color_1;

        // If The Route Is Occupied
        if (this.state.occupied_1) {
            // Switches
            this.state.sw_5_src = SW_U_E_R_Occupied;

            // Crossovers that can change depending on the
other track
            if (this.state.route_2) {
                this.state.sw_1_src =
CX_135_Occupied_Top_Lined_Bottom;
                this.state.sw_3_src =
CX_225_Occupied_Top_Lined_Bottom;
            }
            else if (this.state.occupied_2) {
                this.state.sw_1_src = CX_135_Occupied_Both;
                this.state.sw_3_src = CX_225_Occupied_Both;
            }
        }
    }
}

```

```

    }
    else {
        this.state.sw_1_src = CX_135_Occupied_Top;
        this.state.sw_3_src = CX_225_Occupied_Top;
    }

    // Signals
    this.state.sig_2w1_src = SIG_W_Stop;
    this.state.sig_2w2_src = SIG_W_Stop;
    this.state.sig_2e_src = SIG_E_Stop;
}
// The Route Is Not Occupied
else {
    // Switches
    this.state.sw_5_src = SW_U_E_R_Lined;

    // Crossovers that can change depending on the
other track
    if (this.state.route_2) {
        this.state.sw_1_src = CX_135_Lined_Both;
        this.state.sw_3_src = CX_225_Lined_Both;
    }
    else if (this.state.occupied_2) {
        this.state.sw_1_src =
CX_135_Lined_Top_Occupied_Bottom;
        this.state.sw_3_src =
CX_225_Lined_Top_Occupied_Bottom;
    }
    else {
        this.state.sw_1_src = CX_135_Lined_Top;
        this.state.sw_3_src = CX_225_Lined_Top;
    }

    // Signals
    // West Bound Signals
    if (this.state.routes[i] === "W_3_1__|
__3_ridgewood_bt") {
        this.state.sig_2w1_src = SIG_W_Stop;
        this.state.sig_2w2_src = SIG_W_Clear;
        this.state.sig_2e_src = SIG_E_Stop;
    }
    // East Bound Signals
    else {
        this.state.sig_2w1_src = SIG_W_Stop;
        this.state.sig_2w2_src = SIG_W_Stop;
        this.state.sig_2e_src = SIG_E_Clear;
    }
}
}
else if (this.state.routes[i] === "W_2_2__|

```



```

__4_ridgewood_bt" || this.state.routes[i] === "E_2_2__|
__2_bt_pascack") {
    // Tail Tracks
    this.state.tail_2_e = color_2;
    this.state.tail_2_w = color_2;

    // If the Route Is Occupied
    if (this.state.occupied_2) {
        // Switches
        // Crossovers that can change depending on the
other track
        if (this.state.route_1) {
            this.state.sw_1_src =
CX_135_Lined_Top_Occupied_Bottom;
            this.state.sw_3_src =
CX_225_Lined_Top_Occupied_Bottom;
        }
        else if (this.state.occupied_1) {
            this.state.sw_1_src = CX_135_Occupied_Both;
            this.state.sw_3_src = CX_225_Occupied_Both;
        }
        else {
            this.state.sw_1_src = CX_135_Occupied_Bottom;
            this.state.sw_3_src = CX_225_Occupied_Bottom;
        }

        // Signals
        this.state.sig_4w_src = SIG_W_Stop;
        this.state.sig_4e_src = SIG_E_Stop;
    }
    // The Route In Not Occupied
    else {
        // Switches
        // Crossovers that can change depending on the
other track
        if (this.state.route_1) {
            this.state.sw_1_src = CX_135_Lined_Both;
            this.state.sw_3_src = CX_225_Lined_Both;
        }
        else if (this.state.occupied_1) {
            this.state.sw_1_src =
CX_135_Occupied_Top_Lined_Bottom;
            this.state.sw_3_src =
CX_225_Occupied_Top_Lined_Bottom;
        }
        else {
            this.state.sw_1_src = CX_135_Lined_Bottom;
            this.state.sw_3_src = CX_225_Lined_Bottom;
        }
    }
}

```

```

        // Signals
        // West Bound Signals
        if (this.state.routes[i] === "W_2_2__|
__4_ridgewood_bt") {
            this.state.sig_4w_src = SIG_W_Clear;
            this.state.sig_4e_src = SIG_E_Stop;
        }
        // East Bound Signals
        else {
            this.state.sig_4w_src = SIG_W_Stop;
            this.state.sig_4e_src = SIG_E_Clear;
        }
    }
    }
    else if (this.state.routes[i] === "W_1_2__|
__4_ridgewood_bt") {
        // Tail Tracks
        this.state.tail_1_e = color_1;
        this.state.tail_2_w = color_1;

        // The Route Is Occupied
        if (this.state.occupied_1) {
            // Switches
            this.state.sw_5_src = SW_U_E_Occupied;
            this.state.sw_3_src = CX_225_R_Occupied;
            this.state.sw_1_src = CX_135_Occupied_Bottom;

