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

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

// Import Images
// Switch Images
import SW_U_W from '../../../../public/images/SW_U_W.png';
import SW_U_W_Lined from '../../../../public/images/SW_U_W_Lined.png';
import SW_U_W_Occupied from '../../../../public/images/
SW_U_W_Occupied.png';
import SW_U_W_R from '../../../../public/images/SW_U_W_R.png';
import SW_U_W_R_Lined from '../../../../public/images/
SW_U_W_R_Lined.png';
import SW_U_W_R_Occupied from '../../../../public/images/
SW_U_W_R_Occupied.png';
import SW_D_W from '../../../../public/images/SW_D_W.png';
import SW_D_W_Lined from '../../../../public/images/SW_D_W_Lined.png';
import SW_D_W_Occupied from '../../../../public/images/
SW_D_W_Occupied.png';
import SW_D_W_R from '../../../../public/images/SW_D_W_R.png';
import SW_D_W_R_Lined from '../../../../public/images/
SW_D_W_R_Lined.png';
import SW_D_W_R_Occupied from '../../../../public/images/
SW_D_W_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';
```

```javascript
const Green = '#75fa4c';
const Red = '#eb3323';



/**
 * The React JSX Component Class for the Harriman Interlocking
 *
 * This class is a JSX React Component for the Harriman 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 Harriman extends Component {
    /**
     * State
     * @summary Object that holds the state or status information for
 the component
     *
     * 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_21: this.props.sw_21,
        sw_32: this.props.sw_32,
        // Image File for the switch — Will change depending on route
        sw_21_src: SW_U_W,
        sw_32_src: SW_D_W,
        // Image File for the signals — Will change depending on route
        sig_1w_src: SIG_W,
        sig_1e_src: SIG_E,
        sig_2e_src: SIG_E,
        sig_3e_src: SIG_E,
        // Colors for tail tracks — Will change depending on route
        tail_1_w: Empty,
        tail_2_w: Empty,
        tail_ind: Empty,
        tail_e: Empty,
        // Information For Interlocking Routes
        occupied: this.props.status.occupied,
        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_21: nextProps.status.sw_21,
            sw_32: nextProps.status.sw_32,
            occupied: nextProps.status.occupied,
            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_img();
        // 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="harriman_title">CP HARRIMAN</div>
                <div className="harriman_milepost">MP 45.0JS</div>
                {/* West Side Tail Tracks */}
                <div className="harriman_1_west" style={{background:
this.state.tail_1_w}}></div>
                <div className="harriman_2_west" style={{background:
this.state.tail_2_w}}></div>
                <div className="harriman_industrial"
style={{background: this.state.tail_ind}}></div>
                {/* Switches */}
                <div className="harriman_SW_21"
onClick={this.props.throw_sw_21}><img src={this.state.sw_21_src}/></
div>
```

```jsx
                <div className="harriman_SW_32"
onClick={this.props.throw_sw_32}><img src={this.state.sw_32_src}/></
div>
                {/* East Side Tail Tracks */}
                <div className="harriman_1_east" style={{background:
this.state.tail_e}}></div>
                {/* Signals */}
                <div className="harriman_sig_2e"
onClick={this.props.click_sig_2e}><img src={this.state.sig_2e_src}/></
div>
                <div className="harriman_sig_1e"
onClick={this.props.click_sig_1e}><img src={this.state.sig_1e_src}/></
div>
                <div className="harriman_sig_3e"
onClick={this.props.click_sig_3e}><img src={this.state.sig_3e_src}/></
div>
                <div className="harriman_sig_1w"
onClick={this.props.click_sig_1w}><img src={this.state.sig_1w_src}/></
div>
            </div>
        );
    }
    // ---- END render() ----

    /**
     * @summary Sets the drawing for the route through the
interlocking
     *
     * 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() {
        // Setting the color of the tracks depending on if the
interlocking in occupied or not
        let color = null;
        if (this.state.occupied) {
            color = Red;
        }
        else {
            color = Green;
        }
        for (let i = 0; i < this.state.routes.length; i++) {
            if (this.state.routes[i] === "W_1_1__|__1_valley_harriman"
|| this.state.routes[i] === "E_1_1__|__1_harriman_sterling") {
                // Tail Tracks
                this.state.tail_1_w = color;
                this.state.tail_e = color;

                // The Route Is Occupied
```

```javascript
                if (this.state.occupied) {
                    // Switch Images
                    this.state.sw_21_src = SW_U_W_Occupied;
                    this.state.sw_32_src = SW_D_W_Occupied;

                    // Signal Images
                    this.state.sig_1w_src = SIG_W_Stop;
                    this.state.sig_1e_src = SIG_E_Stop;
                    this.state.sig_2e_src = SIG_E_Stop;
                    this.state.sig_3e_src = SIG_E_Stop;
                }
                // The Route Is NOT Occupied
                else {
                    // Switch Images
                    this.state.sw_21_src = SW_U_W_Lined;
                    this.state.sw_32_src = SW_D_W_Lined;

