Module Pattern

This is one of the most commonly used design patterns. It helps in defining wrapping related function in single object and expose only required information outside object.

// module pattern structure

(function(){

        // declare private variables and functions

        return{

            // declare public variables and functions

        }

    }

)();

Example of module pattern

let UIHandler = function(){

    // This is a private function

    let setText = function(newText){

        let txtElement = document.querySelector(".txtBlock");

        txtElement.textContent = newText;

    }

    return{

        // returning new function which is publically accessible. This

        // function calls private function required.

        updateText(newText){

            setText(newText);

        }

    }

}

// using object we can call public functions exposed.

let ui = new UIHandler();

ui.updateText("Hello this is new text !!!")

# Revealing module pattern

Return private function reference rather than calling it internally through public function. Which relveals private function externally.

//============= Revealing module pattern ======================

let UIHandler = function(){

    // This is a provate function

    let setText = function(newText){

        let txtElement = document.querySelector(".txtBlock");

        txtElement.textContent = newText;

    }

    return{

        // returning private function reference which then can be accessed

        //  publically through object. This is different from normal module approach

        // where we call private functions rather than returning their reference

            updateText : setText

    }

}

// using object we can call public functions exposed.

let ui = new UIHandler();

ui.updateText("Hello this is new text !!!")