            // Signals
            this.state.sig_2w1_src = SIG_W_Stop;
            this.state.sig_2w2_src = SIG_W_Stop;
            this.state.sig_4w_src = SIG_W_Stop;
            this.state.sig_2e_src = SIG_E_Stop;
            this.state.sig_4e_src = SIG_E_Stop;
        }
        // The Route Is NOT Occupied
        else {
            // Switches
            this.state.sw_5_src = SW_U_E_Lined;
            this.state.sw_3_src = CX_225_R_Lined;
            this.state.sw_1_src = CX_135_Lined_Bottom;

            // Signals
            this.state.sig_2w1_src = SIG_W_Clear;
            this.state.sig_2w2_src = SIG_W_Stop;
            this.state.sig_4w_src = SIG_W_Stop;
            this.state.sig_2e_src = SIG_E_Stop;
            this.state.sig_4e_src = SIG_E_Stop;
        }
    }
    }
    else if (this.state.routes[i] === "E_2_1__|

```

```

__1_bt_pascack") {
    // Tail Tracks
    this.state.tail_1_e = color_2;
    this.state.tail_2_w = color_2;

    // The Route Is Occupied
    if (this.state.occupied_2) {
        // Switches
        this.state.sw_5_src = SW_U_E_Occupied;
        this.state.sw_3_src = CX_225_R_Occupied;
        this.state.sw_1_src = CX_135_Occupied_Bottom;

        // Signals
        this.state.sig_2w1_src = SIG_W_Stop;
        this.state.sig_2w2_src = SIG_W_Stop;
        this.state.sig_4w_src = SIG_W_Stop;
        this.state.sig_2e_src = SIG_E_Stop;
        this.state.sig_4e_src = SIG_E_Stop;
    }
    // The Route Is NOT Occupied
    else {
        // Switches
        this.state.sw_5_src = SW_U_E_Lined;
        this.state.sw_3_src = CX_225_R_Lined;
        this.state.sw_1_src = CX_135_Lined_Bottom;

        // Signals
        this.state.sig_2w1_src = SIG_W_Stop;
        this.state.sig_2w2_src = SIG_W_Stop;
        this.state.sig_4w_src = SIG_W_Stop;
        this.state.sig_2e_src = SIG_E_Stop;
        this.state.sig_4e_src = SIG_E_Clear;
    }
}
else if (this.state.routes[i] === "W_3_2__|
__4_ridgewood_bt") {
    // Tail Tracks
    this.state.tail_3_e = color_1;
    this.state.tail_2_w = color_1;

    // The Route Is Occupied
    if (this.state.occupied_1) {
        // Switches
        this.state.sw_5_src = SW_U_E_R_Occupied;
        this.state.sw_3_src = CX_225_R_Occupied;
        this.state.sw_1_src = CX_135_Occupied_Bottom;

        // Signals
        this.state.sig_2w1_src = SIG_W_Stop;
        this.state.sig_2w2_src = SIG_W_Stop;
    }
}

```

```

        this.state.sig_4w_src = SIG_W_Stop;
        this.state.sig_2e_src = SIG_E_Stop;
        this.state.sig_4e_src = SIG_E_Stop;
    }
    // The Route Is NOT Occupied
    else {
        // Switches
        this.state.sw_5_src = SW_U_E_R_Lined;
        this.state.sw_3_src = CX_225_R_Lined;
        this.state.sw_1_src = CX_135_Lined_Bottom;

        // Signals
        this.state.sig_2w1_src = SIG_W_Stop;
        this.state.sig_2w2_src = SIG_W_Clear;
        this.state.sig_4w_src = SIG_W_Stop;
        this.state.sig_2e_src = SIG_E_Stop;
        this.state.sig_4e_src = SIG_E_Stop;
    }
}
else if (this.state.routes[i] === "E_2_3__|__3_bt_nysw") {
    // Tail Tracks
    this.state.tail_3_e = color_2;
    this.state.tail_2_w = color_2;

    // The Route Is Occupied
    if (this.state.occupied_2) {
        // Switches
        this.state.sw_5_src = SW_U_E_R_Occupied;
        this.state.sw_3_src = CX_225_R_Occupied;
        this.state.sw_1_src = CX_135_Occupied_Bottom;

        // Signals
        this.state.sig_2w1_src = SIG_W_Stop;
        this.state.sig_2w2_src = SIG_W_Stop;
        this.state.sig_4w_src = SIG_W_Stop;
        this.state.sig_2e_src = SIG_E_Stop;
        this.state.sig_4e_src = SIG_E_Stop;
    }
    // The Route Is NOT Occupied
    else {
        // Switches
        this.state.sw_5_src = SW_U_E_R_Lined;
        this.state.sw_3_src = CX_225_R_Lined;
        this.state.sw_1_src = CX_135_Lined_Bottom;