                    // Signal Images
                    // West Bound Signals
                    if (this.state.routes[i] === "W_1_1__|
__1_valley_harriman") {
                        this.state.sig_1w_src = SIG_W_Clear;
                        this.state.sig_1e_src = SIG_E_Stop;
                        this.state.sig_2e_src = SIG_E_Stop;
                        this.state.sig_3e_src = SIG_E_Stop;
                    }
                    // East Bound Signals
                    else {
                        this.state.sig_1w_src = SIG_W_Stop;
                        this.state.sig_1e_src = SIG_E_Clear;
                        this.state.sig_2e_src = SIG_E_Stop;
                        this.state.sig_3e_src = SIG_E_Stop;
                    }
                }
            }
            else if (this.state.routes[i] === "W_1_2__|
__2_valley_harriman" || this.state.routes[i] === "E_2_1__|
__1_harriman_sterling") {
                // Tail Tracks
                this.state.tail_2_w = color;
                this.state.tail_e = color;

                // The Route Is Occupied
                if (this.state.occupied) {
                    // Switch Images
                    this.state.sw_21_src = SW_U_W_R_Occupied;
                    this.state.sw_32_src = SW_D_W_Occupied;

                    // Signal Images
                    this.state.sig_1w_src = SIG_W_Stop;
```

```javascript
                        this.state.sig_1e_src = SIG_E_Stop;
                        this.state.sig_2e_src = SIG_E_Stop;
                        this.state.sig_3e_src = SIG_E_Stop;
                    }
                    // The Route Is NOT Occupied
                    else {
                        // Switch Images
                        this.state.sw_21_src = SW_U_W_R_Lined;
                        this.state.sw_32_src = SW_D_W_Lined;

                        // Signal Images
                        // West Bound Signals
                        if (this.state.routes[i] === "W_1_2__|
__2_valley_harriman") {
                            this.state.sig_1w_src = SIG_W_Clear;
                            this.state.sig_1e_src = SIG_E_Stop;
                            this.state.sig_2e_src = SIG_E_Stop;
                            this.state.sig_3e_src = SIG_E_Stop;
                        }
                        // East Bound Signals
                        else {
                            this.state.sig_1w_src = SIG_W_Stop;
                            this.state.sig_1e_src = SIG_E_Stop;
                            this.state.sig_2e_src = SIG_E_Clear;
                            this.state.sig_3e_src = SIG_E_Stop;
                        }
                    }
                }
                else if (this.state.routes[i] === "W_1_3__|
__3_industrial_harriman" || this.state.routes[i] === "E_3_1__|
__1_harriman_sterling") {
                    // Tail Tracks
                    this.state.tail_ind = color;
                    this.state.tail_e = color;

                    // The Route Is Occupied
                    if (this.state.occupied) {
                        // Switch Images
                        this.state.sw_32_src = SW_D_W_R_Occupied;

                        // Signals
                        this.state.sig_1w_src = SIG_W_Stop;
                        this.state.sig_1e_src = SIG_E_Stop;
                        this.state.sig_2e_src = SIG_E_Stop;
                        this.state.sig_3e_src = SIG_E_Stop;
                    }
                    // The Route Is NOT Occupied
                    else {
                        // Switch Images
                        this.state.sw_32_src = SW_D_W_R_Lined;
```

```javascript
                    // Signal Images
                    // West Bound Signals
                    if (this.state.routes[i] === "W_1_3__|
__3_industrial_harriman") {
                        this.state.sig_1w_src = SIG_W_Clear;
                        this.state.sig_1e_src = SIG_E_Stop;
                        this.state.sig_2e_src = SIG_E_Stop;
                        this.state.sig_3e_src = SIG_E_Stop;
                    }
                    // East Bound Signals
                    else {
                        this.state.sig_1w_src = SIG_W_Stop;
                        this.state.sig_1e_src = SIG_E_Stop;
                        this.state.sig_2e_src = SIG_E_Stop;
                        this.state.sig_3e_src = SIG_E_Clear;
                    }
                }
            }
        }
    }
    // ---- END set_route_drawings() ----

    /**
     * set_switch_img()
     * @summary Changes image sources for the switches, depending on
switch status
     *
     * 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 respectivly
     */
    set_switch_img() {
        // Set SW #21
        // SW #21 Reversed
        if (this.state.sw_21) {
            this.state.sw_21_src = SW_U_W_R;
        }
        // SW #21 Normal
        else {
            this.state.sw_21_src = SW_U_W;
        }

        // Set SW #32
        // SW #32 Reversed
        if (this.state.sw_32) {
            this.state.sw_32_src = SW_D_W_R;
        }
```

```
        // SW #32 Normal
        else {
            this.state.sw_32_src = SW_D_W;
        }
    }
    // ---- END set_switch_img() ----

    /**
     * @summary Function to reset the signal images and track colors
     *
     * 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.sig_1w_src = SIG_W;
        this.state.sig_1e_src = SIG_E;
        this.state.sig_2e_src = SIG_E;
        this.state.sig_3e_src = SIG_E;

        this.state.tail_1_w = Empty;
        this.state.tail_2_w = Empty;
        this.state.tail_ind = Empty;
        this.state.tail_e = Empty;
    }
    //---- END reset_drawings() ----
}

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