        // Signals
        this.state.sig_2w1_src = SIG_W_Stop;
        this.state.sig_2w2_src = SIG_W_Stop;
        this.state.sig_4w_src = SIG_W_Stop;
        this.state.sig_2e_src = SIG_E_Stop;
    }
}

```

```

        this.state.sig_4e_src = SIG_E_Clear;
    }
}
else if (this.state.routes[i] === "W_2_1__|
__3_ridgewood_bt") {
    // Tail Tracks
    this.state.tail_2_e = color_2;
    this.state.tail_1_w = color_2;

    // The Route Is Occupied
    if (this.state.occupied_2) {
        // Switches
        this.state.sw_3_src = CX_225_Occupied_Bottom;
        this.state.sw_1_src = CX_135_R_Occupied;

        // Signals
        this.state.sig_2w1_src = SIG_W_Stop;
        this.state.sig_2w2_src = SIG_W_Stop;
        this.state.sig_4w_src = SIG_W_Stop;
        this.state.sig_2e_src = SIG_E_Stop;
        this.state.sig_4e_src = SIG_E_Stop;
    }
    // The Route Is NOT Occupied
    else {
        // Switches
        this.state.sw_3_src = CX_225_Lined_Bottom;
        this.state.sw_1_src = CX_135_R_Lined;

        // Signals
        this.state.sig_2w1_src = SIG_W_Stop;
        this.state.sig_2w2_src = SIG_W_Stop;
        this.state.sig_4w_src = SIG_W_Clear;
        this.state.sig_2e_src = SIG_E_Stop;
        this.state.sig_4e_src = SIG_E_Stop;
    }
}
else if (this.state.routes[i] === "E_1_2__|
__2_bt_pascack") {
    // Tail Tracks
    this.state.tail_2_e = color_1;
    this.state.tail_1_w = color_1;

    // The Route Is Occupied
    if (this.state.occupied_1) {
        // Switches
        this.state.sw_3_src = CX_225_Occupied_Bottom;
        this.state.sw_1_src = CX_135_R_Occupied;

        // Signals
        this.state.sig_2w1_src = SIG_W_Stop;

```

```

        this.state.sig_2w2_src = SIG_W_Stop;
        this.state.sig_4w_src = SIG_W_Stop;
        this.state.sig_2e_src = SIG_E_Stop;
        this.state.sig_4e_src = SIG_E_Stop;
    }
    // The Route Is NOT Occupied
    else {
        // Switches
        this.state.sw_3_src = CX_225_Lined_Bottom;
        this.state.sw_1_src = CX_135_R_Lined;

        // Signals
        this.state.sig_2w1_src = SIG_W_Stop;
        this.state.sig_2w2_src = SIG_W_Stop;
        this.state.sig_4w_src = SIG_W_Stop;
        this.state.sig_2e_src = SIG_E_Clear;
        this.state.sig_4e_src = SIG_E_Stop;
    }
}
}
}
// ---- END set_route_drawings() ----

/**
 * set_switch_img()
 * @summary Changes image sources for the switches, depending on
switch status
 *
 * @description This function uses the data passed in through
status from the CTC classes and
 * shows if the switches are reversed or not on the screen, by
changing the image
 * source files, to the correct .png file respectively
 */
set_switch_images() {
    // Set SW #1
    // SW #1 Reversed
    if (this.state.sw_1) {
        this.state.sw_1_src = CX_135_R;
    }
    // SW #1 Normal
    else {
        this.state.sw_1_src = CX_135;
    }

    // Set SW #3
    // SW #3 Reversed
    if (this.state.sw_3) {
        this.state.sw_3_src = CX_225_R;
    }
}

```

```

        // SW #3 Normal
        else {
            this.state.sw_3_src = CX_225;
        }

        // Set SW #5
        // SW #5 Reversed
        if (this.state.sw_5) {
            this.state.sw_5_src = SW_U_E_R;
        }
        // SW #5 Normal
        else {
            this.state.sw_5_src = SW_U_E;
        }
    }
    // ---- END set_switch_image() ----

    /**
     * @summary Function to reset the signal images and track colors
     *
     * @description This function is need, because if the player was
to remove a route,
     * or when the train clears the interlocking nothing will clear
the route
     * the is displaying on the screen, even if it's gone in the
backend
    */
    reset_drawings() {
        this.state.tail_1_w = Empty;
        this.state.tail_2_w = Empty;
        this.state.tail_1_e = Empty;
        this.state.tail_2_e = Empty;
        this.state.tail_3_e = Empty;

        this.state.sig_2w1_src = SIG_W;
        this.state.sig_2w2_src = SIG_W;
        this.state.sig_4w_src = SIG_W;
        this.state.sig_2e_src = SIG_E;
        this.state.sig_4e_src = SIG_E;
    }
    //---- END reset_drawings() ----
}

// Export the interlocking to be drawn on the screen
export default BT;

```